Politecnico di Milano
MSc Computer Science and Engineering

Software Engineering 2
Academic Year 2016-2017

Project Plan Document

PowerEnJoy

Authors:

Melloni Giulio 876279
Renzi Marco 878269
Testa Filippo 875456

Reference Professor:
Mottola Luca

Release Date: January 22th, 2017
Version 1.0

Table of Contents

1 Introduction **1**
- 1.1 Revision History . . . 1
- 1.2 Purpose and Scope . . . 1
- 1.3 Definitions, Acronyms, Abbreviations . . . 1
- 1.4 Reference Documents . . . 2

2 Project size, cost and effort estimation **3**
- 2.1 Size Estimation: function points . . . 3
  - 2.1.1 Internal Logic Files: ILFs . . . 4
  - 2.1.2 External Logic Files: ELFs . . . 5
  - 2.1.3 External Inputs: EIs . . . 5
  - 2.1.4 External Inquiries: EQs . . . 6
  - 2.1.5 External Outputs: EOs . . . 6
  - 2.1.6 Overall Estimation . . . 7
- 2.2 Cost and effort estimation: COCOMO II . . . 8
  - 2.2.1 Scale Factors . . . 8
  - 2.2.2 Cost Drivers . . . 10

3 Schedule **11**

4 Resource Allocation **12**

5 Risk Management and Mitigation **13**
- 5.1 Project risks . . . 13
- 5.2 Technical risks . . . 14
- 5.3 Business risks . . . 14
- 5.4 Personnel risks . . . 15

6 Effort Spent **16**

List of Figures

1 Introduction

1.1 Revision History

Version	Date	Summary
1.0	22/01/2017	First release of the document

1.2 Purpose and Scope

1.3 Definitions, Acronyms, Abbreviations

PowerEnJoy is the name of the system that has to be developed.

System sometimes called also *system-to-be*, represents the application that will be described and implemented. In particular, its structure and implementation will be explained in the following documents. People that will use the car-sharing service will interact with it, via some interfaces, in order to complete some operations (e.g.: reservation and renting).

Renting it is the act of picking-up an available car and of starting to drive.

Ride the event of picking-up a car, driving through the city and parking it. Every Ride is associated to a single user and to a single car.

Reservation it is the action of booking an available car.

Car a car is an electrical vehicle that will be used by a registered user.

Not Registered User indicates a person who hasn't registered to the system yet; for this reason he can't access to any of the offered function. The only possible action that he can carry out is the registration to get a personal account.

Registered User interacts with the system to use the sharing service. He has an account (which contains personal information, driving license number and payment data) that must be used to access to the application in order to exploit all the functionalities.

Employee it's a person who works for the company, whose main task is to plug into the power grid those cars that haven't been plugged in by the users. He is also in charge of taking care of the status of the cars and of moving the vehicles from a safe area to a charging area and vice versa if needed.

Safe Area indicates a set of parking lots where the users have to leave the car at the end of the rent; the set of the Safe Areas is pre-defined by the system management. These areas are spread all over the city.

Plug defines the electrical component that physically connects the car to the power grid.

Charging Area is a special *Safe Area* that also provides a certain number of plugs that connect the cars to the power grid in order to recharge the battery.

Registration the procedure that an unregistered user has to perform to become a registered user. At the end, the unregistered user will have an account. To complete this operation three different types of data are required: personal information, driving license number and payment info.

Search this functionality lets the registered user search for available cars within a certain range from his/her current position or from a specified address.

RASD is the acronym of *Requirements Analysis and Specification Document*

DD is the acronym of *Design Document*

ITPD is the acronym of *Integration Test Plan Document*

PPD is the acronym of *Project Plan Document*

1.4 Reference Documents

- Project Assignments 2016-2017
- RASD v1.1
- DD v1.1
- ITPD v1.0

2 | Project size, cost and effort estimation

This section talks about the estimation of the size of the project, giving an estimation of lines of code that must be written. This evaluation comes from the calculation of the lines of code that occur to write both the Business Logic part and the User Interface part.
Function Point is the approach used to simulate how the project will be expensive in terms of lines of code.
From this estimation it will be easy to find how much time will be spent on developing the Business Logic part of the project.

2.1 Size Estimation: function points

Functionalities that are to be developed and their complexity are provided by the Function Point approach.
In each table is reported how much load can take developing these parts: from *Low*, not so much expensive, to *High*, the heavier.

Internal Logic Files & External Logic Files

		Data Element		
Record Elements		1-19	20-49	50+
	1	Low	Low	Adw
	2-5	Low	Adw	High
	6+	Adw	High	High

Each line of this table underlines how it becomes more heavy the computation with the increasing of *Data Elements* and *Record Elements*.
Adding few records or some data, the program is rapidly loading, stretching the response time.

