POLITECNICO MILANO 1863

PowerEnjoy

Project Plan

Version 1.0

Software Engineering 2 (A.A. 2016/2017)

- Simone Boglio (mat. 772263)
- Lorenzo Croce (mat. 807833)

Summary

1. Introduction

1.1. Purpose and Scope

The main purpose of the project plan document is the analysis of the complexity and risks of the PowerEnjoy and create an estimate of the budget, the efforts, the time, and the resources allocation to define a plan to develop the entire project.
The project plan in general consists in:

- Identification of the feature;
- Identification of the activities needed to complete these feature;
- Estimate a budget;
- Estimate the effort and the time of the activities;
- Estimate the resource allocation for the activities;
- Develop a schedule;
- Identification of the risks;

For this scope, we use two method of analysis: Functional Point analysis and the COCOMOII analysis.

1.2. Definitions, Acronyms and Abbreviations

The following are used in this document:

- FP: Function Point
- JEE: Java Enterprise Edition
- SLOC: Source Lines of Code
- KSLOC: Thousands of SLOC
- EAF: Effort Adjustment Factor
- API: Application Programming Interface
- COCOMO: COnstructive COst MOdel
- RASD: Requirements Analysis and Specification Document
- DD: Design Document
- ITPD: Integrated Test Planning Document
- CMM: (Capability Maturity Model)
- EAF: Effort Adjustment Factor

1.3. Reference Documents

- The COCOMOII Model Definition Manual
- The Function Points complexity evaluation tables
- Requirement Analysis and Specification Document (RASD) of PowerEnjoy
- Design Document (DD) of PowerEnjoy

2. Function Point Analysis

The Function Point analysis allows to estimate the size of the project by analyzing the set functionalities provided by the software.

The effort for the project development grows with the growth of the number of data structures, inputs and outputs, inquiries and external interfaces. A weight is associated with each of these functionalities and the total effort is obtained adding weights of partial values.

There are five User Function types and they are represented on the table below.

Table 1. User Function Types

Function Point	Description
External Input (EI)	Count each unique user data or user control input type that enters the external boundary of the software system being measured.
External Output (EO)	Count each unique user data or control output type that leaves the external boundary of the software system being measured.
Internal Logical File (ILF)	Count each major logical group of user data or control information in the software system as a logical internal file type. Include each logical file (e.g., each logical group of data) that is generated, used, or maintained by the software system.
External Interface Files (EIF)	Files passed or shared between software systems should be counted as external interface file types within each system.
External Inquiry (EQ)	Count each unique input-output combination, where input causes and generates an immediate output, as an external inquiry type.

Each of these function types is characterized by a complexity levels and these levels are associated to a set of weights. The table below shows the complexity levels and their weights.

Table 3. UFP Complexity Weights

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

The function classification into the complexity level depending on the number of Data Elements, File Types and recorder Elements. The tables below explain this classification.

Table 2. FP Counting Weights

For Internal Logical Files and External Interface Files

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

For External Output and External Inquiry

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

For External Input

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

2.1. Internal Logical Files

The Internal Logical Files that we have identified are:

- **Users:** the system saves information about the users like ID, E-Mail, username, password, telephone number and credit card number.
- **Reservations:** the system saves information about each reservation like ID, IDUser, IDCar time of reservation, status, fee.
- **Rides:** the system saves information about the rides like ID, IDReservation, IDPayment, time of start, time of end, current charge.
- **Areas:** the system saves information about the safe areas like ID and position. The system must save also information about special areas that are a subset of safe areas. For each special area system saves information like number of power grid station and how many of these are available.
- **Cars:** the system saves information about the cars like ID, status, position and a flag that identify if it is plugged into a power grid.
- **Payments:** the system saves information about the payment like ID, the amount.
- **API pemrmission:** the system saves information about the API permission that allows developer to implement other user experience.

