Software Engineering 2 Project

Project Plan

PowerEnJoy

POLITECNICO
DI MILANO

Team: Nico Montali, Enrico Fini

Contents

Document Status

Document Title	Project Plan
Document ID	PowerEnjoy/PP/1.0
Authors	N. Montali, E. Fini
Version	1.0

Changelog

Version	Date	Changes
1.0	21/01/2017	Initial version

Introduction

1.1 Purpose and Scope of this document

This document provides a detailed Project Plan for PowerEnjoy project. Its main objective is to outline the development process, establishing the estimated costs, duration and necessary resources, through an estimation of efforts and complexity. This document is only intended as an help to the Project Manager to define real budget and resources.

In the first chapter, *Project Size, Effort and Costs estimation*, we try to estimate the project size and complexity using the Function Points analysis (providing estimated LOC), then proceeding to an estimation of costs and efforts.

In the second chapter, *Planned Schedule*, we will use the previously found data to outline a possible schedule for the project itself, in all its part.

In the third chapter, *Resource Allocation*, we will assign roles and tasks to our team members, using the previous schedule as an indicator.

In the last chapter, *Risk Management*, we will point out and analyze the possible risks in the development process, providing an analysis of damages they could generate and how they can be avoided.

1.2 Definitions, Acronyms and Abbreviations

- **FP:** Function Points
- **ILF:** Internal Logic File
- **ELF:** External Logic File
- **EI:** External Input

- **EQ:** External Inquiry
- **EO:** External Output
- **LOC:** Lines Of Code
- **SLOC:** Source Lines Of Code
- **IFPUG:** International Function Point Users Group IFPUG

1.3 Referenced Documents

- Project Plan Sample from myTaxiService
- Website Alvin Alexander Website for tables and definitions
- Website Function Point Languages Table for Javascript FP to SLOC conversion values.
- COCOMO online tool
- COCOMO Model Specifications

Project Size, Effort and Costs estimation

This chapter will provide in its first part a detailed analysis of the system functionalities, in order to estimate the size of the project in terms of LOC, using the FP approach. Instead, in the second part, we will use the COCOMO approach to estimate costs and efforts.

2.1 Size estimation using FP

For the FP estimation, we will use the IFPUG standard, so we will divide the analysis into 5 categories (ILF, ELF, EI, EQ, EO), and for each of them, provide a list of functions to be developed and their expected complexity. We will also present all the tables we used to estimate these values.

2.1.1 Internal Logic Files (ILF)

The ILF definition by IFPUG is:

"An ILF is a user-identifiable group of logically related data or control information maintained within the boundary of the application. The primary intent of an ILF is to hold data maintained through one or more elementary processes of the application being counted."

So, we will consider ILF all our data structures that resides inside the system itself. The tables used to estimate complexity and FP are quoted below.
All the ILF that we have identified are:

- **Drivers (extension of users):** we identified 6 DET (email, pwd, birthdate, status, license-info, payment-info) and 3 RET (valid drivers, no license drivers, no payment drivers). So we will consider it **LOW**.

RETS	Data Element Types (DETs)		
	1–19	20–50	51+
1	L	L	A
2 to 5	L	A	H
6 or more	A	H	H

Figure 2.1: Complexity estimation for ILF

Complexity	Points
Low	7
Average	10
High	15

Figure 2.2: Function Point estimation for ILF

- **Workers (extension of users):** we identified 6 DET (name, pwd, rfid, status, position, notification) and 2 RET (active workers, inactive). We will consider it **LOW**.
- **Admins (extension of users):** we identified 3 DET(username, pwd, 2faseed). We will consider it **LOW**.
- **Vehicle:** we identified 6 DET (plate, status, battery, position, last-interaction, current-drive-id, issue) and 3 RET (in use, maintenance, available). We will consider it **LOW**.
- **Reservation:** we identified 3 DET(vehicle, timestamp, driver). We will consider it **LOW**.
- **Drive:** we identified 13 DET(vehicle, driver, reservation, start-timestamp, stop-timestamp, start-position, stop-position, length, occupants, last-battery, report, amount, discounts) and 5 RET (driving, stopped, payed, with report, with discounts). We will consider it **AVG**.
- **Payment:** we identified 5 DET (user, drive, timestamp, amount, method). We will consider it **LOW**.
- **Task:** we identified 5 DET (worker, vehicle, timestamp, description, priority). We will consider it **LOW**.

The total FP from ILF is then 59, as shown in the review Table 2.1.