External Outputs & External Inputs

		Data Element		
Record Elements		1-19	20-49	50+
	1	Low	Low	Adw
	2-5	Low	Adw	High
	6+	Adw	High	High

With *External Outputs* and *External Inputs*, the size of Data and Records must be smaller than the previous ones seen in *Internal Logic Files* and *External Logic Files*.
We have a big load adding only one record, or a few data, getting worst the performance of the system.

External Inquiries

		Data Element		
Record Elements		1-19	20-49	50+
	1	Low	Low	Adw
	2-5	Low	Adw	High
	6+	Adw	High	High

The parts of External Input is the heavier, because it is an interaction between the User and the System.
The table is so heavier at littler variations. One record, or few data are enough to increase the performance of the system.

Complexity Weights

		Complexity Weight		
Function Type	Acronym	*Low*	*Mid*	*High*
Internal Logic Files	ILFs	7	9	15
External Logic Files	ELFs	5	7	10
External Inputs	EIs	2	4	6
External Inquiries	EQs	6	8	10
External Outputs	EOs	2	3	4

The general complexity is so defined.
It is obviously heavier a Function Type like the *Internal Logic Files* or an *External Inquiries*, which manage the update of the Data and the communication between the User and the System, than a *External Output*, *External Input* or an *External Logic Files*, that gives only an output or take an input.

2.1.1 Internal Logic Files: ILFs

Some functionalities store information into the PowerEnjoy system, in order to remember data and let the system use and manage them.
The login is one of these functionalities: the system has to store the *User Data* in order to recognize him. When the data is stored, it will be composed by many fields explaining the personal information, in order to characterize every person.
Safe Area, *Parking Area* and *Car Data* are inserted by the system at each time a new station or vehicle is added into the system. *Safe Area* and *Parking Area* will be condensed in the same table, but diversified by some attributes, in order to have an increase of performance, searching for the data in only one place.

Differently, *Reservation* is continuously added, through a new operation on the app.

ILFs	*Complexity*	*Points*
User Data	Mid	9
Employee Data	Low	7
Safe Area Data	Low	7
Parking Area Data	Low	7
Reservation Data	Mid	9
Ride Data	Mid	9
Car Data	Low	7
Total		55

2.1.2 External Logic Files: ELFs

The application manages the interaction and the system acquires information from the external services used. *Payment Data* and *Map Data* are considered External Logic Files, because they are provided by external component.
Because of these components are not part of our system, PowerEnjoy will obtain information thanks to external APIs in order to link internal to external data.
Payment Data will be more complex to develop, because the system has to guarantee the security of the payment and of the information, avoiding every type of external attack.

ELFs	*Complexity*	*Points*
Maps Data	Mid	7
Payment Data	High	10
Total		17

2.1.3 External Inputs: EIs

These sections underline the methods called by an operation of the User on our application. The action is so called a client-server action, because the system reacts to an User operation. There are two different type of these changes:

- **UI Action**: the user click some buttons, or insert some text on the User Interface of the application, or on the car screen.
- **Physical Action**: the user plugs the car in a Parking Area.

The most complex will be the creation of the reservation, because it has to interact with data, creating them and inserting them linking the User Interface to the remote DBMS.
It is easy to see how is more easy to develop the part that check if the car has been plugged or not, after the stop of the ride.

EIs	*Complexity*	*Points*
Login	Mid	4
Logout	Low	2
Registration	Mid	4
Insert Reservation	High	6
Delete Reservation	Mid	4
Car Plug In	High	6
Update Car Status	Low	2
Insert a new car	Mid	4
Update a car	Mid	4
Delete a car	Mid	4
Insert a new charging area	High	6
Update charging area	Mid	4
Delete charging area	Mid	4
Convey end ride via touch screen	Mid	4
Insert final destination	Mid	4
Total		62

2.1.4 External Inquiries: EQs

These operation that involves an input and an output are the *Ride* actions: when it starts and when it stops.
Starting the ride, the user interact with the Car UI so that the Car System can communicate to the Server that the Ride is started, and the Server can change User Interface on board displaying his navigation.
Otherwise, the system will calculate the cost, when the user interacts with the UI and it will display temporary how much he has to pay.
So *Ride Start* and *Ride Stop* are easily considered as high complex to develop, because they have different communication between the car and the system.
Differently, the *Search* for a car or the *Pick Up* of it, are actions which involve only one communication car - system, so they are considered easier to develop.

EQs	*Complexity*	*Points*
Search for available cars	Mid	8
Show cars information	Mid	8
Enable MoneySavingOption	High	10
Pick Up a car	High	10
Ride initialization	HIgh	8
Total		44

2.1.5 External Outputs: EOs

External Outputs are elementary operations that generates data that are presented to the User, without any client action.
The cost notification is one example from EOs section: the user, stopping the ride, gets out of the car, and after a certain time the system checks if he has plugged or not his vehicle. After this, the system displays how much the user has to pay.