Internal Logical File	**Complexity**	**FP weight**
Users	Low	7
Reservations	Low	7
Rides	Low	7
Areas	Low	7
Cars	Low	7
Payments	Low	7
API information	Average	10
Total:		52

2.2. External Input

The External Inputs that we have identified are:

- **Signin:** this is an average operation that uses the Request Manager and the Account Manager and for this reason it has an average complexity level and it contributes 4FPs.
- **Login/Logout:** these are average operation like the signin and they have an average complexity level and they contribute 2x4FPs.
- **Search Car:** this is a complex operation because it involves in many components and for this reason it has a high complexity level and it contributes 6FPs.
- **Reserve Car:** like the search car operation also this functionality has a high complexity level and it contributes 6FPs.
- **Send Unlock:** this functionality involves in many components and it has a high complexity level and it contributes 6FPs.
- **Edit Profile:** this is a simple operation and it has a low complexity level and it contributes 3FPs.
- **Retrieve profile info:** this is a simple operation and it has a low complexity level and it contributes 3FPs.
- **Retrieve reservation info:** this is a simple operation and it has a low complexity level and it contributes 3FPs.

External Input	Complexity	FP weight
Signin	Average	4
Login/Logout	Average	2x4
Search car	High	6
Reserve car	High	6
Send unlock	High	6
Edit profile	Low	3
Retrieve profile info	Low	3
Retrieve reservation info	Low	3
Total		39

2.3. External Output

The External Outputs that we have identified are:

- **Email/SMS:** this functionality allows to send back to user his password at the registration moment (if the user chose this method to receive the password) and it has a low level of complexity and it contributes 2x4FPs.
- **Send ride information:** this functionality allows to send to the cars the information about the ride and it has an average complexity level and it contributes 5FPs.
- **Send safe areas:** this functionality allows to send to the cars the information about the safe area and it has an average complexity level and it contributes 5FPs.

External Output	Complexity	FP weight
SMS	Low	4
Email	Low	4
Send ride information	Average	5
Send safe areas	Average	5
Total		18

2.4. External Inquiry

The External Inquiries that we have identified are:

- **Reservation/profile info:** these are directly operations and for this reason they have a low complexity level and they contribute 2x3FPs.
- **Database query:** this functionality allows to query the database and it has an average complexity level and it contributes 4FPs.

External Inquiry	Complexity	FP weight
Reservation info	Low	3
Profile info	Low	3
Database query	Average	4
Total		10

2.5. External Interface Files

The External Interface Files that we have identified are:

- **Google Maps:** the interaction with map service implies the exchange of a high number of information and their computation is enough complex, for these reasons it has a high complexity level and it contributes 10FPs.
- **Payment service:** the interaction with the payment service has an average complexity level and it contributes 7FPs.
- **Green e-box:** the interaction with all the green e-box needs many calculations and it has a high complexity level and it contributes 10FPs.

External Input File	**Complexity**	**FP weight**
Google Maps	High	10
Payment Service	Average	7
Green e-box	High	10
Total		27

2.6. Total of Function Point

Type	**FP weight**
Internal Logical File	52
External Input	39
External Output	18
External Inquiry	10
External Interface File	27
Total	146

Considering that the platform used for this project is Java Enterprise Edition, we can use the as conversion factor 46 and 67 (lower bound and upper bound) to estimate the value of the lines of code needed for the development of the application.

These conversion factor can be found at:
http://www.qsm.com/resources/function-point-languages-table

The estimation is:

- **Lower bound:** 146FPs x 46 = 6716 SLOC
- **Upper bound:** 146FPs x 67 = 9782 SLOC

3. Cocomo II Analysis

In this section, we are going to use the COCOMO II model to estimate the cost and the effort needed to develop PowerEnjoy service.

3.1. Scale Factors

To evaluate the scale factor values we refer to the following official COCOMO II table:

Table 10. Scale Factor Values, SF_j, for COCOMO II Models

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	The estimated Equivalent Process Maturity Level (EPML) or					
	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

A brief description for each scale driver (for more details about each scale factor and how estimate it refer to tables from 11 to 15 in the official COCOMOII document):

- Precedentedness (PREC): it reflects 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 (FLEX): it reflects 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 request for the technology to be used, this value will be low.
- Risk resolution (RESL): it reflects the level of awareness and reactiveness with respect to risks. We perform a good risks analysis, so the value will be set to very high.
- Team cohesion (TEAM): it is an indicator of how well the team members know each other and work together in a cooperative way. For our team, the value is set to high.
- Process maturity (PMAT): we use the CMM (Capability Maturity Model) scale for determining this value. We estimate that our process reflects the Level 3 of the CMM scale, so the value for PMAT is set to high.

