Politecnico di Milano

A.A. 2016 - 2017

# Software Engineering 2: PowerEnjoy

Project planning document

Version 0.3

Prasanth Ravulapalli

Fathima B

Lipika L

February 22nd 2017

## Table of contents

# 1. Introduction

## 1.1. Purpose and scope

This document presents the project management process concerning the PowerEnjoy product. The main purpose of this document is to focus mainly on the following three activities:

- Using the Function Points and COCOMO II methodologies to provide an estimation of the expected size and cost to develop the project.
- Identification and scheduling of tasks that covers all activities from the requirement identification to implementation and testing activities. We present the actual execution of such tasks with the respective associated resources.
- Risk management, with prioritization and brief solutions. This deals with possible risks that PowerEnjoy application could face during the various phases of the project.

The information contained in this document can be used to perform Process Improvement (e. g. by comparing the estimated and actual executed values), Risk management, or even as a baseline to create Management.

## 1.2. Definitions and acronyms

The following are abbreviations are used in the present document (and have not already been presented in any of the reference documents):

- COCOMO: Constructive cost model
- EI: External input
- EIF: External interface file
- EO: External output
- EQ: External inquiry
- FP: Function points
- ILF: Internal logical file
- ELF: External logical file
- SLOC: Source lines of code
- UFP: Unadjusted function points
- UI: User Interface

- GPS: Global Positioning System
- FTR: File Type Reference

### 1.3. Reference documents

The following is the list of documents that are related to this Project plan, and that totally define its context:

- Requirement Analysis and Specification Document for PowerEnjoy, RASD.pdf
- Design Document for PowerEnjoy, DD.pdf
- Integration Test Plan for PowerEnjoy, Testplan.pdf
- Sample Project planning assignment document by professor
- Lecture slides on Project Management and Cost Estimation
- Online resources: http://www.projectmanagementdocs.com/#axzz4WUDUBFHf, http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf, http://www.softwaremetrics.com/fpafund.htm

# 2. Cost estimation

In this section, we present the execution and results of the Cost estimation process. As usual, we carry forward the process in two steps:

- First, by applying the Function points methodology we try to obtain an estimation of the size of the project (in terms of lines of code).
- Second, based on the estimated size of the project obtained from the first step we used COCOMO II to approximate the effort that it would demand (in terms of person-month).

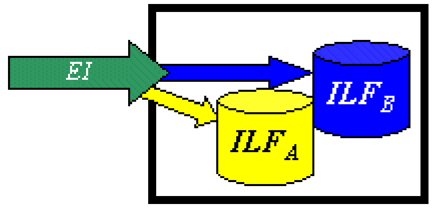
## 2.1. Function points

The Function points methodology is a widely-known mechanism to estimate the size of a software product, based on the functionalities that it is supposed to provide and their inherent complexity. It is a method to break systems into smaller components, so they can be better understood and analyzed. Along this subsection, we present a brief description of the procedure itself, and then we expose the results of applying it to our project.

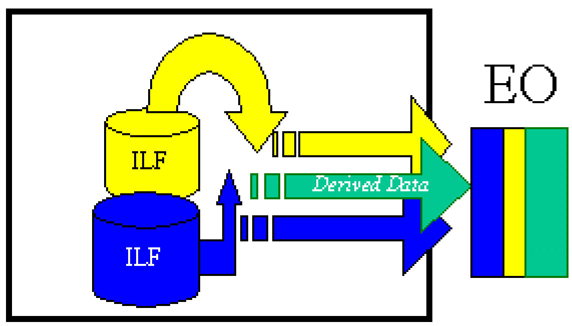
### 2.1.1. Function points methodology

Function points technique is used to evaluate the dimension of software products to be developed and maintained and to evaluate the productivity of the team. The main advantage of FP technique consists in being objective and independent from the technology used in the development. It's a unit measure for software based on user requirements. The functionalities list has been obtained from the RASD document and for each one of them has been evaluated the realization complexity.
The functionalities have been grouped in:

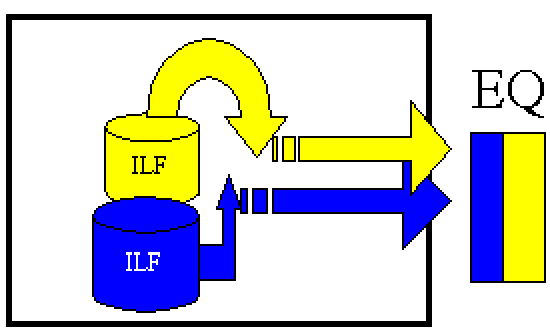
- External Inputs: is an elementary process in which data crosses the boundary from outside to inside. This data may come from a data input screen or another application. The data may be used to maintain one or more internal logical files. The data can be either control information or business information. If the data is control information it does not have to update an internal logical file. The graphic represents a simple EI that updates 2 ILFs.




