Politecnico di Milano

A.A. 2015-2016

Software Engineering 2: “MyTaxiService”

Project planning document

Version 0.1

Ivana Salerno

Alexis Rougnant

Daniel Vacca

February 2nd 2016

Table of contents

1. Introduction 3
1.1. Purpose and scope 3
1.2. Definitions and acronyms 3
1.3. Reference documents 3
2. Cost estimation 5
2.1. Function points 5
2.1.1. Function points methodology 5
2.1.2. Function points calculation 6
2.2. COCOMO II 9
2.2.1. COCOMO II methodology 9
2.2.2. COCOMO II calculation 10
3. Planning process 11
4. Risk management 12

1. Introduction

1.1. Purpose and scope

This document presents a fraction of the project management process concerning the MyTaxiService product. In particular, we treat here the following three activities:

- Size and cost estimation of the project, by means of the Function Points and COCOMO II methodologies.
- Identification and scheduling of tasks. We present the actual execution of such tasks with the respective associated resources.
- Risk management, with prioritization and brief solutions.

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 reports (e. g. to expose to the customer and the stakeholders relevant information concerning the development process).

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
- SLOC: Source lines of code
- UFP: Unadjusted function points

The following definitions are relevant for this document (and have not already been presented in any of the reference documents):

1.3. Reference documents

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

- õ MyTaxiService project AA 2015-2016 description
- õ RASD for MyTaxiService, by Ivana Salerno, Alexis Rougnant and Daniel Vacca
- õ DD for MyTaxiService, by Ivana Salerno, Alexis Rougnant and Daniel Vacca

- Integration Test Plan for MyTaxiService, by Ivana Salerno, Alexis Rougnant and Daniel Vacca
- Project planning assignment description for project AA 2015-2016
- Lecture slides on Project Management and Cost Estimation

The following documents have been used as conceptual references to extract definitions, and as guidelines to apply the methodologies described later on:

- COCOMO Model Definition Manual

2. Cost estimation

In this section we present the execution and results of the Cost estimation process. As usual, we proceeded in two steps: first, an estimation of the size of the project (in terms of lines of code) was obtained by applying the Function points methodology; second, based on the estimated size of the project 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. 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. The idea behind this technique consists in quantifying the functionalities provided by the final product in the matter of data and processes significant for final consumers. 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:

- Internal Logical Files: it represents a set of homogeneous data handled by the system. In MyTaxiService application this corresponds to all the information of the system contained in the database, like actors and requests.
- External Interfaces: it represents a set of homogeneous data used by the application but handled by external application.
- External Inputs: it represents an elementary operation that allows input of data in the system.
- External Outputs: it represents an elementary operation that creates a bit-stream towards the outside of the application.
- External Inquiries: it represents an 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 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 MyTaxiService application, we identified the different items that are used to calculate the FP. We list them below.

- **Internal logical files:**
  Based on the class models we presented in both the RASD and DD, the system will have the following ILFs:
  - Taxi driver: represents the information of a taxi driver. It has simple complexity.
  - Passenger: represents the information of a passenger. It has simple complexity.
  - Request for service: represents the information of a request sent by a passenger. It has simple complexity.
  - Accepted request: represents the information related to an accepted request for a specific passenger. It has simple complexity.
  - Incoming request: represents the information related to the request which is sent to the taxi driver. 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.

  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.

  We get here 7 ILFs of simple complexity.

õ **External interfaces:**

- Taxi driver information from Milano Government: represents the information of the taxi driver that is provided by the Milano Government when the driver creates his account for the first time. It has simple complexity.
- GPSPosition, Path, and Itinerary: represent geographical data used to allocate taxi drivers and passengers, and to calculate the fee of a trip. They have simple complexity.

We get here 4 EIFs of simple complexity.

õ **External inputs:**

- Create account passenger: represents the operation of creating the account of the passenger. It has simple complexity.
- Create account taxi driver: represents the operation of creating the account of the taxi driver. It has medium complexity because the interaction with the Milano Government external system.
- Edit account passenger: represents the operation of editing the account of the passenger. It has simple complexity.
- Edit account taxi driver: represents the operation of editing the account of the taxi driver. It has simple complexity.
- Log in passenger: represents the operation of authenticating the passenger. It has simple complexity.
- Log in taxi driver: represents the operation of creation authenticating the taxi driver. It has simple complexity.
- Make a request: represents the operation of making a request by a passenger. It has medium complexity because the subsequent reactions of the system will create several ILFs.
- Cancel request passenger: represents the operation of cancelling a request by a passenger. It has medium complexity because the subsequent reactions of the system will delete several ILFs.
- Cancel request taxi driver: represents the operation of cancelling a request by a taxi driver. It has medium complexity because the subsequent reactions of the system will delete several ILFs.
- Update availability: represents the operation of updating the availability of a taxi driver. It has simple complexity.