The total result of our evaluation is the following:

Scale Driver	**Factor**	**Value**
PREP	Low	4.96
FLEX	Low	4.05
RESL	Very High	1.41
TEAM	High	2.19
PMAT	High	3.12
Total		15.73

3.2. Cost Drivers

3.2.1. Product Factors

3.2.1.1. Required Software Reliability (RELY)

Since all the PowerEnjoy car service need the software to work and all the cars depend from the central system, a failure of the software determines a high financial loss for the company. Also, there is not risk to human life in case of software failure since the car CU and engine are independent from the central system.

RELY	
Rating Level	High
Effort Multiplier	1.10

3.2.1.2. Database Size (DATA)

The database size is not easy evaluable; we estimate this value considering the SLOC value and a reasonable byte dimension value for the DB (2<GB<5).
The D/P (TestingDBbytes/SLOC) is between 100 and 1000, from this we can evaluate the DATA values.

DATA	
Rating Level	High
Effort Multiplier	1.14

3.2.1.3. Product Complexity (CPLX)

The application need to manage lot of thing such: the reservations for the car, the current rides, the remote communication with all the cars of the car sharing service PowerEnjoy.
These things involve in a lot of concurrent task, callbacks and others; in according with the Component Complexity Rating Levels table of COCOMOII we estimate CPLX as very high.

CPLX	
Rating Level	Very High
Effort Multiplier	1.34

3.2.1.4. Developed for Reusability (RUSE)

The reusability requirements are limited in scope to the project itself, so the RUSE cost is set to nominal.

RUSE	
Rating Level	Nominal
Effort Multiplier	1.00

3.2.1.5. Documentation Match to Life-Cycle Needs (DOCU)

The documentation for this project is right-sized to life-cycle needs.

DOCU	
Rating Level	Nominal
Effort Multiplier	1.00

3.2.2. Platform Factors

3.2.2.1. Execution Time Constraint (TIME)

Since we need to monitor continually the cars ant the actual rides we expect a high use of available execution time (between 50% and 70%).

TIME	
Rating Level	High
Effort Multiplier	1.11

3.2.2.2. Main Storage Constraints (STOR)

Since a normal hard disk can contain TB of data and our system don't continue to expand its data in a relevant way we set this value low as possible.

STOR	
Rating Level	Nominal
Effort Multiplier	1.00

3.2.2.3. Platform Volatility (PVOL)

About the hardware system we don't expect change very often.
The software doesn't need continuous change, maybe just the client's application can request software update to be aligned with the update or new release of the OS system where it runs (Android and iOS).

PVOL	
Rating Level	Nominal
Effort Multiplier	1.00

3.2.3. Personnel Factors

3.2.3.1. Analyst Capability (ACAP)

We think our team had been conducted a good analysis and design analysis so we set this parameter to high.

ACAP	
Rating Level	High
Effort Multiplier	0.85

3.2.3.2. Programmer Capability (PCAP)

We don't have implemented this project, so we just estimate this value.
We have already developed some little project with different technologies but we never develop high real level project so we set this parameter to nominal.

PCAP	
Rating Level	Nominal
Effort Multiplier	1.00

3.2.3.3. Personnel Continuity (PCON)

Not relevant in our case since there is not personnel turnover. We set this value to nominal.

PCON	
Rating Level	Nominal
Effort Multiplier	1.00

3.2.3.4. Applications Experience (APEX)

We have some experience with development of little Java applications but we never develop a complex system of this kind. We set this value to low.

APEX	
Rating Level	Low
Effort Multiplier	1.10

3.2.3.5. Platform Experience (PLEX)

We don't have any previous experience with JEE so this can be problematic at the start, also we don't have experience with distributed middleware.
We have some little experience with DB, server side development and graphic user interfaces.

PLEX	
Rating Level	Low
Effort Multiplier	1.09

3.2.3.6. Language and Tool Experience (LTEX)

We have some experience with JSE but no experience with JEE.

We have some little experience with DB, server side development and graphic user interfaces.

LTEX	
Rating Level	Low
Effort Multiplier	1.09

3.2.4. Project Factors

3.2.4.1. Use of Software Tools (TOOL)

Our application environment is well integrated, complete and mature.

TOOL	
Rating Level	High
Effort Multiplier	0.90

3.2.4.2. Multisite Development (SITE)

We live in two different cities but we communicate with VOIP calls, email and chat so we set this value to high.