- External Outputs (EO) - an elementary process in which derived data passes across the boundary from inside to outside. Additionally, an EO may update an ILF. The data creates reports or output files sent to other applications. These reports and files are created from one or more internal logical files and external interface file. The following graphic represents on EO with 2 FTR's there is derived information (green) that has been derived from the ILF's




- External Inquiry (EQ) - an elementary process with both input and output components that result in data retrieval from one or more internal logical files and external interface files. The input process does not update any Internal Logical Files, and the output side does not contain derived data. The graphic below represents an EQ with two ILF's and no derived data



- Internal Logical Files (ILF's) - a user identifiable group of logically related data that resides entirely within the applications boundary and is maintained through external inputs.
- External Interface Files (EIF's) - a user identifiable group of logically related data that is used for reference purposes only. The data resides entirely outside the application and is maintained by another application. The external interface file is an internal logical file for another application.

After the components, have been classified as one of the five major components (EI's, EO's, EQ's, ILF's or EIF's), a ranking of low, average, or high is assigned. For transactions (EI's, EO's, EQ's) the ranking is based upon the number of files updated or referenced (FTR's) and the number of data element types (DET's). For both ILF's and EIF's files the ranking is based upon record element types (RET's) and data element types (DET's). A record element type is a user recognizable subgroup of data elements within an ILF or EIF. A data element type is a unique user recognizable, non-recursive, field.

Each of the following tables assists in the ranking process (the numerical rating is in parentheses).

| FTR's | DATA ELEMENTS | | |
|---|---|---|---|
| | 1-4 | 5-15 | > 15 |
| 0-1 | Low | Low | Ave |
| 2 | Low | Ave | High |
| 3 or more | Ave | High | High |

EI Table

| FTR's | DATA ELEMENTS | | |
|---|---|---|---|
| | 1-5 | 6-19 | > 19 |
| 0-1 | Low | Low | Ave |
| 2-3 | Low | Ave | High |
| > 3 | Ave | High | High |

Shared EO and EQ Table

| Rating | VALUES | | |
|---|---|---|---|
| | EO | EQ | EI |
| Low | 4 | 3 | 3 |
| Average | 5 | 4 | 4 |
| High | 7 | 6 | 6 |

Value for transactions

| RET's | DATA ELEMENTS | | |
|---|---|---|---|
| | 1-19 | 20 - 50 | > 50 |
| 1 | Low | Low | Ave |
| 2-5 | Low | Ave | High |
| > 5 | Ave | High | High |

ILF and EIF table

| Rating | Values | |
|---|---|---|
| | ILF | EIF |
| **Low** | **7** | **5** |
| **Average** | **10** | **7** |
| **High** | **15** | **10** |

Value for transactions

The counts for each level of complexity for each type of component can be entered in a table such as the following one. Each count is multiplied by the numerical rating shown to determine the rated value. The rated values on each row are summed across the table, giving a total value for each type of component. These totals are then summed across the table, giving a total value for each type of component. These totals are then summoned down to arrive at the Total Number of Unadjusted Function Points.

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

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

### 2.1.2. Function points calculation

After an inspection of the domain of the PowerEnjoy application, we identified the different items that are used to calculate the FP. We list them below.

Note: The complexity of this items was stated based on the amount of data that they will contain and not on the actual complexity of managing them, since this is considered together with the external inputs and outputs.

#### External inputs:

- Create account user: represents the operation of creating the account of the user. It has medium complexity because the interaction with the external system like government to check if the user can drive in public.
- Take guest user details: represents the operation of saving the details of the guest user. It has simple complexity.
- Edit user account: represents the operation of editing the account of the passenger. It has simple complexity.
- Log in user: represents the operation of authenticating the passenger. It has simple complexity.

- Requesting a car: represents the operation of making a request by a user to book a car. It has medium complexity because the subsequent reactions of the system will create several ILFs.
- Cancel request: represents the operation of cancelling a request by user. It has medium complexity because the subsequent reactions of the system will delete several ILFs.
- Request to end the ride: represents the operation of request of the user to end the ride. It has medium complexity because the subsequent reactions of the system will delete several ILFs.
- User location: represents the operation of tracking the user location. It has simple complexity.

We get here 4 EIs of simple complexity, and 4 of medium complexity.

### External outputs:

- Accept request: represents the response of the system when a user accepts an incoming request by another user to accept the shared ride. It has medium complexity because the subsequent reactions of the system will create several ILFs.
- Signal to unlock the car: represents the response of the system when the system send the signal to the car to get unlocked while the user approaches the car. It has medium complexity because the subsequent reactions of the system will create several ILFs.
- Signal to the car to notify user on some violations: represents the response of the system when the system sends the signal to the car to notify him if something goes wrong or something the user is not supposed to do. It has medium complexity because of the subsequent allocation and tracking of user all the time.

