Project Report

MyTaxiService

Pozzi Matteo

Scandroglio Stefano

Table of contents

1 Introduction

In this document we present an evaluation of the cost required to develop the myTaxiService application. For the analysis of the size of the project we will use the Function Points method, while for the effort estimation we will use the COCOMO II method. In particular we are going to compare the result obtained using those methods with the real data relative to the development of myTaxiService.

2 Function Point approach

In this chapter we use the FPs approach to estimate the size of the application by taking into account the different functionalities that it will provide.
The following table shows the weights assigned to each Function Type depending on its complexity:

Function Type	**Complexity**		
	Simple	**Medium**	**Complex**
Internal Logic File (ILF)	7	10	15
External Logic File (ELF)	5	7	10
External Input	3	4	6
External Output	4	5	7
External Inquiry	3	4	6

2.1 Internal Logic Files

Our Database will store informations about Users, Drivers, Zones, Requests, Reservations, Notifications and Taxis. Users, Zones and Taxis are simple entities because of their simple structure. Drivers, Requests and Reservations are of medium complexity as they contain many fields. Finally, Notifications are complex entities as they contain many fields in their tables due to the fact that they can be of different types.
As a result we get:

$$7 * 3 + 10 * 3 + 15 * 1 = 66 \text{ FPs}$$

2.2 External Logic Files

In our application there aren't any ELFs, because we don't use data created and managed by other applications.

2.3 External Input

The following are all the actions that Users and Drivers can perform:

- Simple: Login/Logout, Registration and Availability Editing are simple actions as they involve a small number of operations.

  $$3 * 3 = 9 \text{ FPs}$$
- Medium: Request, Reservation, Reservation Editing and Account Modification are actions of Medium complexity.

  $$4 * 4 = 16 \text{ FPs}$$

- Complex: Accept/Refuse Notifications is a complex action because it involves interaction with many entities (i.e. User, Driver, Zone, Request).

6 * 1 = 6 FPs

2.4 External Output

We have one External Output which is the Notification. This output is of two different types depending on to whom is sent (UserNotification or DriverNotification), therefore it is considered as two different complex outputs and so we get:

7 * 2 = 14 FPs

2.5 External Inquiry

There are two types of External Inquiries which are:

- *Request to view account details*: this is a simple inquiry as it involves just one entity and it is a simple operation;
- *Request to view Request/Reservation details*: this is a medium complexity inquiry as it involves two entities but it is a more complex operation.

3 * 1 + 4 * 1 = 7 FPs

2.6 Total FPs

In conclusion, we have obtained the following Unadjusted Functional Points (UFP) results.

Function Type	**Value**
Internal Logic Files	66
External Logic Files	0
External Inputs	31
External Outputs	14
External Inquiries	7
Total	**118**

3 COCOMO II Approach

Using this method we estimate the effort in terms of person/months, using the following formula:

$$Effort = 2.94 * EAF * (KSLOC)^{E}$$

To derive the number of KSLOC from the FPs, we use the average of conversion factor of 46 specified at the following link www.qsm.com/resources/function-point-languages-table.

$$46 * 118\ FPs = 5428\ SLOC$$

To define E parameter, we derive the following Scale Factors (SFs).

- PREC = 0.04 (low)
- FLEX = 0.02 (high)
- RESL = 0.03 (nominal)
- TEAM = 0.01 (very high)
- PMAT = 0.04 (low)

$$E = B + sum\{i\}\ SF[i] = 0.91 + 0.14 = 1.05$$

To define EAF, we have defined some Cost Drivers:

- Reliability (RELY) = 0.92 (low) : because software failures don't have critical consequences and are easily recoverable;
- Analyst Capability (ACAP) = 0.85 (high) : because we deeply analized the problem and the domain of application in the RASD;

- Programmer Capability (PCAP) = 0.88 (high) : this parameter is evaluated according to our degree of cooperation;
- Applications Experience (APEX) = 1.1 (low) : this parameter is evaluated according to our previous experience on J2EE framework and in web projects;
- Programming Language Experience (LEXP) = 1.09 (low) : because we have no experience on J2EE programming.

$$EAF = product\ of\ Cost\ Drivers = 0.83$$

Now we can evaluate the effort value:

$$Effort = 2.94 * 0.83 * (5.428)^{1.05} = 14.41\ person/months$$

Now we can compute the schedule equation based on the effort value that gives an estimation of the duration of the project:

$$Duration = 3.67 * (Effort)^{SE} = 8.35\ months$$

Where SE is the schedule equation exponent derived from the Scale Drivers:

$$SE = 0.28 + 0.2*(E - B) = 0.308$$

Now we compute the number of people N required to develop the application:

$$N = Effort / Duration = 1.73\ people \rightarrow \textbf{2 people}$$

4 Task Identification and Resources Allocation

In this section we define the tasks and how they have been shared among us.

Months	**Documents**	**Hour/Person**
15 Oct 2015 - 6 Nov 2015	RASD	30
6 Nov 2015 - 3 Dec 2015	Design Document (DD)	20
7 Jan 2016 - 21 Jan 2015	Integration Testing Plan Document (ITPD)	9
21 Jan 2016 - 2 Feb 2016	Project Report	3

Implementation and Testing: for this task we have estimated a total working time of 120 hours / person.

With such an estimation, the total amount of time spent on the development of this project is 364 hours.

*Real Effort = 364 / (15 * 4) = 6.07 person/months*

where we have supposed that our working time is of 3 hours/day.

With this real effort the real duration for working on this project is:

Real Duration = Real Effort / N = 6.07 / 2 = 3.035 months

Conclusion: the effort and duration estimated with COCOMO II are oversized with respect to the real ones, maybe because of a wrong estimation of both the Cost Drivers and the Scale Drivers or maybe we have underestimated the implementation and testing duration.

5 Risks and Strategies

Risks	Probability	Effects	Proactive Strategies
Team member unavailability during the project development	Medium	Moderate	Organize the team in such a way that all members are able to replace a missing member of the team
Hardware and Software failures	Low	Catastrophic	Always keep a backup of the project in a more secure location
Changes in the requirements	Medium	Serious	Maximize information hiding in the design phase
Goldplating	Low	Low	Focus only on the core functionalities for the first release of the application