

**My Taxi:**

**Project Plan**

Authors:

Giovanni Bucci

Riccardo De Togni

1. **Introduction**

This document aims to evaluate the effective effort needed to entirely develop MyTaxi project in all its features, and after that give an hypothesis on how to schedule the development. The evaluation starts with a Function Point Analysis, which will provide a rough estimate of the SLOC (Source Lines of Code). Then with SLOC value, we will proceed with a COCOMO II analysis to calculate correspondent Effort and Duration.

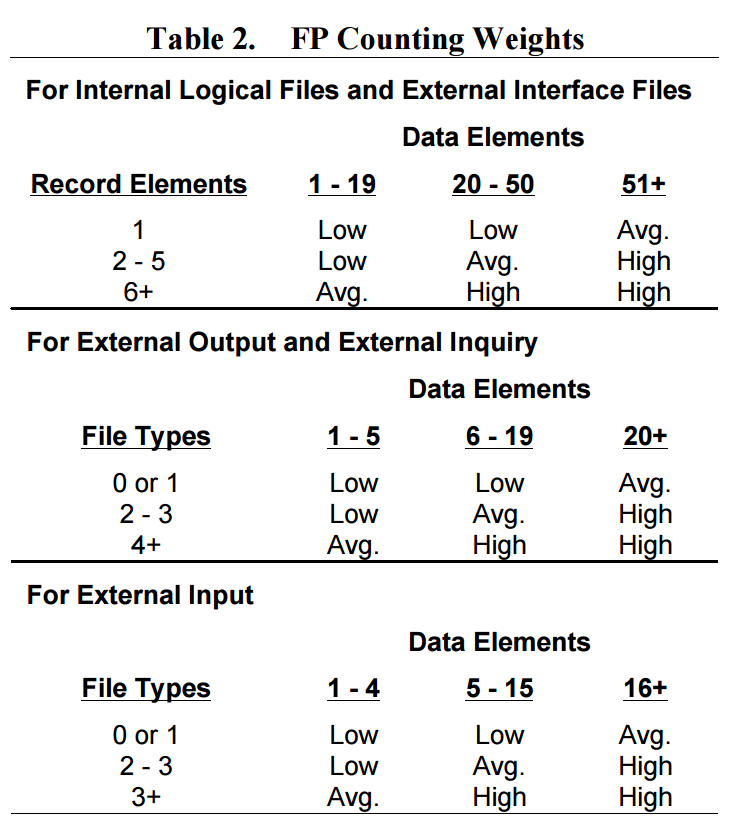
The second part of the document will explain the project schedule through tasks identification and allocation to team members.

1. **Project Size and Cost evaluation**
   1. **Function Point Analysis**

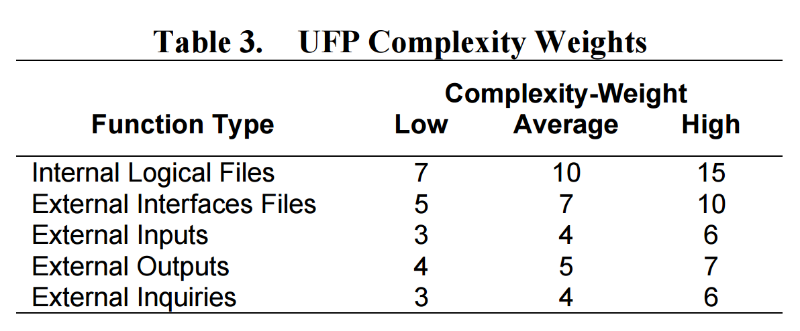
In order to perform the function point analysis we identify all the features of the project such that Internal Logical Files, External Interface Files and External Inquiries, Inputs and Outputs. To evaluate the complexity and the correspondent function point amount of each functionality we referred to the COCOMO II Function Point Weight Tables available at:

<http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf>

The first Table is used to evaluate the complexity:



The second one, given a complexity returns a function point number.



The total amount of function points will represent the UFP (Unadjusted Function Points) converted in SLOC through a language-dependent factor. Since the lack of any implementation technique constraint all over the project documentation, we are free to adopt any language in the evaluation. For this evaluation, we will use Java Enterprise Edition, which has a converting factor of 46 SLOC/FP.

