Power EnJoy
Project Plan Document

Version 1.0.0

Redaelli Marco 877622   Zanoli Francesco 877471

22/01/2017

Contents

Figure Contents

Chapter 1

Introduction

his section contains a brief introduction to the **P**roject **P**lan **D**ocument.

1.1 Purpose and Scope

The purpose of this document is to explain the Project Plan devised for the system to be. All the people involved in the project could be considered as possible readers of the document, but the document itself is more of a guide for the Project Manager and the Management in general. The Project Plan consists in tables, Gantt diagrams, charts and natural language descriptions of the planning, scheduling and management of *PowerEnJoy* development.

1.2 List of Definitions and Abbreviations

In the document are often used some technical terms whose definitions are here reported:

- See the correspondent section in the **RASD** for more definitions.

For sake of brevity, some acronyms and abbreviations are used:

- **PPD:** Project Plan Document.
- **FP:** Function Points.
- **ILF:** Internal Logic File.
- **ELF:** External Logic File.
- **EI:** External Input.
- **EO:** External Output.
- **EIQ:** External Inquiry.

- **COCOMO:** Constructive Cost Model.

- See the correspondent section in the **RASD** for more acronyms and abbreviations.

1.3   List of Reference Documents

- ***PowerEnJoy* RASD v1.3:** Requirements Analysis and Specification Document

- **Assignment 5 - Project Plan:** Project Plan specification document

- **COCOMO II - Model Definition Manual:** Document link

Chapter 2

Function Points

The Functional Point approach is a technique that allows to evaluate the effort needed for the design and implementation of a project. We have used this technique to evaluate the application dimension basing on the functionalities of the application itself. The functionalities list has been obtained from the RASD document and for each one of them the realization complexity has been evaluated. The functionalities has been groped in:

- Internal Logic File: it represents a set of homogeneous data handled by the system

- External Interface File: it represents a set of homogeneous data used by the application but handled by external application

- External Input: elementary operation that allows input of data in the system

- External Output: elementary operation that creates a bit stream towards the outside of the application

- External Inquiry: elementary operation that involves input and output operations

The following table outline the number of Functional Point based on functionality and relative complexity:

Function Type	Complexity		
	Simple	Medium	Complex
Internal Logic File	7	10	15
External Interface File	5	7	10
External Input	3	4	6
External Output	4	5	7
External Inquiry	3	4	6

2.1 Internal Logic File

Internal Logic Files are homogeneous sets of data used and managed by the application.
Summary of the calculated weights:

$$\begin{aligned} FP_{ILF} \quad &= 5 \cdot w_{Simple,ILF} + 3 \cdot w_{Medium,ILF} + 1 \cdot w_{Complex,ILF} \\ &= 5 \cdot 7 + 3 \cdot 10 + 1 \cdot 15 \\ &= 80 \end{aligned}$$

In details:

- **Ride:**

  Ride data. Frequent modifications, frequent insertions, frequent readings.

  $$1 \cdot w_{Medium,ILF}$$

- **User:**

  Personal Registered Passenger data. Infrequent modifications, but frequent insertions and readings.

  $$1 \cdot w_{Medium,ILF}$$

- **Car:**

  Car information. Infrequent insertion and modification, frequent reading

  $$1 \cdot w_{Medium,ILF}$$

- **Customer Service:**

  Customer help request. Very infrequent modifications, insertions and readings.

  $$1 \cdot w_{Simple,ILF}$$

- **Payment:**

  Data of the ended rent. Infrequent modification and reading, very frequent insertion

  $$1 \cdot w_{Complex,ILF}$$

- **Payment Method:**

  User's payment data, Frequent readings, infrequent modification and insertion

  $$1 \cdot w_{Simple,ILF}$$

- **Parking:**

  Parking data, frequent reading, infrequent insertion and modifications

  $$1 \cdot w_{Simple,ILF}$$

- **Service Station:**

  Service station data. frequent reading, infrequent insertion and modifications

  $$1 \cdot w_{Simple,ILF}$$

- **GPS Data:**

  GPS Data, Database Server side. Very frequent insertions and readings, but very infrequent modifications.

  $$1 \cdot w_{Simple,ILF}$$

2.2 External Interface File

External Interface Files are homogeneous sets of data used by the application but generated and maintained by other applications.
Summary of the calculated weights:

$$
\begin{aligned}
FP_{ELF} \quad &= 2 \cdot w_{Simple,ELF} \\
&= 2 \cdot 5 \\
&= 10
\end{aligned}
$$