ILF	Complexity	FPs
Drivers	LOW	7
Workers	LOW	7
Admins	LOW	7
Vehicle	LOW	7
Reservation	LOW	7
Drive	AVG	10
Payment	LOW	7
Task	LOW	7
Total		59

Table 2.1: Total FP from ILF

2.1.2 External Logic Files (ELF)

The ELF definition by IFPUG is:

"An external interface file (EIF) is a user identifiable group of logically related data or control information referenced by the application, but maintained within the boundary of another application. The primary intent of an EIF is to hold data referenced through one or more elementary processes within the boundary of the application counted. This means an EIF counted for an application must be in an ILF in another application."

The only external data we have is the vehicle state, but since it is replicated by polling inside our database, it is considered again ILF (see above).

2.1.3 External Inputs (EI)

The EI definition by IFPUG is:

"An external input (EI) is an elementary process that processes data or control information that comes from outside the application boundary. The primary intent of an EI is to maintain one or more ILFs and/or to alter the behavior of the system."

We have many external input, so we will divide them per user type (as we have always done). As before, the tables used to estimate FP are copied here.

FTR's	Data Element Types (DET's)		
	1-4	5-15	16+
0-1	L	L	A
2	L	A	H
3 or more	A	H	H

Figure 2.3: Complexity estimation for EI

Drivers EI

- **Login/Logout:** the login phase need to access few components, and the operations of password checking (and possibly token generation) are quite standard. **LOW** complexity.
- **Signup:** The signup phase requires not only the insertion of the new user, but also the processing of license and payment infos. **AVG** complexity.
- **Reserve:** The reserve process access different components, need to check if a vehicle is available, check if the user has the minimum amount and check if user is authorized. **AVG** complexity.
- **Unlock:** This operation is quite simple, since it only need to send a request to the vehicle. **LOW** complexity.
- **CancelReservation:** This operation only requires to check that the reservation exists and if the time limit has not passed. **LOW** complexity.
- **Report:** This operations generates a report, that must be validated and then transformed into a Task possibily. **AVG** complexity.

Workers EI

- **Login:** Since the login is slightly different it will be counted again for the workers. The Logout instead is the same. **LOW** complexity.
- **ChangeMyState:** Only changes the state (available/ unavailable) of the worker. **LOW** complexity.
- **UpdateTask:** This operations changes the state of the task.**LOW** complexity.

Complexity	Points/Weight
Low	3
Average	4
High	6

Figure 2.4: Function Point estimation for EI

EI	Complexity	FPs
Driver-Login	LOW	3
Logout	LOW	3
Driver-Signup	AVG	4
Driver-Reserve	AVG	4
Driver-Unlock	LOW	3
Driver-CancelReservation	LOW	3
Driver-Report	LOW	3
Worker-Login	LOW	3
Worker-ChangeMyState	LOW	3
Worker-UpdateTask	LOW	3
Admin-Login	LOW	3
Admin-ManageWorker	LOW	3
Admin-MakeTask	LOW	3
Total		41

Table 2.2: Total FP from EI

Admin EI

- **Login:** Since the login is slightly different it will be counted again for the admins. The Logout instead is the same. **LOW** complexity.
- **ManageWorker:** Only applies to the workers dataset. **LOW** complexity.
- **MakeTask:** This operations only generates a task.**LOW** complexity.

The total FPs from EI are 41, as in the review Table 2.2.

FTR	Data Element Types (DET)		
	1–5	6–19	20+
0–1	L	L	A
2–3	L	A	H
4 or more	A	H	H

Figure 2.5: Complexity estimation for EO

Complexity	Points/Weight
Low	4
Average	5
High	7

Figure 2.6: Function Point estimation for EO

2.1.4 External Outputs (EO)

The definition of EO by IFPUG is:

"An external output (EO) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external output is to present information to a user through processing logic other than, or in addition to, the retrieval of data or control information . The processing logic must contain at least one mathematical formula or calculation, create derived data maintain one or more ILFs or alter the behavior of the system."

We only have 2 External Output, that are:

- **Push-Notification to Workers** Since it is only a communication of a new task assignment, we will assign **LOW** complexity to this.
- **Vehicle Polling** Vehicle polling is another primary operation of our system, but it is quite simple. **LOW** complexity.

The resulting FPs for EO are 8, as stated in the Table 2.3.

EO	Complexity	FPs
Worker push	LOW	4
Vehicle polling	LOW	4
Total		8

Table 2.3: Total FP from EO

FTRs	Data Element Types (DETs)		
	1–5	6–19	20+
0–1	L	L	A
2–3	L	A	H
4 or more	A	H	H

Figure 2.7: Complexity estimation for EQ

2.1.5 External Inquiry (EQ)

The definition of EQ by IFPUG is:

"An external inquiry (EQ) is an elementary process that sends data or control information outside the application boundary. The primary intent of an external inquiry is to present information to a user through the retrieval of data or control information from an ILF of EIF. The processing logic contains no mathematical formulas or calculations, and creates no derived data. No ILF is maintained during the processing, nor is the behavior of the system altered."