We get here 3 EOs of medium complexity.

`

*External inquiries:*

- View account information of user: represents the displaying of the information of the account of the user. It has simple complexity.
- View account information guest: represents the displaying of the information of the account of the guest. It has simple complexity.
- View cars in the selected zone: represents the displaying of the information of the cars in the zone of request sent by the user. It has simple complexity.
- View accepted request information: represents the displaying of the information of a request that has been accepted. It has simple complexity.
- View the information of the user and the pick-up position: represents the displaying of the geographical information of the request. It has medium complexity because it includes data from several ILFs and EIFs.
- View incoming request information: represents the displaying of the information of the incoming request for a user. It has simple complexity.

We get here 5 EQs of simple complexity and 1 of medium complexity.

Internal logical files:

Based on the class models we presented in both the RASD and DD, the system will have the following ILFs:

- Guest User: represents the information of a guest user. It has simple complexity.
- Registered User: represents the information of a registered user. It has simple complexity.
- Request for car: represents the information of a request sent by users. It has simple complexity.
- Request for ending the ride: represents the information of a request sent by users. It has simple complexity.
- Accepted request: represents the information related to an accepted request for a specific user. It has simple complexity.
- Sharing request: represents the information of up to four sharing requests that can be merged into a single incoming request. It has simple complexity.

- Zone: represents the information of a zone in the city. It has simple complexity.
- GPS: represents the information of location of the user. It has simple complexity.

We get here 8 ILFs of simple complexity.

### External interfaces:

- User information: represents the information of the user that is provided by the user when the user creates his account for the first time. It has simple complexity.
- GPS Position, Path, and route: represent geographical data used to allocate users who booked a car and users who are willing to share the ride, and to calculate the price of a ride, trigger which discounts are to be applied. They have simple complexity.

We get here 4 EIFs of simple complexity.

The following table uses the previous information and calculates the Unadjusted Function Points.

| | **Simple** | **Medium** | **Complex** | **Total** |
|---|---|---|---|---|
| EI | 4 | 4 | | 28 |
| EO | | 3 | | 15 |
| EQ | 5 | 1 | | 19 |
| ILF | 8 | | | 32 |
| EIF | 4 | | | 12 |
| | | | **UFP** | 108 |

To get a more accurate estimation, we take the UFP and adjust them by using the fourteen parameters. The result is shown in the following table.

| | |
|---|---|
| Does the system require reliable backup and recovery? | 2 |
| Are data communications required? | 5 |
| Are there distributed processing functions? | 0 |
| Is performance critical? | 4 |

| | |
|---|---|
| Will the system run in an existing, heavily utilized operational environment? | 0 |
| Does the system require on-line data entry? | 4 |
| Does the on-line data entry require the input transaction to be built over multiple screens or operations? | 0 |
| Are the master files updated on-line? | 0 |
| Are the inputs, outputs, files, or inquiries complex? | 1 |
| Is the internal processing complex? | 4 |
| Is the code designed to be reusable? | 4 |
| Are conversion and installation included in the design? | 3 |
| Is the system designed for multiple installations in different organizations? | 3 |
| Is the application designed to facilitate change and ease of use by the user? | 2 |
| **Total** | **32** |

So, the final value of the function points is:

$$FP = UFP\times\left(0.65 + 0.01\times\sum_{i=1}^{14} Fi\right) = 105$$

Based on this, the size of the product in terms of lines of code is:

$$SLOC = 53\times FP = 53\times 108 = 5724$$

### 2.2. COCOMO II

COCOMO II is a widely-known mechanism to estimate the effort required to develop software product, based on previously obtained estimations of its size. Along this subsection, we present a brief description of the procedure itself, and then we expose the results of applying it to our project.

#### 2.2.1. COCOMO II methodology

COCOMO II model is part of a suite of Constructive Cost Model. This suite is an effort to update and extend the COCOMO software cost estimation model. COCOMO II allows one to estimate the cost, effort, and schedule when planning a new software development activity. It consists of three sub models, each one offering increased fidelity the further along one is in the project planning and design process.

For the calculation, we need some parameters:

- Scale driver (*EAF*), whose selection is based on the rationale that it is a significant source of exponential variation on a project' effort or productivity variation. Each scale driver has a range of rating levels from *very low* to *extra high*,
- Cost driver (*E*), which are the 17 multipliers used to adjust the nominal effort, person months, to reflect the software product under development. They are grouped into 4 categories: product, platform, personnel, and project. Whenever an assessment of a cost driver is between the rating level always round to the nominal rating,
- *KSLOC*, which stands for Kilo-SLOC, Thousand Source/Software Lines of Code employed to develop the product.