In details:

- **Google Maps API:**

  Google Maps API related data about Travel Time and Addresses. Very frequent readings.

  $$1 \cdot w_{Simple,ELF}$$

- **Paypal API:**

  Paypal API request related to user's payement. Frequent insertion

  $$1 \cdot w_{Simple,ELF}$$

2.3   External Input

External Inputs are elementary operations to elaborate data coming from the external environment.
Summary of the calculated weights:

$$
\begin{aligned}
FP_{EI} \quad &= 5 \cdot w_{Simple,EI} + 2 \cdot w_{Medium,EI} + 2 \cdot w_{Complex,EI} \\
&= 5 \cdot 3 + 2 \cdot 4 + 2 \cdot 6 \\
&= 35
\end{aligned}
$$

In details:

- **Login:**

  This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

  $$1 \cdot w_{Simple,EI}$$

- **Logout:**

  This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

  $$1 \cdot w_{Simple,EI}$$

- **Registration:**

  This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

  $$1 \cdot w_{Simple,EI}$$

- **Handle Personal Profile:**

  This operation requires a simple effort. In fact it has to perform few steps in order to conclude the procedure.

$$1 \cdot w_{Simple,EI}$$

- **Car Interaction:**

  This operation requires a medium effort. In fact it is very frequent.

$$1 \cdot w_{Medium,EI}$$

- **Car Reservation:**

  This operation requires a complex effort. In fact it has to perform many elementary steps in order to handle it

$$1 \cdot w_{Complex,EI}$$

- **Problem:**

  This operation requires a medium effort

$$1 \cdot w_{Medium,EI}$$

- **Rent:**

  This operation requires a simple effort.

$$1 \cdot w_{Simple,EI}$$

- **End of Rent:**

  This operation requires a complex effort. In fact it has to perform many elementary steps in order to handle it

$$1 \cdot w_{Complex,EI}$$

2.4   External Output

External Outputs are elementary operations that generate data for the external environment, and they usually include the elaboration of data from logic files. Summary of the calculated weights:

$$
\begin{aligned}
FP_{EO} \quad &= 4 \cdot w_{Simple,EO} \\
&= 4 \cdot 4 \\
&= 16
\end{aligned}
$$

In details:

- **User Notification:**

  The result of the end of the rent must be send to the user who handle it

$$1 \cdot w_{Simple,EO}$$

- **Email Confirmation:**

  The result of this operation must be sent to the user that registers

$$1 \cdot w_{Simple,EO}$$

- **Maintenance Notification:**

  The result of the maintenance call must be sent to the specific user.

$$1 \cdot w_{Simple,EO}$$

- **Reservation:**

  Informations about the reservation must be sent to the related user

$$1 \cdot w_{Simple,EO}$$

2.5   External Inquiry

External Inquiries are elementary operations that involve input and output, without significant elaboration of data from logic files.
Summary of the calculated weights:

$$
\begin{aligned}
FP_{EIQ} \quad &= 2 \cdot w_{Medium,EIQ} \\
&= 2 \cdot 4 \\
&= 8
\end{aligned}
$$

In details:

- **Manager car status:**

  In order to perform these operations the system has to retrieve, send and render simple data.

$$1 \cdot w_{Medium,EIQ}$$

- **Manager user status:**

  In order to perform these operations the system has to retrieve, send and render simple data.

$$1 \cdot w_{Medium,EIQ}$$

2.6   Summary

All the calculated FP_i sums up to FP, which is the total Function Points value:

$$\begin{aligned} FP \quad &= FP_{ILF} + FP_{ELF} + FP_{EI} + FP_{EO} + FP_{EIQ} \\ &= 80 + 10 + 35 + 16 + 8 \\ &= 149 \end{aligned}$$

The total FP value is then multiplied by a constant factor $k_{i,j}$ that depends on the programming language i used to develop the software and the company gearing ratio j.

The gearing ratio is the level of a company's debt related to its equity capital, usually expressed in percentage form. Gearing is a measure of a company's financial leverage and shows the extent to which its operations are funded by lenders versus shareholders.

