

**P**roject **P**lan **D**ocument (**PPD**)

Computer Science and Engineering (CSE)

Software Engineering 2 Project

Year 2015/16

*Date: 30/01/2016 – Version: 1.0*



**STUDENTS:**

*Martino Andrea (****788701****)*

*Marchesani Francesco (****852444****)*

**PROFESSOR:**

*Mirandola Raffaela*

# Introduction

## Table of contents

1. Introduction 2

1.0. Table of contents 2

1.1. Revision History 3

1.2. Purpose and Scope 3

1.3. Glossary 4

1.4. List of reference documents 4

2. Function Points and COCOMO II estimations 5

2.1. Function Points technique 5

2.2. Function Points estimation 7

2.3. COCOMO II technique 7

2.4. COCOMO II estimation 8

3. Project tasks and schedule 8

3.1. Project tasks 8

3.2. Project schedule 8

4. Resources allocation 8

5. Risks analysis of the project 8

5.1. Risks identification, relevance and recovery actions 8

6. Appendix 10

6.1. Hours of work 10

## Revision History

We will keep the **revision history** of the **Project Plan Document** (**PPD**) in this chapter.

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author(s)** | **Summary** |
| 1.0 | 30/01/2016 | Andrea Martino, Francesco Marchesani | Document  Creation |

## Purpose and Scope

This **Project Plan Document (PPD)** contains information about several topics about project planning. We will discuss these topics in detail in the next chapters, with respect to the standards. We will do some assumptions in order to clarify specific points of the project (e.g. the implementation phase, not covered in the real course).

This **document** is coherent with *the official guidance template* of the project on the *Beep platform* with some additional chapters for the sake of completeness.

As we said for the past documents, it is important to underline that some parts of this document may evolve in the future (this may occurs for several causes).

Anyway, we will try to maintain coherence as much as possible.

Here is a resume of the steps of the project, with the related deadlines (in green documents already delivered, in yellow the current document):

The main scope of this **PPD** (*Project Plan Document*) is to give an overall guidance to the **project-planning phase** of the **project**, which is *myTaxiDriver* (**Software Engineering 2 project** of year 2015/16 - **Politecnico di Milano**).

We will focus in particular on the **algorithmic methods** used in practice (*Function points* and *COCOMO II*), on the **tasks identification**, on the **project scheduling**, on the **resources allocation** and on the **risk analysis**.

## Glossary

* **RASD**: *Requirements Analysis and Specification Document*
* **DD**: *Design Document*
* **ITPD**: *Integration Test Plan Document*
* **PPD**: *Project Plan Document*
* **FPs**: *Function Points*
* **UFPs**: *Unadjusted Function Points*
* **COCOMO**: *Constructive Cost Model*
* **mTS**: *myTaxiService*
* **SE**: *Software Engineering*

**Note:** *for the full Glossary may be helpful to see also the paragraph 1.5 of the RASD 2.0, paragraph 2.3 of DD and paragraph 1.3 of ITPD.*

## List of reference documents

Here is a list of the **reference documents** for the current *Project Plan Document (PPD)* of *myTaxiService*:

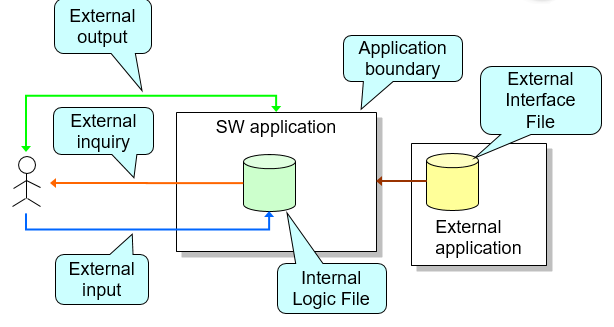
* **Project Description** (from *Beep* platform)
* **RASD 2.0 [RASD Revision]** (hosted on *GitHub Repository*)
* **Design Document [DD]** (hosted on *GitHub Repository*)
* **Integration Test Plan Document [ITPD]** (hosted on *GitHub Repository*)

# Function Points and COCOMO II estimations

## Function Points technique

Historically, **Allan Albrecht** (from **IBM**) introduced **function points** in **1979** in his work "*Measuring Application Development Productivity*". The functional user requirements of the software are grouped in **five macro-categories:**

* **Internal Logical File (ILF):** homogeneous set of data used and managed by the application.
* **External Interface File (EIF)**: homogeneous set of data used by the application but generated and maintained by other external applications.
* **External Input**: elementary operation to elaborate data coming from the external environment (from users).
* **External Output**: elementary operation that generates data for the external environment. It usually includes the elaboration and a proper representation of data from logic files.
* **External Inquiry**: elementary operation that involves input and output (e.g. specific requests of the user in order to visualize his/her information). This category does not implies significant elaboration of data from logic files.