SITE	
Rating Level	High
Effort Multiplier	0.93

3.2.4.3. Required Development Schedule (SCED)

Although our efforts were well distributed over the available time, the definition of the RASD and DD require a consistent amount of time (more than we expected), for this reason we set this parameter to high.

SCED	
Rating Level	High
Effort Multiplier	1.00

3.2.5. Total Cost Drivers

In the following table, there are all the cost driver that we discuss above with the assigned value:

Cost Driver	**Factor**	**Value**
RELY	High	1.10
DATA	High	1.14
CPLX	Very High	1.34
RUSE	Nominal	1.00
DOCU	Nominal	1.00
TIME	High	1.11
STOR	Nominal	1.00
PVOL	Nominal	1.00
ACAP	High	0.85
PCAP	Nominal	1.00
PCON	Nominal	1.00
APEX	Low	1.10
PLEX	Low	1.09
LTEX	Low	1.09
TOOL	High	0.90
SITE	High	0.93
SCED	High	1.00
EAF		1.7342

3.3. Effort Equation

The following equation give us the effort estimation in Person-Months (PM):

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

Where:

$A = 2.94$ = constant for COCOMOII

$EAF = 1.7342$ = product of all cost drivers (without SCED)

$KSLOC = SLOC * 0.001$

$E = 1.0673 = B + 0.01 * \sum(SF) = 0.91 + 0.01 * 15.73$

$B = 0.91$ = constant for COCOMOII

$\sum(SF) = 15.73$ = summation of all Scale Factors

The effort value has a lower bound of:

$$Effort_{L} = A * EAF * KSLOC^{E} = 2.94 * 1.7342 * 6.716^{1.0673} = 38.92 \approx 39\ PM$$

And an upper bound of:

$$Effort_{U} = A * EAF * KSLOC^{E} = 2.94 * 1.7342 * 9.782^{1.0673} = 58.15 \approx 59\ PM$$

3.4. Schedule Estimation

3.4.1. Approach 1: Estimate Size of the Team

The following equation give us an estimation of the duration for the time schedule:

$$Duration = 3.67 * Effort^{F} * (SCED\%/100)$$

Where:

$$F = 0.28 + 0.2\ (E - B) = 0.28 + 0.2 * (E - B) = 0.31146$$
$$SCED\% = 130\%\ (High\ value)$$

The duration (Upper and Lower bound) is:

$$Duration_{L} = 3.67 * Effort_{L}^{F} * (SCED\%/100) = 14.91\ months$$
$$Duration_{B} = 3.67 * Effort_{U}^{F} * (SCED\%/100) = 16.91\ months$$

The size of the team (Upper and Lower bound) is:

$$N_{people} = Effort / Duration$$
$$N_{people\text{-}L} = Effort_{L} / Duration_{L} = 2.61 \approx 3$$
$$N_{people\text{-}U} = Effort_{U} / Duration_{U} = 3.44 \approx 4$$