This final calculation gives us the number of SLOC n_{SLOC} estimated for *PowerEnJoy*:

$$\begin{aligned} n_{SLOC} \quad &= FP \cdot k_{Java,Avg} \\ &= 149 \cdot 53 \\ &= 7897 \text{ SLOC} \end{aligned}$$

Chapter 3

COCOMO

This chapter describes the estimation achieved through COCOMO II: a complex, non linear model that takes in account the characteristics of the product, people and processes. In order to generate the Constructive Cost Model we decided to use an online COCOMO II calculator on http://csse.usc.edu/tools/COCOMOII.php, using the FP Sizing Method. We also usedCOCOMO II - Model Definition Manual to make better choices of the parameters we had to insert into the model.

- **Software Size:**

  - **Unadjusted Function Points:** The value FP has been taken as parameter, so 149 is the value chosen for this field.

  - **Language:** The language of choice is Java, and not only Java EE, because the software is possibly a combination of different flavours of Java (Java SE, Java EE and Java for Android).

- **Software Scale Drivers:**

  - **Precedentedness:** Since we had a previous experience using Java SE for medium-size projects, but we have never used Java EE for developing such a big application we decided to set this parameter to Nominal.

  - **Development Flexibility:** Given that we had not strict specifications so we set this parameter to High.

  - **Architecture/Risk Resolution:** Since we have designed several documents before the actual development, including this PPD, the development of the system to be has little chances of failing, so we choose a Nominal value.

  - **Team Cohesion:** After a few days dedicated to create a efficient and fast workspace, the cohesion reached a Very High level.

- **Process Maturity:** We understood, support and follow the process so we choose a High level for this parameter.

- **Software Cost Drivers:**

  - Product

    * Required Software Reliability: Given that a failure in the software system could lead to moderate problems we choose Nominal level.
    * Data Base Size: Since we have a distributed application, the focus is on the lines of code instead of being on the size of the testing Database; so we choose a Low level parameter.
    * Product Complexity: We made an average of the various complexity areas and we choose a High level parameter.
    * Developed for Reusability: We decided to develop reusable system components, so we came up with an High level parameter.
    * Documentation Match to Lifecycle Needs: The standard level of documentation is required, so the chosen level is Nominal.

  - Personnel

    * Analyst Capability: The personnel demonstrated a Nominal level of analysis ability.
    * Programmer Capability: The personnel demonstrated efficiency working together as a team, so we chose High level for this parameter.
    * Personnel Continuity: The project will be developed always by the initial programmers, so the projects annual personnel turnover is very low (7
    * Application Experience: Since the last time the development team has worked on a so complicated project was a year ago, we have chosen Nominal for this parameter.
    * Platform Experience: The same as Application Experience ; we have chosen Nominal level for this parameter too.
    * Language and Toolset Experience: The team is quite familiar with the development, analysis and design representation, so we choose High level for this parameter.

  - Platform

    * Time Constraint: We have no relevant time constraints, so we choose Nominal level for this parameter.
    * Storage Constraint: We have no relevant storage constraint, so we choose Nominal level for this parameter.
    * Platform Volatility: Our hardware and software platforms will not change often, so we will have no volatility and therefore we choose a Low level for this constraint.
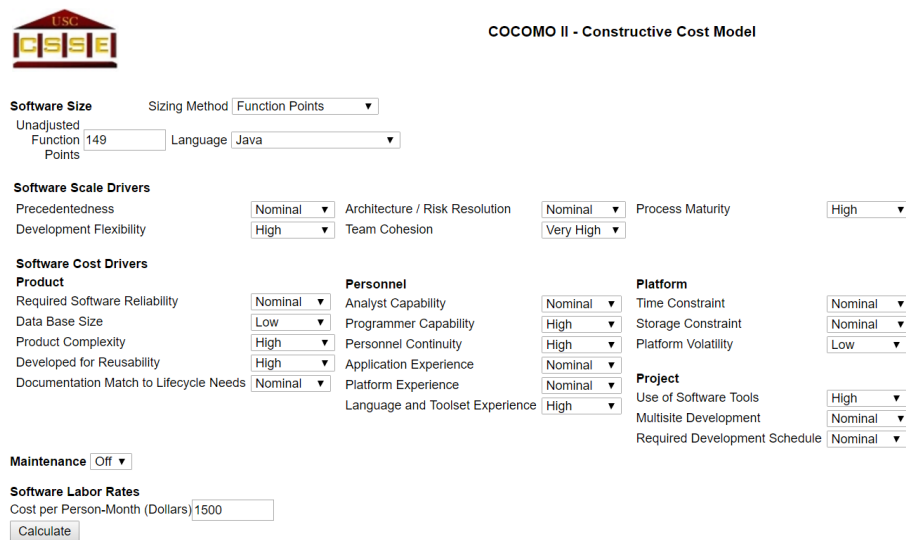
– Project

* Use of Software Tools: The team is provided of a set of strong and mature life-cycle tools, moderately integrated one into each other. So we choose an High level for this parameter.