We only have 2 External Inquiries, that are:

- **Available Vehicles:** It needs to retrieve all the available vehicles nearby a position, so it also requires a distance computation to order the vehicles. **AVG** complexity.
- **Admin-GetStatistics:** It simply replies with the selected records given some filters, so **LOW** complexity.

The resulting FPs for EQ are 6, as stated in the Table 2.4.

Complexity	Points/Weight
Low	3
Average	4
High	6

Figure 2.8: Function Point estimation for EQ

EQ	**Complexity**	**FPs**
Worker push	LOW	3
Vehicle polling	LOW	3
Total		6

Table 2.4: Total FP from EQ

2.1.6 FP estimation recap

The totals of every level from before is copied here and a final total is provided.

Type	**FPs**
ILF	59
ELF	0
EI	41
EO	8
EQ	6
Total	114

The total of 114FPs will then be converted to SLOC using the parameter we found on the referenced website, since we are using NodeJS (and so JavaScript). The average SLOC for a FP in JS is 47. So the total SLOC of the system:

$$\textbf{SLOC} = \textbf{114} * \textbf{47} = \textbf{5358}$$

Using instead the maximum FP to SLOC conversion rate (JS: 63) we get an upper bound of

$$\textbf{SLOC} = \textbf{114} * \textbf{63} = \textbf{7182}$$

2.2 COCOMO estimation

We are going to list all our estimation on factors, and later use an online calculator compute the results. Extensive explanation of the factor types and their values are provided in the course lectures.

2.2.1 Scale Drivers

Precedentedness (PREC)

Since we have already developed different system that uses a similar architecture (NodeJS, Postgres, etc.) and there is no need for any innovative algorithm here, this factor will be set to **Very High**

Development Flexibility (FLEX)

The requirements where not so detailed and formal and where open to interpretation. Also, we had only another constraint, i.e. the API of the vehicles, that are considered already developed. We will set this to **Nominal**

Architecture / Risk Resolution (RESL)

As stated below in our risk analysis, the dangerous risks are only few, so we are setting this to **High**

Team Cohesion (TEAM)

Since we are long-time friends and also collaborated effectively in another project, this value will be set to **Very High**

Process Maturity (PMAT)

We will opt for a Level 3 maturity (Organization Level), so **High**

2.2.2 Cost Drivers

Required Software Reliability (RELY)

Because of the design of our vehicles, we exclude any threat to human life (our system cannot stop or control a vehicle). So we are setting this to "moderate, easily recoverable losses", **Nominal**

Data Base Size (DATA)

Since our test dataset will be very small, because there are not many different situations to test, we will set this to **Low**

Product Complexity (CPLX)

We will set this to **Nominal**, because our code will user simple callbacks, middle-wares etc., and also, in mobile development, we will user device-dependant operations that are not straight-forward.

Developed for Reusability (RUSE)

Since the components of the system are very specific to the problem, we will not take into account the Reusability of the code. The only component that could be reused is the AuthorizationManager, but it is probably better to use a third-party middleware like PassportJS instead of developing one inernally. Set to **Low**

Documentation Match to Life-Cycle Needs (DOCU)

The need of documentation is clearly fundamental, but since we are a team of 2 we will not provide a super-detailed documentation, but the right size. Set to **Nominal**

Execution Time Constraint (TIME)

Our system should interact with a lot of users together but with a not so difficult complexity. We expect a medium-high CPU usage. Set to **High**

Main Storage Constraint (STOR)

Our system won't probably need a lot of storage space, so we are leaving this to **Nominal**

Platform Volatility (PVOL)

The platform we use is composed of NodeJS (v6), Postgres (v9) that have a similar period between release (6 months for major). The vehicle on-board system instead will be updated very seldomly. Set to **Nominal**

Analyst Capability (ACAP)

Since we are not experienced analysts (requirements, design etc.) we will set this to **Low**

Programmer Capability (PCAP)

Instead we are quite good programmers, with a lot of developed team projects and a lot of lines of code. Set to **High**

Personnel Continuity (PCON)

Our team will probably remain the same, so set to **Very High**

Applications Experience (APEX)

Since we are dealing with NodeJS and SQL and we are both experienced in Android and iOS development, we will set this to **High**

Platform Experience (PLEX)

Again the team has already experience with the platform, set to **High**

Language and Tool Experience (LTEX)

Given that the tools and languages used were chosen by us, we chose the ones we have experience on. Set to **High**

Use of Software Tools (TOOL)

Since we have not a great experience with tools, but we will perform (for the first time) integration and a more formal development. Set to **Low**

Multisite Development (SITE)

Since we live in the same city, Milan, we are setting this to **Very High**

Required Development Schedule (SCED)

Given that we are not given any strong constraint on the development, we are setting this to **Nominal**

2.2.3 Drivers recap

Here a recap, in table form, of the choosen parameters is given:

Driver	**Value**
PREC	Very High
FLEX	Nominal
RESL	High
TEAM	Very High
PMAT	High
RELY	Nominal
DATA	Low
CPLX	Nominal
RUSE	Low
DOCU	Nominal
TIME	High
STOR	Nominal
PVOL	Nominal
ACAP	Low
PCAP	High
PCON	Very High
APEX	High
PLEX	High
LTEX	High
TOOL	Low
SITE	Very High
SCED	Nominal

2.2.4 Results

To compute the COCOMO estimation we used an online tool (referenced in Paragraph 1.3). A view of the setup is presented here, with the 2 different results provided (using both the average SLOC and maximum SLOC)

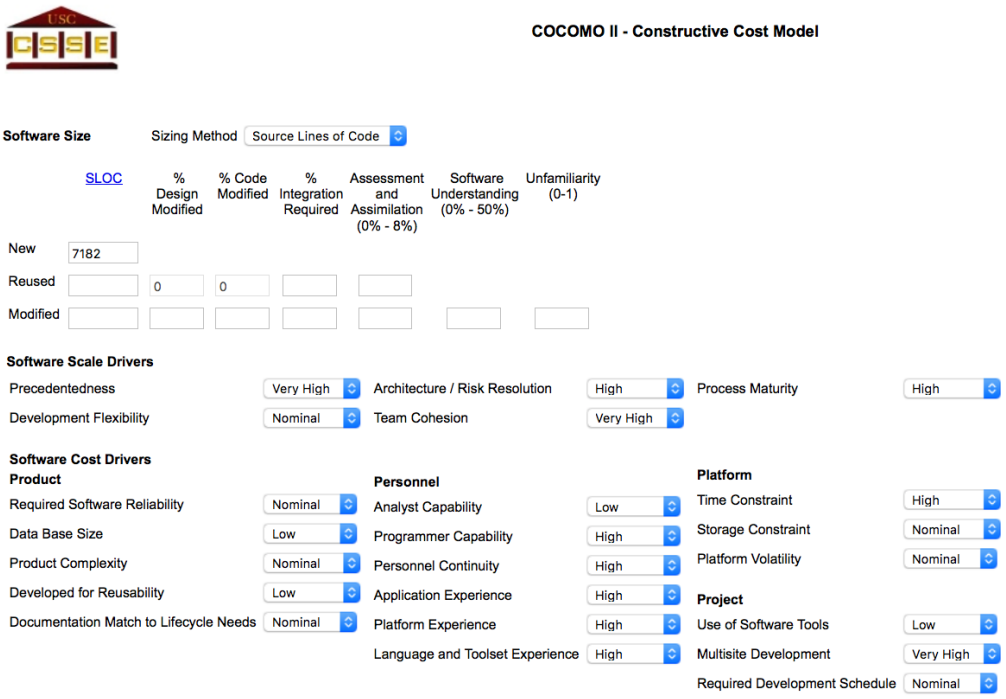


Figure 2.9: Screenshot of the online tool.

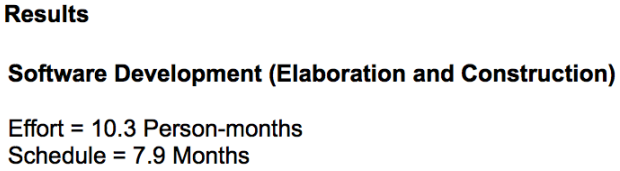


Figure 2.10: Results for the average SLOC.

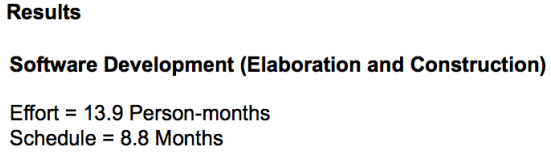


Figure 2.11: Results for the maximum SLOC.

Planned Schedule

We propose here a likely schedule for the project, shown in 2 GANTT charts. The chart are divided in a temporal basis to enhance readability. The first chart includes the Requirements Analysis and Design phases, whereas the second one includes the Development and Deployment phases. The total schedule duration is almost 7 months, that is very near the project effort estimation from the previous chapter.