3.4.2. Approach 2: Estimate Duration with Team of 2 People

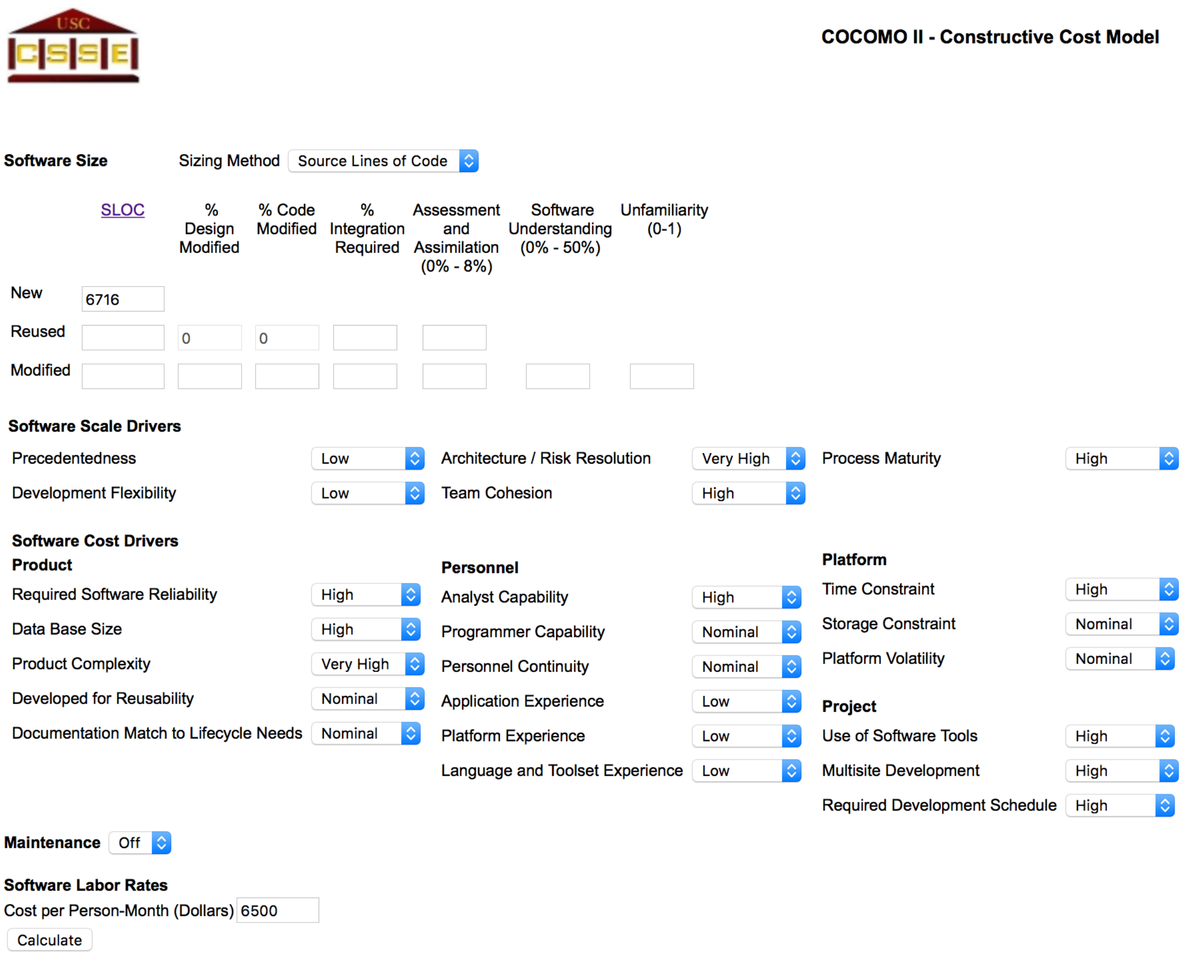
If we consider our team composed by 2 people, the duration (Upper and Lower bound) is:

$$Duration = Effort / N_{people}$$
$$Duration_{L} = Effort_{L} / 2 = 19.46\ months$$
$$Duration_{U} = Effort_{U} / 2 = 29.07\ months$$

3.4.3. Approach 3: Analysis with COCOMOII Automated Tool

A second analysis is here presented; it has been done with the help of an online tool (http://csse.usc.edu/tools/COCOMOII.php), where it is easier and immediate to evaluate the difference in effort and scheduling.

The setting of the analysis tool:



The output of the analysis tool:

Results

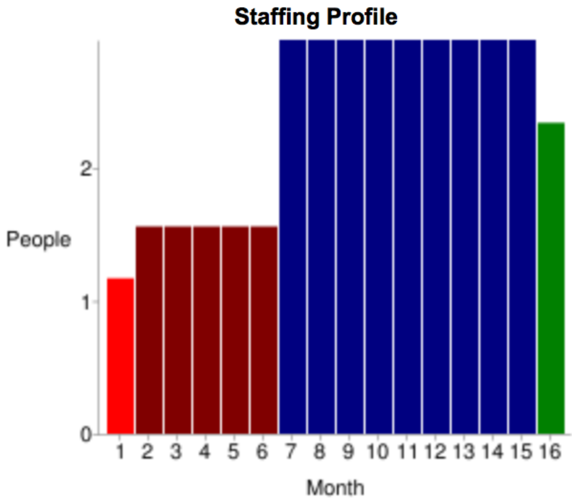
Software Development (Elaboration and Construction)

Effort = 38.9 Person-months
Schedule = 16.0 Months
Cost = $253017

Total Equivalent Size = 6716 SLOC

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	2.3	2.0	1.2	$15181
Elaboration	9.3	6.0	1.6	$60724
Construction	29.6	10.0	3.0	$192294
Transition	4.7	2.0	2.3	$30362



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

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.3	1.1	3.0	0.7
Environment/CM	0.2	0.7	1.5	0.2
Requirements	0.9	1.7	2.4	0.2
Design	0.4	3.4	4.7	0.2
Implementation	0.2	1.2	10.1	0.9
Assessment	0.2	0.9	7.1	1.1
Deployment	0.1	0.3	0.9	1.4

The estimated size of the team is:

$$N_{people} = Effort / TimeSchedule = 2.43 \approx 3$$

4. Task Identification

The tasks are the activities which must be developed to reach the project goals. These tasks are the same that are represented in the RASD, DD and ITPD.
Some tasks need to be developed before other and the table below shows the dependencies between the tasks and their development time.