* + 1. **Internal Logical Files**

According to the E-R Diagram [DD 5.1] the system stores data about Guests, Users, Administrators (a particular type of User), Taxi Drivers, Requests, Reservations and Queues. The functionality of these entities has been deeply analysed in the previous documents so we will just summarize the complexity in the following table:

|  |  |  |
| --- | --- | --- |
| Internal Logical Files | Complexity | Function Points |
| User | High | 15 |
| Guest | High | 15 |
| Taxi Driver | Average | 10 |
| Request | High | 15 |
| Reservation | Average | 10 |
| Queue | Low | 7 |
| Total: |  | 72 |

* + 1. **External Interface Files**

The system has an interface with three different external component: Localization System, Mail Server and SMS Server. The localization system given a GPS Signal provides a couple of coordinates but we expect an intense flow of data since the number of user could be very high. The mail server and the SMS server manages the delivery of messages. The complexity is presented in the table below:

|  |  |  |
| --- | --- | --- |
| External Interface Files | Complexity | Function Points |
| Localization System | Average | 7 |
| Mail Server | Average | 7 |
| SMS Server | Average | 7 |
| Total: |  | 21 |

* + 1. **External Inputs**

The input are divided by the entity that perform it:

* Users can Login, Logout, Register to the system.
* Taxi Drivers can Login, Logout, Register, give availability, accept/deny request

Login and Logout inputs have been count just once although they are a functionality proper of both Users and Taxi Drivers. Registration has been differentiated because the two procedures involve different steps.

|  |  |  |
| --- | --- | --- |
| External Inputs | Complexity | Function Points |
| Login/Logout/Register | Low | 3x3 |
| Taxi Registration | Average | 4 |
| Accept/Deny Request  Give Availability | Low  Low | 3  3 |
| Total: |  | 19 |

* + 1. **External Inquiries**

As we did for Inputs we will divide Inquiries by entity:

* Inquiries that involve users: Create request/Reservation, manage personal data
* Inquiries that involve taxi drivers: Manage Personal Data

|  |  |  |
| --- | --- | --- |
| External Inquiries | Complexity | Function Points |
| Create request/reservation  Manage Personal Data | Average  Low | 4x2  3 |
| Manage Personal Data | Low | 3 |
| Total: |  | 14 |

* + 1. **External Outputs**

The application alert the taxi driver with an incoming Request, and alert a user when its reservation has forwarded a request to the system (i.e. the reservation time has come).

|  |  |  |
| --- | --- | --- |
| External Output | Complexity | Function Points |
| Taxi Alert | Low | 4 |
| User Alert | Low | 4 |
| Total: |  | 8 |

* + 1. **Unadjusted Function Points**

Now we proceed with the evaluation of UFP:

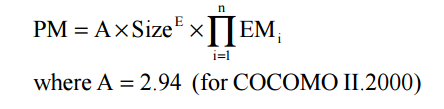
Therefore, the number of Source Lines of code will be:

* 1. **COCOMO II Analysis**

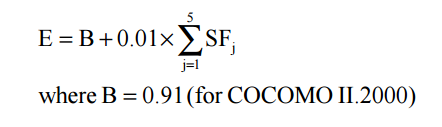
Once estimated the size of the system-to-be, it is possible to make a first prevision of the software cost in terms of time spent and people allocated to the project.

The COCOMO (Constructive Cost Model) approach is based on effort and duration estimation using ad-hoc formulae that consider many parameters derived from previous projects data and future previsions.

The formula used for effort calculation is the following:

The effort is calculated in Persons-Month; in this case, the parameter Size is derived from the Function Points evaluation done before, E and EM are factors derived from respectively scale factors and cost drivers, i.e. elements that let the project manager consider the system necessities and have a preview of what the team needs to deal with.

The exponent E is obtained from the following expression:

SF are the mentioned scale factors, consider elements like developers experience, team cohesion, and project specifications.

Their value is decided with the help of the dedicated table.

Let’s analyse them in detail:

**Precedenceness:**

It reflects the previous experiences related to this kind of projects. In this case the team had already developed similar systems so the nominal value will be reflecting the actual situation.

**Development flexibility:**

It reflects the flexibility of costraints in the development process. The stakeholders set

precise specifications but without letting the development team free to choose the majority of implementation details, for this reason this value will be nominal.