Through these parameters and the following equations

- Effort equation (E)
- Schedule equation (D)
- Number of people (N)

We will obtain a measure of work effort (in person-month), the time spent working on the project (in month) and the number of people involved in the project (in person).

### 2.2.2. COCOMO II calculation

For applying the COCOMO II method we will use the nominal values both for the scale and cost drivers (that amount to 1.00 and 1.0997, respectively)

The effort equation would give us a value of:

$$E = 2.94 \times EAF \times KSLOC^{E} = 2.94 \times 1.00 \times 5.724^{1.0997}$$
$$= 20.025\ person\ month$$

The schedule equation shows the following result:

$$D = 3.67 \times E^{1/3} = 3.67 \times 20.025^{0.33} = 9.96\ month$$

Then the number of people required for developing the project would be:

$$N = \frac{E}{D} = \frac{20.025\ person\ month}{9.96\ month} = 2.0093\ person$$

Which rounded up to prevent delays on the process will be a total of 3 people.

## 3. Planning process

In this section, we present the result of the planning activity. To do so, we have listed the tasks that were developed along the project, and the corresponding allocation of the resources for each one of them.

This information is summarized in the attached document schedule, which is a Gantt diagram representation of such data. This includes precedence relations as well as dates of the execution of tasks. The percentages indicate the proportion of the total effort of each person (resources) invested for that task.

Based on the presented schedule and the available information concerning the number of hours used to perform the activities, the following table was built to relate the invested time (in hours) for each one of the high-level tasks of the project.

| | RASD | DD | Code inspection | Integrations testing | Project management |
|---|---|---|---|---|---|
| **Prasanth R** | 37 | 36 | 14 | 18 | 8 |
| **Fathima B** | 13 | 10 | 4 | 5 | 3 |
| **Lipika L** | 16 | 12 | 5 | 7 | 4 |

## 4. Risk management

This section shows the results of the risk analysis process. The table below relates, for each risk, the following elements:

- Description: brief explanation of the situation that could affect the project or the software product to be developed.
- Risk type: the type of risk, derived from the nature of the effects associated to it. Risks can be categorized into the following:
    - Project risks
    - Technical risks
    - Business risks (Market, Strategic, Sales, Management, Budget)
    - Known risks
    - Predictable risks
    - Unpredictable risks
- Probability: both quantitative and qualitative values that assess the probability that the event occurs. The qualitative value is one among Very low, Low, Medium, High, and Very high; they are mapped into the quantitative values 1, 2, 3, 4, and 5, respectively.
- Effect: both quantitative and qualitative values that assess the impact for the project and product of the event, whenever the risk is materialized. The qualitative value is one among Insignificant, Serious, Critical, and Catastrophic; they are mapped into the quantitative values 1, 2, 3, and 4, respectively.
- Priority: it is calculated based on the probability and the effect. This is used to determine which risks are to be monitored more carefully during the execution of the project. The value is calculated with the following formula:

$$Priority = Probability \times Effect \times 4$$

We will try to assess main risks that the project development may face in the following:

One of the risks can be members of the team not able to finish his/her assignments either due to sickness or due to some personal issues. This falls under Project risk as this is directly proportional to the project work hours and this can’t be avoided at any cost. The effect of such risk can be treated as serious and in the scale, it can be illustrated as 2. The only fix for this kind of risk is to provide flexible time lines to finish the task assigned by considering all such scenarios.

| Risk type | Description | Probability | Probability value | Effect | Effect value | Priority |
|---|---|---|---|---|---|---|
| Project | One of the team member gets sick and is not able to finish his/her assignments. | Low | 2 | Serious | 2 | 16 |
| Project | One of the team members has personal issues that do not allow him/her to work in the project. | Low | 2 | Serious | 2 | 16 |
| Project | The team misunderstands the requirements set provided by the customer. | Low | 2 | Catastrophic | 4 | 32 |
| Technical | The development team has not enough skills to use the tools to implement the software. | Low | 2 | Critical | 3 | 24 |

| | | | | | | |
|---|---|---|---|---|---|---|
| Technical | The components of the system cannot communicate each other because of connection problems. | Medium | 3 | Serious | 2 | 24 |
| Technical | The external systems do not behave as expected (the interfaces that they provide are no longer working properly). | High | 4 | Serious | 2 | 32 |
| Business | The calculation of the fee is not actually fair either for passenger or taxi drivers. | High | 4 | Serious | 2 | 32 |
| Business | The allocation of an available taxi driver for the passengers is taking too much time. | Medium | 3 | Serious | 2 | 24 |

## 5. Hours of work

Prasanth R: 8 hours

Fathima: 3 hours

Lipika: 4 hours