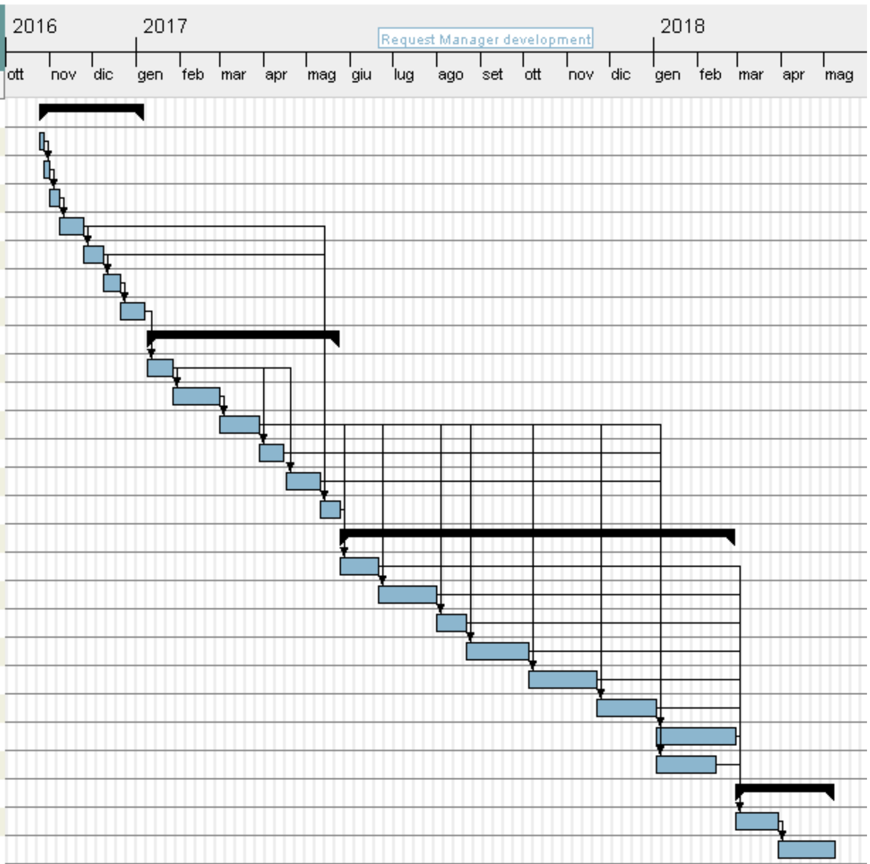
ID	Task	Effort (person-days)	Duration (days)	Dependencies
T1	Stakeholders identification	5	3	
T2	Actors identification	4	2	T1
T3	Goals identification	10	5	T2
T4	Requirements identification	25	13	T3
T5	Use Case and scenarios	20	10	T4
T6	Class diagrams	15	8	T5
T7	Alloy model	25	13	T6
T8	Components and interfaces	28	14	T7
T9	Deployment architecture	23	23	T8
T10	Runtime units	20	20	T9
T11	Sequence diagrams	25	13	T8
T12	Algorithm design	36	18	T8
T13	Mock-up and UX diagrams	20	10	T5
T14	View Client development	38	19	T10, T11, T12, T13
T15	Request manager development	57	29	T10, T11, T12
T16	Account manager development	30	15	T10, T11, T12
T17	Reservation manager development	64	32	T10, T11, T12
T18	Ride manager development	67	34	T10, T11, T12
T19	Car manager development	60	30	T10, T11, T12
T20	Area manager development	40	40	T10, T11, T12
T21	Payment manager development	30	30	T10, T11, T12
T22	Integration testing strategy	43	22	T14 – T21
T23	Integration testing execution	55	28	T22

The effort person-days can be calculated using a factor that represent the number of working day in a month and it is equal to 19. So, the 38.9 person-months calculated by COCOMOII is equal to 740 person-days.
If we sum all the effort values represented in the table above, we obtain 740 person-days.
The duration is calculated dividing the effort by the number of workers. Our group's worker is composed by only 2 people.
The assignment of the workers to each task is done keeping in mind the dependencies, when two activities can be developed concurrently and its effort isn't very high the workers can work in a parallel way.