The colors in the chart have the following meaning:

- **GREEN:** Macro-phase duration
- **BLUE:** Meeting with PowerEni
- **ORANGE:** Single task
- **GRAY:** System is online

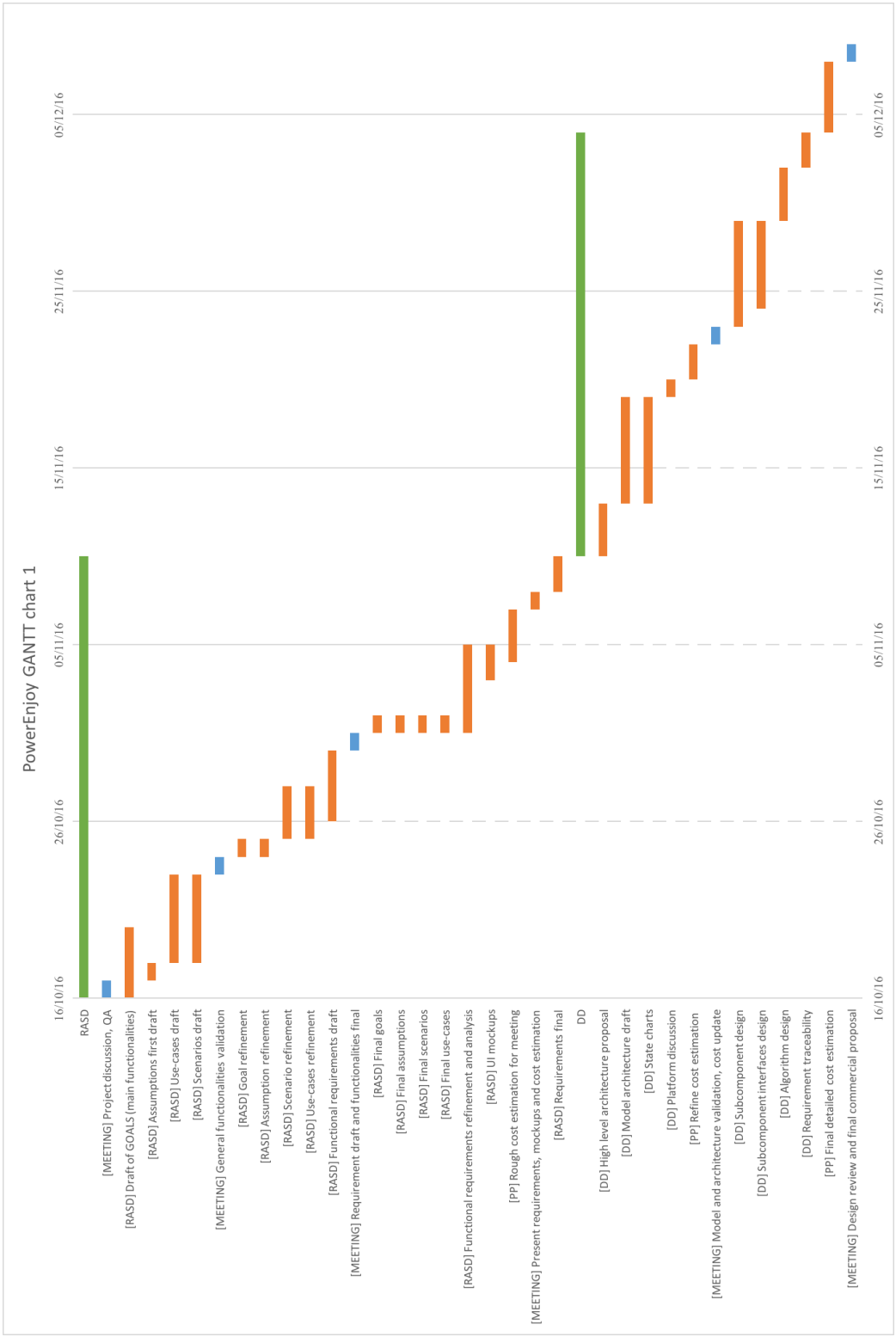


Figure 3.1: GANTT chart of first period from 16/10/2016 to 09/12/2016.