The general idea is to give a **weight** for each category, with respect to a "*difficulty range*” (Simple, Medium and Complex). Here it is the related table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Function types**   * N. Inputs * N. Outputs * N. Inquiry * N. ILF * N. EIF | **Weight** | | |
| **Simple**  3  4  3  7  5 | **Medium**  4  5  4  10  7 | **Complex**  6  7  6  15  10 |

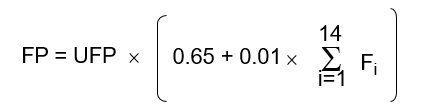
Then it is possible to compute the amount of weights: the total number obtained (**UFPs**, *Unadjusted Function Points*) is an indicator of the project size.



Optionally we can compute also the **Function Points** with respect to an adjustment based on **14 General System Characteristics [GSCs]** that can give a final correction of *±35%*. These GSC are:

|  |  |
| --- | --- |
| 1. *Data Communication* 2. *Distributed data processing* 3. *Performance* 4. *Heavily used configuration* 5. *Transaction rate* 6. *Online data entry* 7. *End user efficiency* | 1. *Online update* 2. *Complex processing* 3. *Reusability* 4. *Installation ease* 5. *Operational ease* 6. *Multiple sites* 7. *Facilitate change* |

The formula for the **Function Points** computation is the following:



Where **Fi** are the coefficients for **GCSs** ∈ {0, 1, 2, 3, 4, 5} depending from the specific weight.

## Function Points estimation

|  |  |
| --- | --- |
| **mTS Function Types** | **Total weight** |
| * N. Inputs |  |
| * N. Outputs |  |
| * N. Inquiry |  |
| * N. ILF |  |
| * N. EIF |  |
| * **TOTAL NUMBER OF UFPs** |  |

## COCOMO II technique

The **Constructive Cost Model** (**COCOMO**) is an algorithmic software cost estimation model developed by Barry W. Boehm. The model uses a basic **regression formula** with parameters obtained from historical project data and current as well as future project characteristics.

We will focus on **COCOMO II**, which is the improved version of the original model.

By the way, we will use this approach to estimate **effort** and **cost** of *mTS* project.

## COCOMO II estimation

# Project tasks and Schedule

SEE SLIDES 43-44-45 of Lesson PM 1 slides

## Project tasks

## Project schedule

# Resources Allocation

# Risks Analysis of the project

## Risks identification, relevance and recovery actions

Here we can see in a compact way the **risk analysis** of myTaxiService project. Note that we will consider the main categories of possible risk. Of course it is impossible to understand at the beginning each possible source of risk (unless the risk estimator is a sort of Cassandra, as in the Greek mythology).

We will give a **relevance** with respect to a five points scale (where *0=NEGLIGIBLE, 1=VERY LOW, 2= LOW, 3=MEDIUM, 4=HIGH, 5=VERY HIGH*):

|  |  |  |
| --- | --- | --- |
| **Risk** | **Relevance** | **Recovery actions** |
| Organizational finance problems | 3/5 | Prepare a briefing document for senior management of the project in order to show how the project is making a very important contribution to the goals of the business and presenting reasons why cuts to the project budget would not be cost-effective, (they can only contribute to decrease the income). |
| Recruitment problems | 2/5 | Tell the customers about probability of delays and search for pre-developed components that can fit myTaxiService project (without developing similar components from scratch). |
| Staff illness | 4/5 | Considering that the probability of illness in winter is high, in case of illness we can perform team re-organization and meetings about work done by ill colleagues (in order to understand each other’s job). |
| Defective components | 2/5 | Substitute potentially defective components with bought-in components of known reliability, in order to avoid problems. |
| Requirements changes | 3/5 | Derive traceability information to assess requirements change impact (it can be catastrophic in certain cases). We also need to perform information hiding in the project design in order to avoid chains of changes in the architecture. |
| Organizational restructuring | 1/5 | As said for *Organizational finance problems*, we can prepare a document for a senior management to show that the project is very important for the business of the company and it is important to fulfill respecting the deadlines. |
| Database performances | 2/5 | Consider the possibility of a general improvement of the database, with better performances. |
| Underestimated development time | 4/5 | Consider the “*make or buy*” tradeoff, searching for suitable components of the project developed externally. |

# Appendix

## Hours of work

* **Andrea Martino**: Hours(x)
* **Francesco Marchesani**: Hours(x)