5. Task Schedule

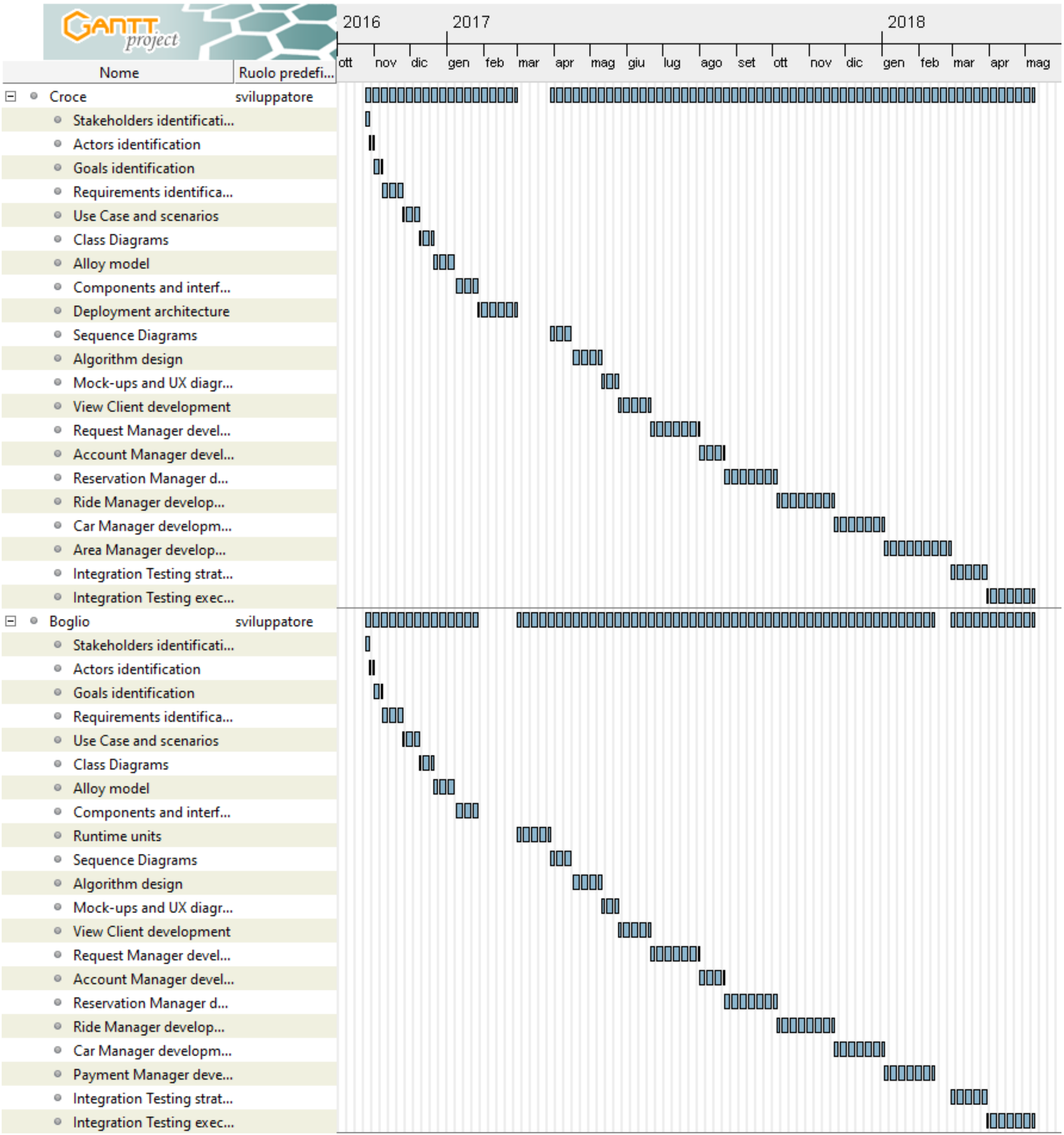
The image below shows an example of the possible high level project schedule.
The activities and their duration are the same we can found at the previous task identification table. They are grouped into four main groups: RASD, DD, DEVELOPMENT and ITPD.