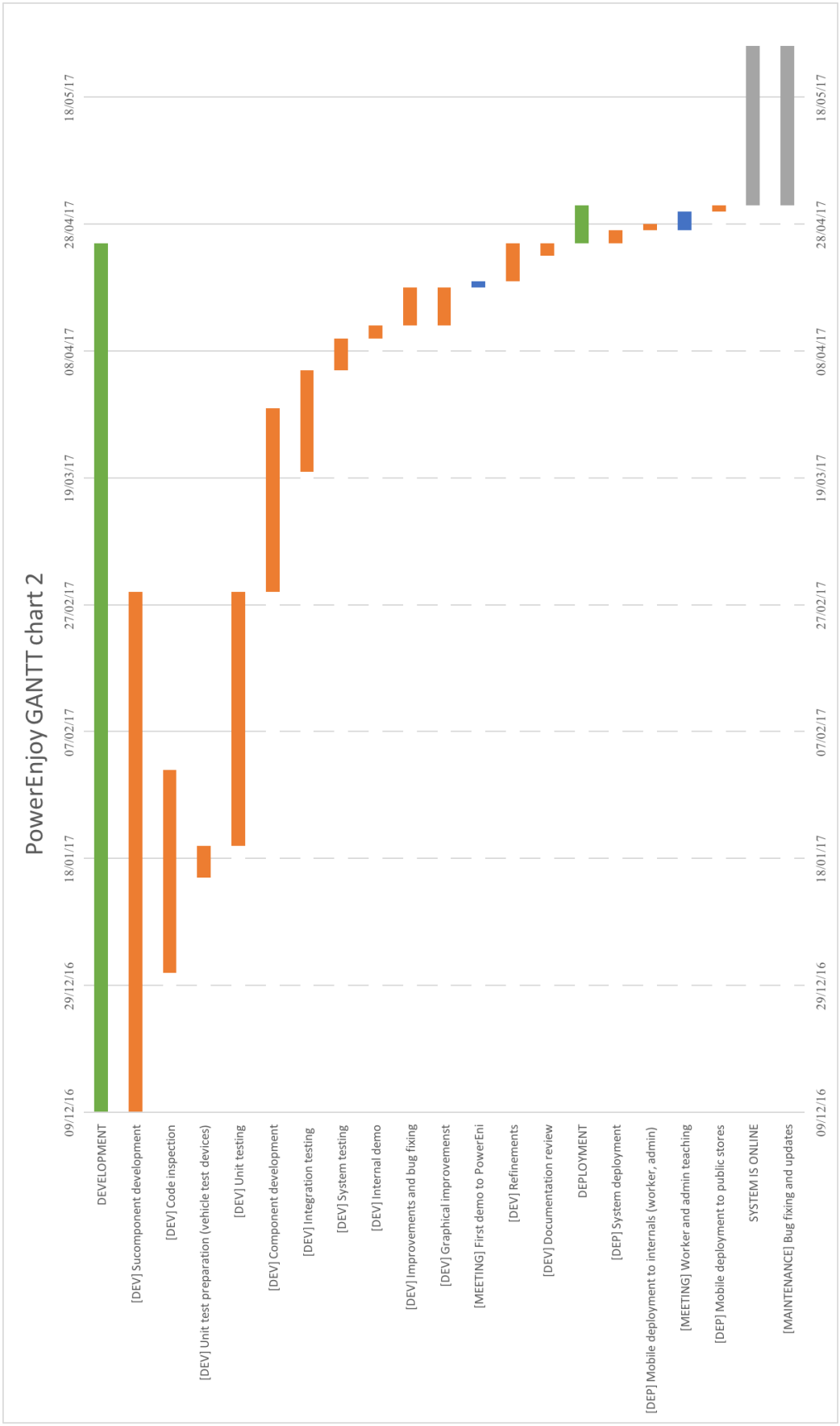


Figure 3.2: GANTT chart of second period from 09/12/2016 to 01/05/2017.

Resource Allocation

In this chapter we will instead present similar GANTT charts, but instead of displaying the time-schedule only, we will combine it with a resource allocation schedule, i.e. what member of the team is executing the task. The tasks in this schedule and the division in 2 periods are the same of Chapter 3.

The colors in the chart have the following meaning:

- **GRAY:** Macro-phase duration
- **BLUE:** Meeting with PowerEni
- **GREEN:** Nico Montali (aka nicomon) is executing the task
- **RED:** Enrico Fini (aka donkeyshot21) is executing the task
- **ORANGE:** Task is executed by both team members, possibly dividing it into sub-tasks.

