My Taxi Service

Project **P**lan

Authors:

Andrea DONATI {andrea4.donati@mail.polimi.it}

Gabriele CARASSALE {gabriele.carassale@mail.polimi.it}

Manuel DELEO {manuel.deleo@mail.polimi.it}

Prof: Elisabetta Di Nitto

Milan, February 2, 2016

Contents

1 Function Points Estimation

1.1 Method Description

The Function Point Analysis is a method used to estimate the complexity of a system. The elements of the system are divided into five classes, and to each component a weight is assigned depending on the relative complexity.
Here it's reported a table defining all the weights used to compute the final result of the Unadjusted Function Points.

Function Type	**Weights**		
	Simple	**Medium**	**Complex**
EI	3	4	6
EO	4	5	7
EQ	3	4	6
ILF	7	10	15
EIF	5	7	10

Table 1: Weights definition

1.2 Internal Logic Files

The Internal Logic Files (ILF) are homogeneous set of data used and managed by the application.

- User
- Taxi Driver
- Ride Request
- Ride Reservation
- Shared Ride
- Taxi Queue
- Administrator

Internal Logic Files	Complexity	Weight
User	Simple	7
Taxi Driver	Simple	7
Ride Request	Medium	10
Ride Reservation	Medium	10
Shared Ride	Medium	10
Taxi Queue	Simple	7
Administrator	Simple	7
Total		**58**

Table 2: ILF and their weights

1.3 External Interface Files

The External Interface Files (EIF) are homogeneous set of data used by the application but generated and maintained by other applications. The system interacts with the two external services reported below.

- Google Maps API
- Payment Service

External Interface Files	Complexity	Weight
Google Maps API	Simple	5
Payment Service	Complex	10
Total		**15**

Table 3: ELF and their weights

1.4 External Inputs

The External Input (EI) is an elementary operation to elaborate data coming from the external environment. Here it's presented a list of all the interactions between the different types of users and the system.

- Login
- Logout
- Registration
- Taxi request

- Taxi reservation
- Taxi sharing
- Taxi availability confirmation
- Taxi Driver call
- Evaluate taxi driver's registration
- Users management by the administrator
- Add new administrator

External Inputs	**Complexity**	**Weight**
Login	Simple	3
Logout	Simple	3
Registration	Simple	3
Taxi Request	Complex	6
Taxi Reservation	Complex	6
Taxi Sharing	Complex	6
Taxi Availability Confirmation	Simple	3
Taxi Driver Call	Simple	3
Evaluate Taxi Driver's Registration	Simple	3
Users Management by the Administrator	Simple	3
Add New Administrator	Simple	3
Total		**42**

Table 4: EI and their weights

1.5 External Inquiries

The External Inquiry (EQ) is an elementary operation that involves input and output. We have not found inquiries in the system. However, it's not relevant because from the point of view of the Function Points computation, the External Inquiries and the External Inputs have exactly the same weights.

1.6 External Outputs

The External Output (EO) is an elementary operation that generates data for the external environment. The only External Output of the system are the notifications sent to the users and the taxi drivers as a confirmation of the requested operations.

- Notifications to users and taxi drivers

External Output	Complexity	Weight
Notifications to users and taxi drivers	Simple	4
Total		**4**

Table 5: EO and their weights

1.7 Unadjusted Function Points Computation

Internal Logic Files	Subtotal
ILF	58
EIF	15
EI	42
EQ	0
EO	4
Total UFP	**119**

Table 6: Total UFP

1.8 SLOC Estimation

We used UFP to estimate the overall number of Source Lines of Code (SLOC) of the entire system. In order to obtain the SLOC estimation we multiplied the UFP value computed in the previous section by a typical factor for J2EE, that's equal to **47**.

$$SLOC = UFP \times 47$$

So, the estimated SLOC are 5593.

2 COCOMO II Estimation

2.1 Scale Drivers

Precedentedness (PREC) Reflects the previous experience of the organization with this type of project. Very low means no previous experience, Extra high means that the organization is completely familiar with this application domain.

Development Flexibility (FLEX) Reflects the degree of flexibility in the development process. Very low means a prescribed process is used; Extra high means that the client only sets general goals.

Architecture/Risk Resolution (RESL) Reflects the extent of risk analysis carried out. Very low means little analysis, Extra high means a complete a thorough risk analysis.

Team Cohesion (TEAM) Reflects how well the development team know each other and work together. Very low means very difficult interactions while extra high means an integrated and effective team with no communication problems.

Process Maturity (PMAT) Reflects the process maturity of the organization. The computation of this value depends on the CMM Maturity Questionnaire but an estimate can be achieved by subtracting the CMM process maturity level from 5.

Scale Driver	**Factor**	**Value**
PREC	Low	4.96
FLEX	High	2.03
RESL	Nominal	4.24
TEAM	Nominal	3.29
PMAT	Nominal	4.68
E	$\mathbf{0.91 + 0.01 \times \sum SF}$	**1.102**

Table 7: Scale Drivers and E parameter

2.2 Cost Drivers

COCOMO II has 17 cost drivers, i.e., multiplicative factors that determine the effort required to complete the software project. They appears as factors in the Effort equation, inside the Effort Adjustment Factor (EAF)

Code	Name	Factor	Value
RELY	Required Software Reliability	High	1.10
DATA	Database Size	Nominal	1.00
CPLX	Product Complexity	Nominal	1.00
RUSE	Required Reusability	Very High	1.15
DOCU	Documentation Match to Life-cycle Needs	Nominal	1.00
TIME	Execution Time Constraints	Nominal	1.00
STOR	Main Storage Constraint	Nominal	1.00
PVOL	Platform Volatility	Low	0.87
ACAP	Analyst Capability	Nominal	1.00
PCAP	Programmer Capability	Nominal	1.00
APEX	Application Experience	Low	1.10
PLEX	Platform Experience	Low	1.09
LTEX	Language and Tool Experience	Low	1.09
PCON	Personnel Continuity	Nominal	1.00
TOOL	Usage of Software Tools	Nominal	1.00
SITE	Multisite Development	Nominal	1.00
SCED	Required Development Schedule	Nominal	1.00
EAF	**EAF = $\prod$ CD**		**1.438**

Table 8: Cost Drivers and EAF factor

The EAF value is given by the product of all the Cost Driver values.