This is an example of an operation send by the system without any actions of the user.

EOs	*Complexity*	*Points*
Payment Notification	Low	2
Notify Expired Reservation	Mid	4
Total		6

2.1.6 Overall Estimation

Considering the whole estimation of the Function Types, the total will be the summation of each one of the Total previously found.

Function Type	*Acronym*	*Value*
Internal Logic Files	ILFs	55
External Logic Files	ELFs	17
External Inputs	EIs	62
External Inquiries	EQs	44
External Outputs	EOs	6
Total		184

Having this result, the total lines of code are easily calculable using *conversion factor* and the *usage percentage* used to specify how much the project is divided in different languages. Using this method we can consider the entire project, from the Business Logic to the UI development part, adopting a weighted average between languages:

Language	**Conversion factor (SLOC/FP)**	**Usage percentage**
HTML	34	15%
JEE	46	75%
Javascript	43	10%

According to these constants and the usage percentage of the language, he total lines of code to write are:

$$\textit{Code Lines} = H_t \cdot (C_{\text{HTML}} \cdot \text{P}_{\text{HTML}} + C_{\text{JEE}} \cdot \text{P}_{\text{JEE}} + C_{\text{JS}} \cdot \text{P}_{\text{JS}}) \tag{2.1}$$

Where:

- C_i express the *conversion factor* of the i language
- P_i express the *usage percentage* of the i of the language in the whole project
- H_t is the *FPs result*

The result is so found:

$$\textit{Code Lines} = 184 \cdot (34_{\text{HTML}} \cdot 0.15_{\text{HTML}} + 46_{\text{JEE}} \cdot 0.75_{\text{JEE}} + 43_{\text{JS}} \cdot 0.1_{\text{JS}}) = \textit{8078} \tag{2.2}$$

2.2 Cost and effort estimation: COCOMO II

This section deals with the estimation of the cost and the effort that are required to develop the system. The evaluation will be carried out using the *COnstructive COst MOdel II* (later on referred as *COCOMO II*) and will make use of the estimated size of the project obtained in the previous section. Let us first introduce the empirical formula that will guide the effort estimation process:

$$Effort = A \cdot Size^{E} \cdot \prod_{i=1}^{17} EM_i \tag{2.3}$$

where:

- Effort is expressed in PM (Person-Months)
- $A = 2.94$ is a productivity constant in PM/KSLOC (Person-Months/Kilo-Source Lines of Code)
- Size is the estimated size of the project in KSLOC obtained in the previous analysis
- E is an aggregated index of five *Scale Factors*
- EM_i stands for *Effort Multiplier* and corresponds to the value of a specific entry that will be explained later in the *Cost Drivers* subsection

Next subsections are focused on the evaluation of the two last operands of the previous formula. At the end of the section the final result obtained applying formula 2.3 is shown.

2.2.1 Scale Factors

The first step consists in evaluating the actual value of exponent E in formula 2.3. To do so, a list of *Scale Factors* have to be computed. These entries represent an heterogeneous set of variables and their values are assigned according to the following table:

Scale Factor	Very low	Low	Nominal	High	Very high	Extra high
PREC	Thoroughly unprecedented	Largely unprecedented	Somewhat unprecedented	Generally familiar	Largely familiar	Thoroughly familiar
SF_i	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_i	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_i	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_i	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	Level 1 Lower	Level 1 Upper	Level 2	Level 3	Level 4	Level 5
SF_i	7.80	6.24	4.68	3.12	1.56	0.00

Let us discuss each of these entries:

- **Precedentedness**: It is the index that defines the previous experience of the team in developing such projects. This is the very first project, thus a *Very low* level is reasonable.
- **Development flexibility**: The index of the flexibility for the project choices with respect to the external interfaces and the requirements. The project has strict requirements to abide by in terms of functionalities of the system, but no specific architecture is required. This variable is then assigned a level of *Nominal*.
- **Architecture and risk resolution**: It reflects the capability of managing possible risks with an accurate contingency plan and the fact that a reasonable schedule is sketched. In the document both these aspects are covered in detail, so for this variable the right choice is to pick the *Very High* level.
- **Team cohesion**: It deals with the capability of the members of the team to work together in harmony and sharing the same vision. The team is made of only three people in this project and they work with great commitment and together, thus *Very high* is the right degree.

- **Process Maturity**: it is a index of the maturity of the organization. Even though this is the very first project of the team, the schedule has been accurately defined, the project widely documented and discussed among the members of the team. Consequently *Nominal* level is reasonable.