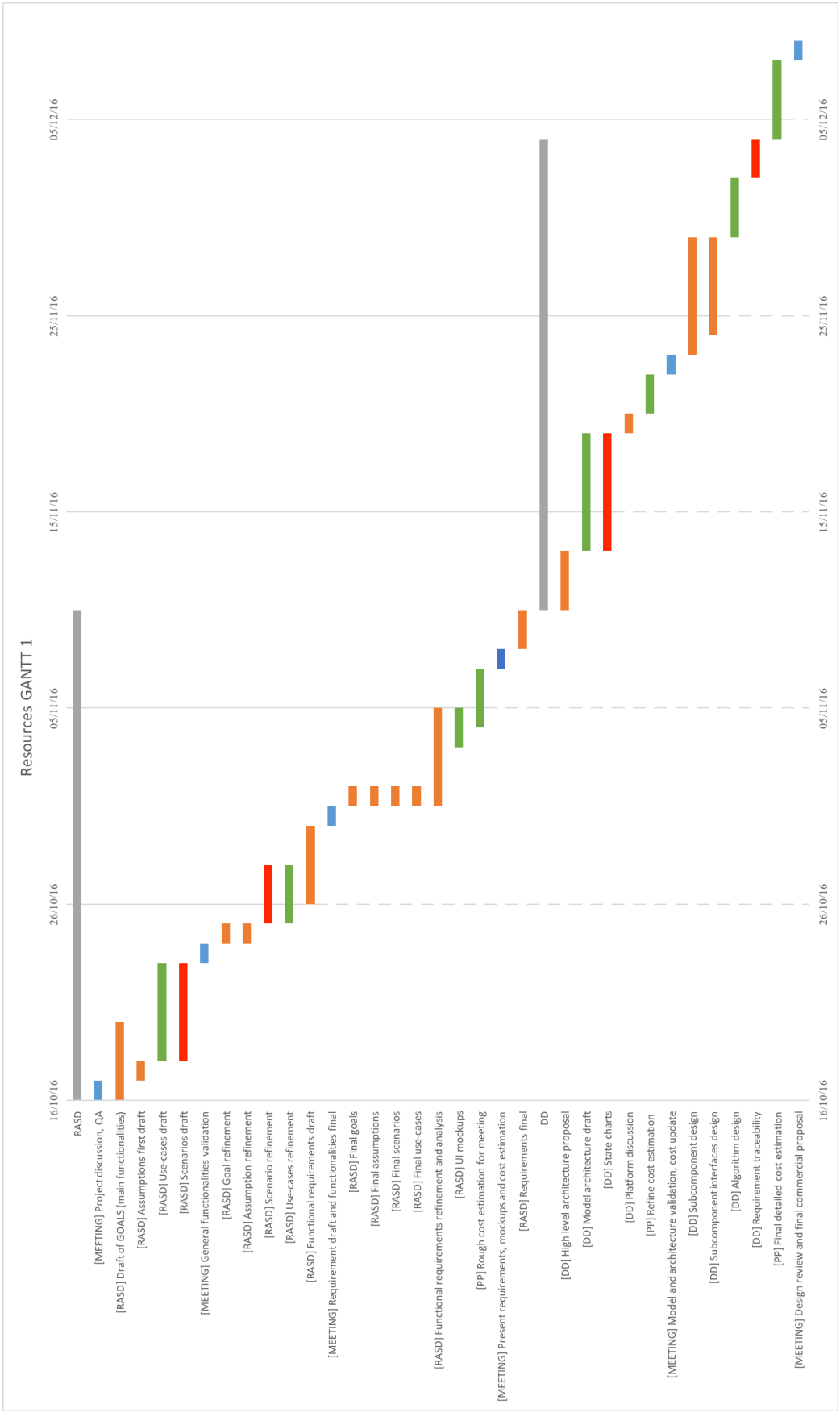
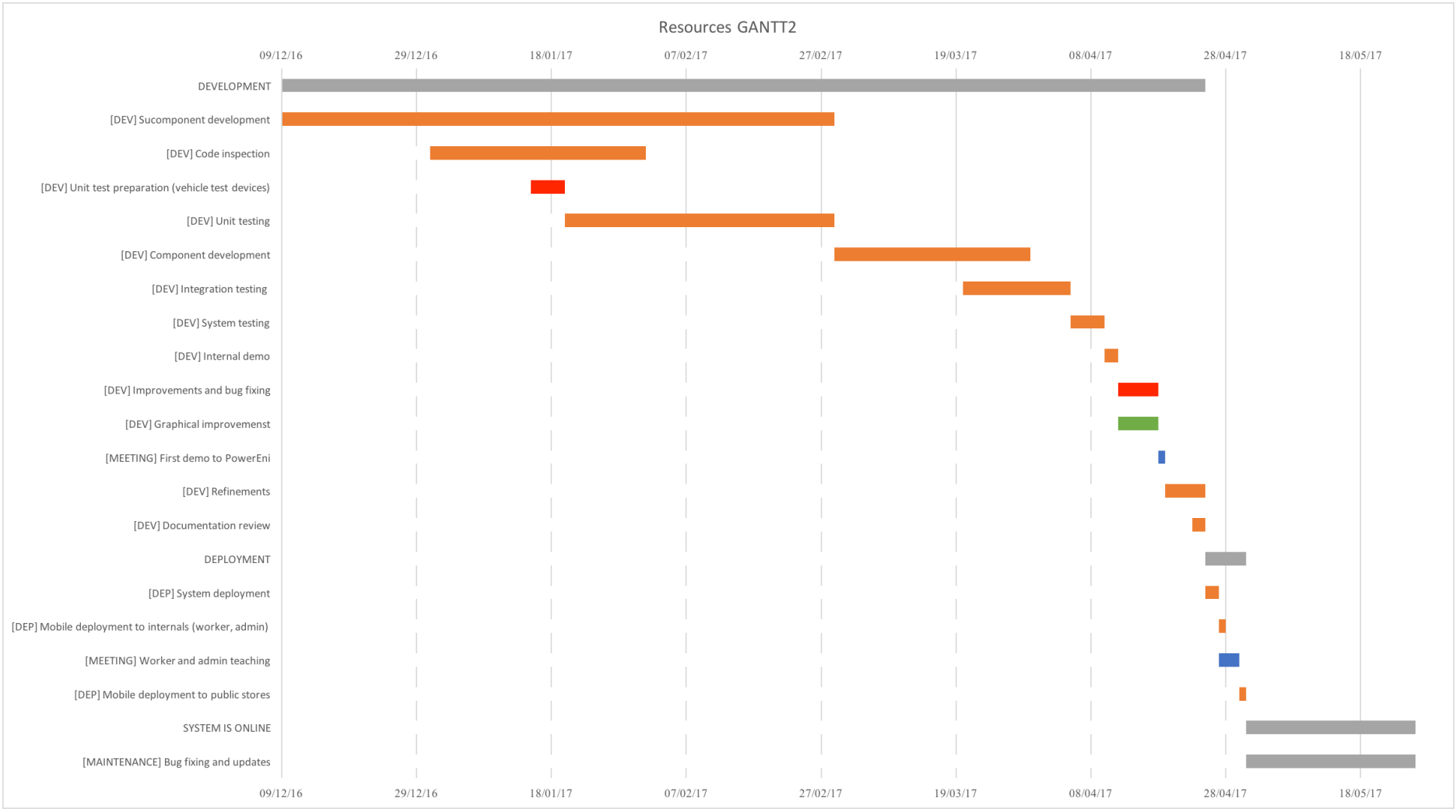