2.3 Effort Estimation

Using the the E and EAF parameters computed in the previous sections, the effort is computed in the following way.
The KSLOC factor is obtained by dividing by 1000 the SLOC value estimated using the UFP.

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

In this way, we obtain an effort equals to **28.18** person-months.

2.4 Duration

Finally we can compute an estimation of the duration of the project, starting from the Effort value obtained in the previous section.

$$Duration = 3.67 \times (Effort)^{0.28+0.2\times(E-0.91)}$$

The duration is **10.63** months.
Now it's possible to compute the estimated number of people needed to develop the project, that is:

$$N = \frac{Effort}{Duration} = \frac{28.18}{10.63} = \mathbf{2.65}$$

So the recommended number of people for the project is **3**, that's exactly the number of people in our group.

3 Project Tasks and Schedule

3.1 Tasks Definition

In this section it's presented the list of the defined task for the project. Each task is related to a specific deliverable of the project or to a component of the system.

Code	Name	Effort	Duration	Dependencies
T1	Requirements Engineering (RASD)	45	15	—
T2	Design (DD)	45	15	T1
T3	Integration Test Plan (ITPD)	15	5	T1, T2
T4	Project Plan	15	5	T1, T2
T5	Mobile App Development	60	60	T1, T2, T4
T6	Web App Development	60	60	T1, T2, T4
T7	Database Development	20	40	T1, T2, T4
T8	Account Manager Development	15	30	T1, T2, T4
T9	Request Manager Development	30	30	T1, T2, T4
T10	Notification Manager Development	15	30	T1, T2, T4
T11	Queue Manager Development	20	40	T1, T2, T4
T12	Integration Testing	30	10	T1 - T10
T13	System Testing	30	10	T1 - T11

Table 9: Tasks

In the task related to the development of a component of the system, it's also considered the relative code inspection and unit testing, that will be performed inside the task.

This is a graphical representation of all the tasks with their dependencies.

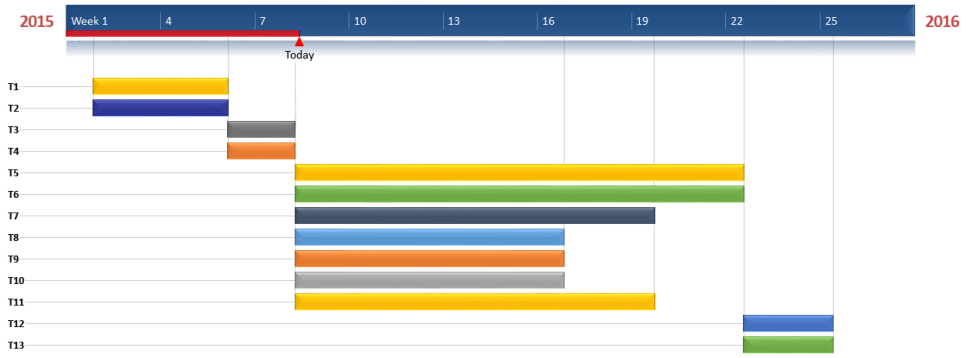

Figure 1: Task diagram

3.2 Allocation

In the first four tasks (T1-T4) we worked together at the same time, in order to produce all the documents needed to start the development of the system.
Then, the tasks T5-T11 have been divided between the three group members and will be developed simultaneously.
After that, we will perform the integration testing phase (T12) together and finally the system test (T13) in order to verify that the system works in the proper way.
The tasks will be allocated to the group members in the following way.

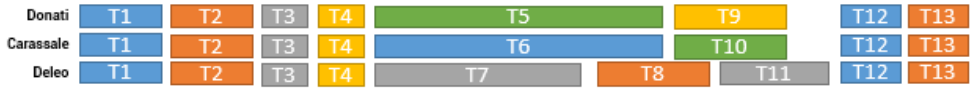

Figure 2: Tasks Allocation

4 Project Risks Analysis

We have identified all possible risks that could affect **myTaxiService**. So we looked for each risk the probability of occurrence and the impact it would have on the project. We also looked for a solution for each risk.

4.1 Project Risks

Description	**Probability**	**Effects**	**Recovery Strategy**
Product size	Low	Moderate	Hire more staff
Staff experience	Low	Moderate	Hire more experienced staff
Staff size	Low	Moderate	Hire more staff
New laws	Medium	Moderate	Adapt project plan to new legislation

Table 10: Project Risks Analysis

4.2 Technical Risks

Description	**Probability**	**Effects**	**Recovery Strategy**
New technologies	Moderate	Low	Consider to use only new technologies that could really boost the project

Table 11: Technical Risks Analysis

4.3 Business Risks

Description	Probability	Effects	Recovery Strategy
Market risk	Low	Critical	Keep analyzing the market so that the project could be adapted to its needs
Competitors	Medium	Moderate	Keep developing new ideas and features
Shortage of money	Low	Critical	Try to avoid big expense

Table 12: Business Risks Analysis

5 Times spent working on this document

Group Member	**Total Hours**
Andrea Donati	10
Gabriele Carassale	7
Manuel Deleo	8