Nome	Data d'inizio	Data di fine
RASD	25/10/16	06/01/17
Stakeholders identification	25/10/16	27/10/16
Actors identification	28/10/16	31/10/16
Goals identification	01/11/16	07/11/16
Requirements identification	08/11/16	24/11/16
Use Case and scenarios	25/11/16	08/12/16
Class Diagrams	09/12/16	20/12/16
Alloy model	21/12/16	06/01/17
DD	09/01/17	24/05/17
Components and interfaces	09/01/17	26/01/17
Deployment architecture	27/01/17	28/02/17
Runtime units	01/03/17	28/03/17
Sequence Diagrams	29/03/17	14/04/17
Algorithm design	17/04/17	10/05/17
Mock-ups and UX diagrams	11/05/17	24/05/17
DEVELOPMENT	25/05/17	27/02/18
View Client development	25/05/17	20/06/17
Request Manager development	21/06/17	31/07/17
Account Manager development	01/08/17	21/08/17
Reservation Manager development	22/08/17	04/10/17
Ride Manager development	05/10/17	21/11/17
Car Manager development	22/11/17	02/01/18
Area Manager development	03/01/18	27/02/18
Payment Manager development	03/01/18	13/02/18
ITPD	28/02/18	08/05/18
Integration Testing strategy	28/02/18	29/03/18
Integration Testing execution	30/03/18	08/05/18



6. Resource Allocation

The image below shows how the activities indicated in the previous tasks schedule plan are assigned to the workers.
The development team is composed by two developers (Boglio and Croce).



7. Risks Planning and Management

Our project during its development can run into some trouble and in this section, we try to find these possible problems (risks) and the way to manage them.
To manage a risk there are four ways:

- **Avoid:** it's the prevention of the risk before it happening
- **Mitigate:** if you can't avoid the risk, you can take some prevention measures to contain the damage
- **Transfer:** pay someone else to accept the risk for you
- **Accept:** if you can't do none of the previous strategy, you must accept the risk

Some risks of PowerEnjoy project can be the following:

- **Project delays** during development of the project
  To manage this risk, we can apply an avoid strategy. We can make a project schedule considering more days of the necessary.
- **Stakeholders bankrupt**
  To manage this risk, we can apply a mitigate strategy. We can request a part of the total cost of the whole project when we reach each milestone.
- **Wrong functionalities/user interfaces**
  To manage this risk, we can apply an avoid strategy. We can schedule more than one meeting with stakeholder to inform them of the current result and in case of mistakes or other we can easily fix the problem. Also, we can let the stakeholders to have an active role in the RASD and DD develop to avoid all possible misunderstand and problems in future parts of the project develop. Wrong user interface problems can be avoided with the use of mock-ups in early phase of the project.
- **Project team problem**: some members can quit or ill; we can also underestimate the quality and capability of a person in our team.
  To manage this risk we can split works and responsibilities to multiple people in the team and try to hire people with knowledgeable and flexible capability.
- **External component failure**: PowerEnjoy use as external services Google Maps, email provider, mobile provider and payment service.
  To manage this risk, we can apply a mitigate strategy. We can build our components in a modular way to substitute these external service, in case of their failure, with others that have the same functionalities.
- **Loss of work**
  To manage this risk, we can apply a mitigate strategy. We can plan daily backups to recover data in case of problems. We must be able to store these backups in a secure place, also in another physical place to avoid losses in case of fire, flooding etc.

8. Appendix

8.1. External References

- CMM model: https://en.wikipedia.org/wiki/Capability_Maturity_Model
- FP model: http://www.functionpointmodeler.com/fpm-infocenter/index.jsp?topic=%2Fcom.functionpointmodeler.fpm.help%2Fditafiles%2Fconcepts%2Fcon-86.html
- Risk management: https://opentextbc.ca/projectmanagement/chapter/chapter-16-risk-management-planning-project-management/

8.2. Software and Tools Used

- Microsoft Office Word to redact and format this document.
- GanttProject for the Gantt chart of the project schedule.

8.3. Hours of Work

- Simone Boglio 21 hours of work
- Lorenzo Croce 21 hours of work