* Multisite Development: The team is in average fully collo- cated, so the chosen level is Nominal.

* Required Development Schedule: The project is not sub- jected on a particular constraint oppression, so we have chosen Nominal for this parameter.

- **Maintenance:** This value is set to Off

- **Software Labour Rates:**

– Cost per Person-Month (Dollars): We have chosen the average value of 1500$/month for this parameter.

Figure 3.1: COCOMO Parameters

And here below we can see the result:

Results

Software Development (Elaboration and Construction)

Effort = 16.4 Person-months
Schedule = 9.2 Months
Cost = $24660

Total Equivalent Size = 7897 SLOC

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.0	1.2	0.9	$1480
Elaboration	3.9	3.5	1.1	$5918
Construction	12.5	5.8	2.2	$18742
Transition	2.0	1.2	1.7	$2959

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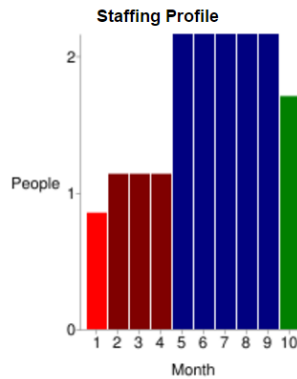
Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.1	0.5	1.2	0.3
Environment/CM	0.1	0.3	0.6	0.1
Requirements	0.4	0.7	1.0	0.1
Design	0.2	1.4	2.0	0.1
Implementation	0.1	0.5	4.2	0.4
Assessment	0.1	0.4	3.0	0.5
Deployment	0.0	0.1	0.4	0.6

Your output file is http://csse.usc.edu/tools/data/COCOMO_January_20_2017_16_05_48_617569.txt

Created by Ray Madachy at the Naval Postgraduate School. For more information contact him at rjmadach@nps.edu

Figure 3.2: COCOMO Results

Chapter 4

Task and Schedule

4.1 Task

Several tasks has been identified in our project. In the following table we can find three categories: one that labels the tasks, one for the description and the last one for the completion state to each task.

Task	Description	Completed?
T1a	RASD - Writing	Yes
T1b	RASD - Presentation	Yes
T2a	DD - Writing	Yes
T2b	DD - Presentation	Yes
T3	ITPD - Writing	Yes
T4	PPD - Writing	Yes
T5	Implementation	No
T6	Unit Testing	No
T7	Integration Testing	No
T8	System Testing	No
T9	User Acceptance - Alpha Testing	No
T10	User Acceptance - Beta Testing	No
T11	Release To Market	No

4.2 Schedule

Below there are represented a table and a diagram that show how the time has been divided between the various phase of the software life cycle ; the result is then more clear in Gantt chart. The table identifies:

- The date in which the given task starts,

- The date in which the given task ends,

- The interval in [**day**] that separates the starting date from the ending date

Task	Start	End	Interval
T1a	16/10/2016	13/11/2016	28
T1b	16/11/2016	16/11/2016	1
T2a	17/11/2016	11/12/2016	24
T2b	14/12/2016	14/12/2016	1
T3	09/01/2017	15/01/2017	6
T4	16/01/2017	22/02/2017	6
T5	04/02/2017	30/05/2017	117
T6	31/05/2017	29/06/2017	29
T7	30/06/2017	28/07/2017	28
T8	29/07/2017	29/08/2017	31
T9	30/08/2017	27/09/2017	28
T10	28/09/2017	26/10/2017	28
T11	27/10/2017	27/10/2017	1