Figure 4.1: GANTT chart of first period from 16/10/2016 to 09/12/2016.


Resources GANTT2
09/12/16
29/12/16
18/01/17
07/02/17
27/02/17
19/03/17
08/04/17
28/04/17
18/05/17
DEVELOPMENT
[DEV] Sucomponent development
[DEV] Code inspection
[DEV] Unit test preparation (vehicle test devices)
[DEV] Unit testing
[DEV] Component development
[DEV] Integration testing
[DEV] System testing
[DEV] Internal demo
[DEV] Improvements and bug fixing
[DEV] Graphical improvemenst
[MEETING] First demo to PowerEni
[DEV] Refinements
[DEV] Documentation review
DEPLOYMENT
[DEP] System deployment
[DEP] Mobile deployment to internals (worker, admin)
[MEETING] Worker and admin teaching
[DEP] Mobile deployment to public stores
SYSTEM IS ONLINE
[MAINTENANCE] Bug fixing and updates

Risk Management

After a risk analysis, we discovered several possible issues that may occur during the development of the project. They are divided in Project Risks, Technical Risks and Legal and Financial Risks.

5.0.5 Project Risks

- Requirement Change: during the development of the project it could happen that our stakeholders change some requirement. This eventuality is completely unpredictable and the risk level is really high both for probability (it is not uncommon) and because it would require a partial restructuration of the System. This risk can be mitigated using a modular design and writing reusable code or letting the stakeholders be an active part of the project (i.e. using Agile methodology with frequent deliveries);

- Lack of Experience: since it is the first time we commit in the development of a large scale system, some problems caused by inexperience could arise. For example, during the implementation phase, it could happen that one of our developers is not able to solve a specific issue, causing delays. This problem can be partially mitigated hiring flexible developers. Also team working could be fundamental in this scenario.

5.0.6 Technical Risks

- Security Problems: when the service becomes publicly accessible there is a chance that malicious attackers could try to steal data from our servers. Since we deal with personal information of Drivers and credit cards it is of prime importance that we put all our effort in vulnerability detection and fixing. Also architectural choices can play an important role in this scenario.

- Performance Problems: it is not easy to foresee the magnitude of workload of the System. If the application becomes very popular it could be that the computational power or the storing space available on our servers are not sufficient to handle the traffic. To overcome this issue we should use a scalable cloud computing service which is able to increase its power when needed.

- Dependence on Third Party Software: the development of such a complex system implies a massive use of external services. Some of them could be currently under development. For example it could happen that Google releases a new version of Android OS with new APIs. This could cause delays in the project schedule because our components should be readapted. The bad news is that there are no countermeasures for issues of this type and it is also hard to foresee them.

5.0.7 Legal and Business Risks

- Legal Problems: since our cars are supposed to be driven trough public streets the chance of accidents or malfunctions is really high. In this case the System and the cars should be highly reliable because if something goes wrong the responsibility is ours. To overcome this issues we should plan an accurate maintenance schedule and more importantly contact a legal firm.

- Competition: nowadays there are tons of car sharing services in every city. To persuade people to use ours service in place of another we should advertise it in a proper way, focusing on its main assets: electric cars, low cost, simple interface.

Work review

Based on our log of the work phases, the total amount of hour of work required were:

- N. Montali: 16 hours
  - FP, COCOMO
  - Schedule
  - Resource allocation
- E. Fini: 7 hours
  - Introduction
  - Risk Management