We get here 6 EIs of simple complexity, and 4 of medium complexity.

õ **External outputs:**

- Accept request: represents the response of the system when a taxi driver accepts an incoming request. It has medium complexity because the subsequent reactions of the system will create several ILFs.
- Zone allocation for the taxi driver: represents the response of the system when a taxi driver sets himself to available. It has medium complexity

because of the subsequent allocation and maintenance of the driver inside a zone queue.

We get here 2 EOs of medium complexity.

õ **External inquiries:**

- View account information passenger: represents the displaying of the information of the account of the passenger. It has simple complexity.
- View account information taxi driver: represents the displaying of the information of the account of the taxi driver. It has simple complexity.
- View request status: represents the displaying of the information of the status of a request sent by the passenger. 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 taxi driver 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 taxi driver. It has simple complexity.

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

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

	Simple	**Medium**	**Complex**	**Total**
EI	6	4		34
EO		2		10
EQ	5	1		19
ILF	7			49
EIF	4			20
			UFP	132

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:

$$FP = UFP \times \left(0.65 + 0.01 \times \sum_{i=1}^{14} Fi\right) = 129$$

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

$$SLOC = 46 \times FP = 46 \times 128 = 588$$

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 submodels, 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:

$$E = 2.94 \times EAF \times KSLOC^{E} = 2.94 \times 1.00 \times 5.888^{1.0997} = 20.66\ person\ month$$

The schedule equation shows the following result:

$$D = 3.67 \times E^{SE} = 3.67 \times 20.66^{0.32} = 9.67\ month$$

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

$$N = \frac{E}{D} = \frac{20.66\ person\ month}{9.67\ month} = 2.14\ person$$

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. In order 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 amount 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**
Ivana Salerno	27	34	20	24	8
Alexis Rougnant	31	28	24	28	8
Daniel Vacca	31	29	24	28	8

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. It can be one among Project, Technical and Business.
- Probability: both quantitative and qualitative values that assess the probability that the event actually 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:

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

- Solution: a brief description of the actions to be taken either to prevent or to react the risk.

ID	Risk type	Description	Probability	Probability value	Effect	Effect value	Priority	Solution
Rk-001	Project	One of the team member gets sick and is not able to finish his/her assignments.	Low	2	Serious	2	16	The schedule of the activities should be flexible enough to reassign the remaining tasks among the other members.
Rk-002	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	The schedule of the activities should be flexible enough to reassign the remaining tasks among the other members.
Rk-003	Project	It is impossible or very difficult to communicate with the customer and stakeholders to get important information.	Very low	1	Critical	3	12	Fix proper communication channels and conditions with both customer and stakeholders.
Rk-004	Project	The customer requests changes in the requirements that strongly modifies the structure of the product to be delivered.	Very low	1	Catastrophic	4	16	Requirements have to be frozen in agreement with the customer, before the implementation phase.
Rk-005	Project	The team misunderstands the requirements set provided by the customer.	Low	2	Catastrophic	4	32	Create mechanisms to validate the perspective that the team members have about the requirements.
Rk-006	Project	One of the team member quits.	Very low	1	Catastrophic	4	16	The schedule of the activities should be flexible enough to reassign the remaining tasks among the other members.
Rk-007	Technical	The team members do	High	4	Catastrophic	4	64	The schedule plan should take

		not have enough time to complete the project.						into account the potential additional work that the team members will have to develop along the semester. Eventual negotiations with the teacher can be considered.
Rk-008	Technical	The development team has not enough skills to use the tools to implement the software.	Low	2	Critical	3	24	The selection of the development languages and tools must take into account the skills of the members.
Rk-009	Technical	The physical equipment is damaged.	Very low	1	Catastrophic	4	16	Have backup of the information stored in the devices. Buy new equipment to replace the damaged ones.
Rk-010	Technical	The software application is affected by a security problems.	Low	2	Critical	3	24	Ensure the security of the system during the development phase.
Rk-011	Technical	The components of the system cannot communicate each other because of connection problems.	Medium	3	Serious	2	24	Have alternative communication ways in case of failure.
Rk-012	Technical	The external systems do not behave as expected (the interfaces that they provide are no longer working properly).	High	4	Serious	2	32	Find a different provider of the service or change the implementation to adapt to the new interfaces.
Rk-013	Business	The calculation of the fee is not actually fair either for passenger or	High	4	Serious	2	32	The calculation of the fee should be easily modifiable.

		taxi drivers.						
Rk-014	Business	The allocation of an available taxi driver for the passengers is taking too much time.	Medium	3	Serious	2	24	Load tests should be performed to ensure an optimal response. The implementation has to be done with efficient algorithms.