Here is the re-cap of the *Scale Factors* analysis:

Scale Factor	*Level*	*Value*
Precedentness (PREC)	Very low	6.20
Development flexibility (FLEX)	Nominal	3.04
Architecture and risk resolution (RESL)	Very high	1.41
Team cohesion (TEAM)	Very high	1.10
Process Maturity (PMAT)	Nominal	4.68
Total		16.43

It is now possible to evaluate exponent E with the following formula:

$$E = 0.91 + 0.01 \cdot \sum_{i=1}^{5} SF_i$$

The result is

$$E = 1.0743$$

2.2.2 Cost Drivers

3 | Schedule

A fundamental part of the project plan is also the scheduling. This step allows to divide the overall work that has to be done in separated tasks and moreover to define when and how they will be completed; in particular the estimation of the calendar time required to execute each task is provided through the Gantt diagram.
It should be noted that the time estimation of the tasks is not trivial and while doing this it's important to consider that the productivity is not proportional to the number of people of the team and that some extra time must be added because sometimes the flow of the events does not follow the expected one.
The analysis and the definition of the various tasks is carried out at a quite coarse grained level thus only the main activities are included into the Gantt diagram, however they shouldn't be too small (for example a reasonable length is between 1 and 2 weeks).
Lastly, the start of the schedule of the project is set to the month of October.

4 | Resource Allocation

The Resource Allocation phase is strongly correlated with the Schedule phase because its main goal is to properly assign every single task, defined in the previous step, to the components of the development team. The assignment process should be done carefully and in such a way to optimize the effort of each team member; for instance, the allocation of all the activities and tasks is done with respect to the three authors of these documents assuming that will be also the development team.
As for the Schedule, the Resource Allocation is presented via a Gantt diagram too.

5 | Risk Management and Mitigation

This section deals with the possible risks that can threaten the system during its development process. These hazards can stem from a variety of sources and thus can be classified as:

- **Project risks**: if they threaten the project plan, such as the project schedule. As can be seen in the following tables, the most common problem that can arise from such risks is a delay in the project release.
- **Technical risks**: they are related to the technical part of the project. They include a major variety of problems such as unskilled staff, flaws in the external adopted components, changes in the set of the requirements and so on.
- **Business risks**: they include a set of heterogeneous hazards such as budget cuts, sales falls and market policy flaws. If they become concrete, they can compromise the viability of the project.
- **Personnel risks**: this type of hazards deals with all the possible problems that can be met within the team of work.

Since a pro-active risk-management approach is desirable, for each category of risk a list of the most recurrent hazards will be taken into account, along with their probabilities to be faced, their levels of impact on the project and the strategies that can be adopted to mitigate them.

5.1 Project risks

Risk	Probability	Impact	Strategy
Underestimated development time	Moderate	Serious	If the spare time is not sufficient, try to negotiate with the customer the possibility to have two releases, the first to the planned one and the second one as early as possible.
A change in the direction of the project	Low	Moderate	Redact very precise and detailed documentation of the project so that new managers are able to handle the process.

5.2 Technical risks

Risk	Probability	Impact	Strategy
Difficulty in recruiting a skilled staff	Moderate	Serious	Be prepared and inclined to pay extra to find skilled people. If it is not sufficient, may consider the possibility of buying external components that are already developed.
Changes in the requirements	Low	Serious	Be compliant with the requirements traced in the *RASD* and in case changes occur evaluate the trade off related to the needed variations in the project. Increment on the price should come from the customer.
External components have flaws	Moderate	Serious	Choose components that are on the market for a long time, so that they can be reliable. Experience with precedent projects may be an advantage in this case.
Modification in the APIs of the external components	Low	Moderate	Write the code as portable as possible. Changes in the APIs should not break the system. Also plan updates in case such modifications occur.

5.3 Business risks

Risk	Probability	Impact	Strategy
Budget cuts during the development	Moderate	Tremendous	This kind of risk is the most critical one. In order to mitigate it, redact a document in which the financial benefits associated to the business of the project is clearly stated and try to convince managers not to apply cuts.
Poor sales	Moderate	Serious	Make special discounts and offers at the launch of the product in order to achieve great popularity. In doing so, a market analysis has to be carried out .
Competitors in the market	Moderate	Moderate	Choose a guide line for your product: should it be cheaper than the other or should it aim to optimality? In the launch phase also consider the possibility of making special discounts to attract clients.

5.4 Personnel risks

Risk	Probability	Impact	Strategy
Poor motivation	Moderate	Moderate	Try to select people that are skilled in particular fields and have experience in their work. Praise their commitment with rewards.
Conflicts within the work team	Low	Moderate	Promote the cooperation of people by making a community that periodically meets and discuss on the level of the project.
Key staff are ill at critical times in the project	Moderate	Serious	Consider the possibility to have several people working on common tasks in order to prevent excessive time losses.

6 | Effort Spent

In order to complete this document, each author worked for XX hours.