**Risk resolution:**

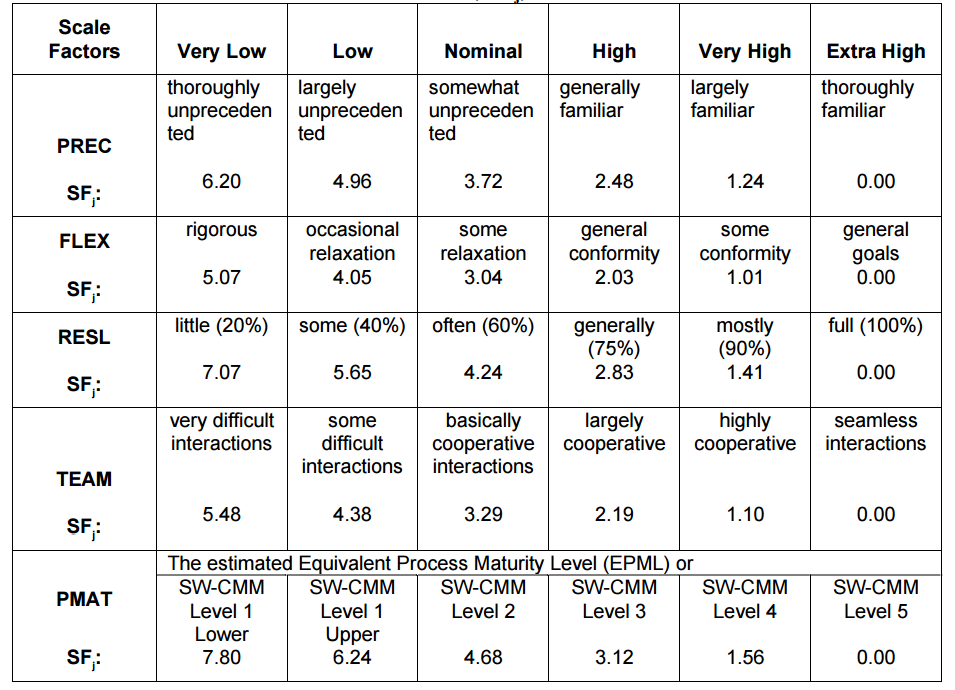
Reflects the extent of risk analysis. A well developed risk management plan corresponds to a high value in the table. In this case the value considered is nominal.

**Team cohesion:**

Reflects how the development team know each other and cooperate. In this case the team is united; people communicate and cooperate in an efficient way, so it is possible to consider a high value for this parameter.

**Process maturity:**

Reflects team maturity regarding project development management. Organization and adopted techniques influence this factor. For this project, the correct value is the nominal one since the project is developed under standard conditions.



In this particular project the values considered are often the ones in the “Nominal” column, since the project conditions are standard and often idealized. The only higher value is the one regarding Team Cohesion.

With the chosen factors, we can calculate the exponent E with the already presented formula.

E = 0.91 + 0.01 x (3.72 + 3.04 + 4.24 + 2.19 + 4.68) = 0.91 + 0.01 x 17.87 = 0.91 + 0.1787 ≈ 1.08

Now it is necessary to calculate effort multipliers, and it is done in the same way used for scale factors, using the dedicated tables.

1. **Project Scheduling**
   1. **Tasks Identification**

The aim of this paragraph is to highlight the main tasks of the project development. These tasks are just a guideline since the project could be subject of modification or introduction of new requirements and functionalities.

There are the main tasks:

[T1]: Write and deliver Requirements Analysis and Specification Document (RASD)

[T2]: Write and deliver Design Document

[T3]: Write and deliver Integration Test Plan

[T4]: Write and deliver Project Plan

[T5]: Project Implementation

[T6]: Unit Test

[T7]: Integration Test

[T8]: Deliver and test a Beta Release

[T9]: Final Release

In the table below, we present the task interdependencies:

|  |  |  |  |
| --- | --- | --- | --- |
| Task | Start | Deadline | Dependencies |
| T1 | 15/10/2015 | 6/11/2015 | // |
| T2 | 12/11/2015 | 4/12/2015 | T1 |
| T3 | 13/01/2016 | 21/01/2016 | T2 |
| T4 | 21/01/2016 | 2/02/2016 | T2,T1 |
| T5 | 3/02/2016 | 3/05/2016 | T2,T4 |
| T6 | 3/05/2016 | 10/05/2016 | T5,T4 |
| T7 | 10/05/2016 | 20/05/2016 | T5,T6 |
| T8 | 20/05/2016 | 30/05/2016 | T5,T6,T7 |
| T9 | 30/05/2016 | 15/6/2016 | T8 |

* 1. **Tasks Allocation**

1. **Risk Analysis**