4.3   Gant Diagram

Here is built a Gantt Diagram showing the schedule chosen for *PowerEnJoy* project tasks
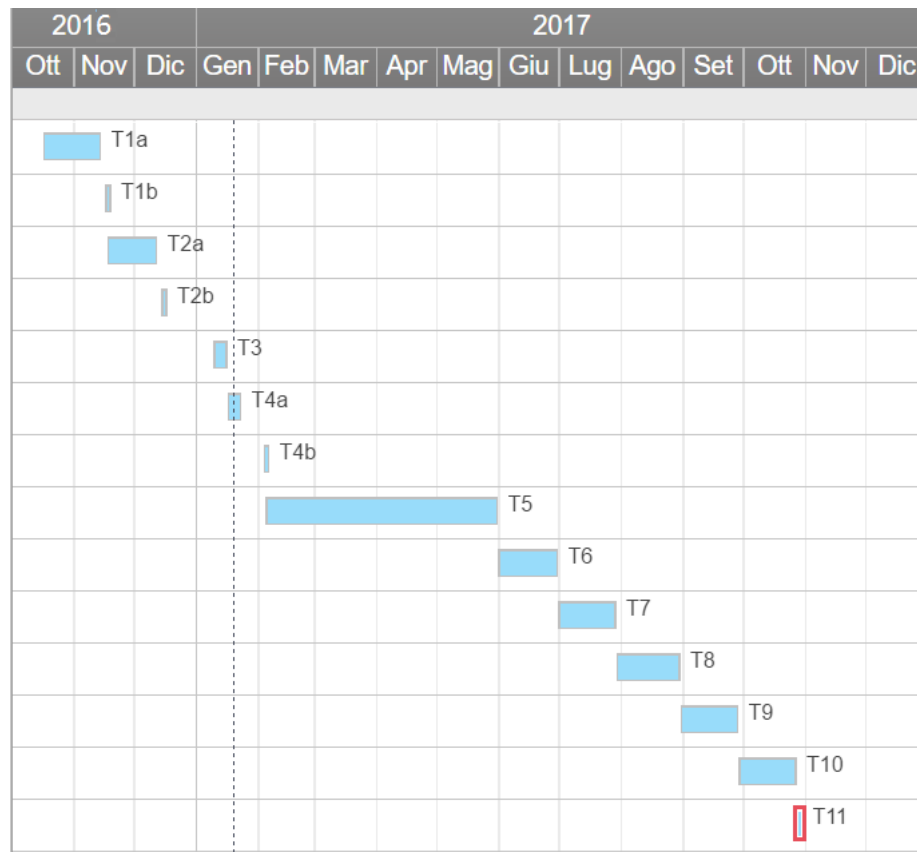
Figure 4.1: Gant Diagram

Chapter 5

Resources Allocation

This section covers the problem of allocating the human resources to each task in order to respect the identified scheduling. We are a group of two people, and we do not have any kind of concurrency in the tasks that we identified, so we are going to work together on the same task at the same time during the whole duration of the project. As shown before the project can be managed by more or less two people in nine months, and this respects the scheduling we devised.

5.1 Staff Allocation Charts

A chart is provided for the best visualization of the staff allocation. For graphical issues, some consecutive tasks are merged one into the other.

Figure 5.1: Staff allocation chart

Chapter 6

Risk

The project is subject to a series of risks with different levels of danger. In order to prevent so in necessary to have an **Proactive Risk Management Strategy** that could prevent or eventually solve some of the major risk. In order to build a good **Proactive Risk Management Strategy** we describe in this chapter the potential risk the project is subject to, categorysing them and proposing a solution for each main category. Business risk are not take into account in this chapter as we presume an carefull analysis of the market, budget and business strategy has already be done taking in account also the related risk.

- **Project Risk:**

  - The work force available is less than expected. It could be caused by developers lack of experience in programming and in project planning and management. it could potentially slow down the overall project development thus leading to the risk of an extension of some deadline. A solution for this problem could be, in scheduling phase, take into account that possibility and assign a bit more effort than needed to each task.

- **Technical Risk:**

  - Faults in reusable software components have to be repaired before these components are reused. It could be prevented by a develop strong unit tests, in fact, it reduces the error' probability and in the worst case the reparation time is minimized.

  - The database and other external software components used in the system cannot process as many operations per second as expected. It could be due to a under-estimation of the city dimension or a bigger amount of users than expected. A possible solution could be over-estimate the operations throughput in order to choose the appropriate support infrastructure.

Appendix A

Appendix

A.1 Tools

- **TeXstudio:** LaTeX editor used to write the document.
- **COCOMO Tool II:** at http://csse.usc.edu/tools/COCOMOII.php to estimate the project dimension.
- **StarUML:** To draw Gant diagram and Resources allocation.

A.2 Hours of work

In the following are listed the hours of work that each member of the group did:

1. Marco Redaelli: 16 *hours*
2. Francesco Zanoli: 16 *hours*

A.3 Version History

In the following are listed the differences between versions:

1. **22/01/2017:** First version