Politecnico di Milano
A.A. 2015-2016
Software Engineering 2

Project Plan Document

Alessandro Macchi
Caterina Finetti
Simone Manzoli

Summary

1. Introduction

In this document the time and resources necessary to the development of the MyTaxiApp are evaluated. For the analysis the Functional Point approach has been used, and the results have been compared with the dimension of the project. Furthermore, the COCOMO model has been used to evaluate the effort for MyTaxiApp implementation and the results have been compared whit the time actually spent.

2. Function Point Approach

2.1 Function Point Overview

The Functional Point approach is a technique that allows to evaluate the effort needed for the design and implementation of a project. We have used this technique to evaluate the application dimension basing on the funcionalities of the aplication itself. The funcionalities list has been obtained from the RASD documet

and for each one of them the realization complessity has been evaluated.

The functionalities has been groped in:

- **Internal Logic File:** it represents a set of homogeneous data handled by the system. In our application this corresponds to the database table.
- **External Interface File:** it represents a set of homogeneous data used by the application but handled by external application. In our application it corresponds to the interaction with the taximeter and the GPS.
- **External Input:** elementary operation that allows input of data in the system.
- **External Output:** elementary operation that creates a bit stream towards the outside of the application.
- **External Inquiry:** elementary operation that involves input and output operations without any kind of data manipulation from the application.

The following table outline the number of Functional Point based on functionality and relative complexity

Funcion Type	Complexity		
	Simple	Medium	Complex
ILF	7	10	15
EIF	5	7	10
External Input	3	4	6
External Output	4	5	7
External Inquery	3	4	6

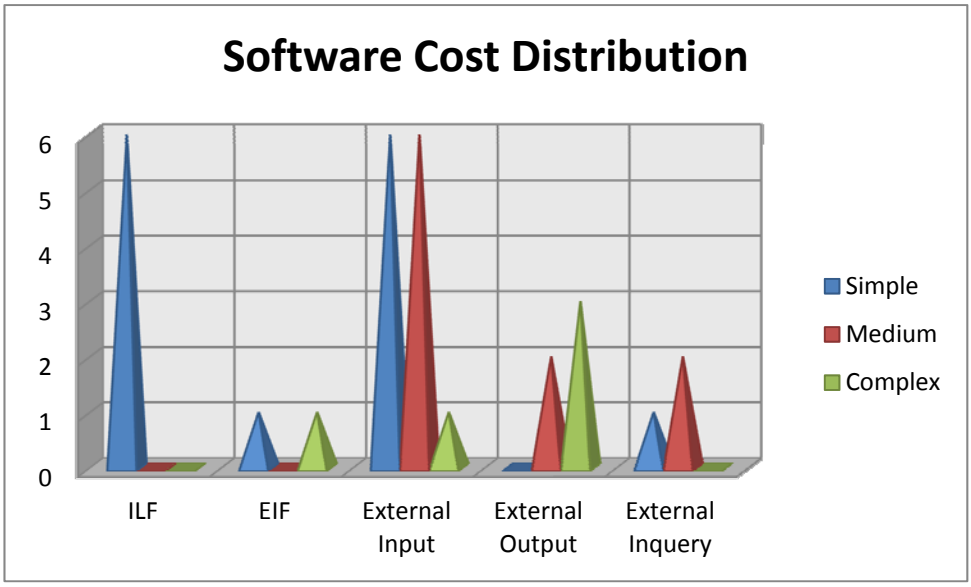
2.2 Application of the Function Point method

Using the Function Points method, the estimation of the effort required to complete the application depends on the functionalities the application has to offer. Specifically, the number of FPs can be computed as the weighted sum of function types using the coefficient of the table above. We perform the calculation step by step:

- **Internal Logic Files (ILF):** the application include 6 simple ILF one for each table of our database (see the Design Document for further information). We choose to give them the lightest weight because they all have few fields. So we got 6*7=42 FPs
- **External Interface File (EIF):** We have 2 EIF the taximeter, considered simple since it requires only one read every call and the GPS data that are Complex since we must have a continuous update of those files. 1*5+1*10=15 Fps
- **External Inputs:** The application have many interaction with the users we try to sum they below:
    - *Login/logout*: these are simple operation. 2*3=6
    - *Create a new Taxi driver/user:* these are also simple operation because they involve only one entity each. 2*3=6
    - *Insert a new favorite location:* As said before this also involve only one entity so it's simple. 1*3=3
    - *Create a new call:* this is a quite complex operation because involve every entity of our system. 1*6=6
    - *Accept/refuse a call by the taxi driver/user:* these are medium operation that modify the call, queue and users entities. 4*4=16
    - *Signal a blank call and drop off a user:* These are medium operations because involves call, queue, and users entities. 2*4=8
    - *Change taxi driver status:* this is a simple operation because involves only the queue entity and don't require any computations. 1*3=3
- **External Output:** The application notifies to the users information about the operations they perform, and to the taxi drivers the calls they receive.
    - After a call has been created, application notify to the taxi driver the incoming call. This operation involves X entities: User, Taxi Driver, Address, Call. We can consider it a complex operation so we can adopt complex cost 1*7=7.
    - After a call has been accepted (or refused) by a taxy driver, the system notify to the user that he/she will be picked up as soon as possible. This operation involves 3 entities: user, taxi driver, call, so we consider it as an operation with medium cost. 1*5=5

    - If a delayed call has been created before, 10 minutes before the pick-up time the application notify it to a taxi driver. This operation involves 4 entities: User, taxi driver, Address, Call. So we consider it an operation with a complex cost. 1*7=7
    - If a user is taking a shared ride, he receives a notification that inform him/her of the cost of is part of the ride. This is a quite complex operation, that involves 4 entities (User, Taxi driver, Address, Call) and also an EIF that will perform the calculation. Is probably one of the most complex operations. 1*7=7
    - If a user is reported for a blank call, the application send him a notification after the login. This is a simple operation, that involves 3 entities: user, taxi driver and call. We consider it an operation with a medium cost. 1*5=5
- **External Inquiry:** The application allows the users to see their information and location:
    - *See user's information:* this operation involves only the user table so it's simple. 1*3=3
    - *Se user's favorite locations:* this is a simple operation but involves two entities so we give it a medium cost. 1*4=4
    - *Visualize Taxi information:* this is a simple operation but involves two entities so we consider it a medium weight operation. 1*4=4**.**

SW Cost		Simple		Medium		Complex	Total
ILF	6	7	0	10	0	15	42
EIF	1	5	0	7	1	10	15
External Input	6	3	6	4	1	6	48
External Output	0	4	2	5	3	7	31
External Inquery	1	3	2	4	0	6	11
							147



3. COCOMO Approach

To pass from FP to SLOC we use an average conversion factor of 46 as described at http://www.qsm.com/resources/function-point-languages-table, an updated version that adds J2EE of the table included in official manual (http://sunset.usc.edu/research/COCOMOII/Docs/modelman.pdf).

$$147\ FPs * 46 = 6762\ SLOC$$

A first estimation is based on FPs approach converted to SLOC. We Consider a project with all “Nominal” Cost Drivers and Scale Drivers would have an EAF of 1.00 and exponent E of 1.0997. Following this formula

$$effort = 2.94 * EAF * (K\ SLOC)^{E}$$

we obtain

$$effort = 2.94 * 1.00 * (6.762)^{1.0997} = 24.05\ Person/Month$$

EAF: Effort Adjustment Factor derived from Cost Drivers.

E:Exponent derived from Scale Drivers.

Now we try to calculate the schedule (duration) of project in month with the following formula

$$Duration = 3.67 * (effort)^{E}$$

We consider an exponent E of 0.3179

$$Duration = 3.67 * (24.05)^{0.3179} = 10.09\ Months$$

Now we can estimate the number of people needed to complete the project with the following formula

$$N_{people} = effort/Duration$$

$$N_{people} = 24.05/10.09 = 2.38 \rightarrow 3\ people$$

We want to give a more precise estimation adjusting some Scale Driver. To evaluate the COCOMO II and determine the effort required to complete the software project we also use an online tool that helps us to do some calculus (http://csse.usc.edu/tools/COCOMOII.php). We add the report of that site and the choice made about the Scale Driver to obtain that result.

COCOMO II - Constructive Cost Model

Model(s): COCOMO
Monte Carlo Risk: Off
Auto Calculate: Off

Software Size Sizing Method: Source Lines of Code

	SLOC	% Design Modified	% Code Modified	% Integration Required	Assessment and Assimilation (0% - 8%)	Software Understanding (0% - 50%)	Unfamiliarity (0-1)
New	6762						
Reused		0	0				
Modified							

Software Scale Drivers

Precedentedness	Low	Architecture / Risk Resolution	High	Process Maturity	Nominal
Development Flexibility	Nominal	Team Cohesion	Very High		

Software Cost Drivers

Product		Personnel		Platform	
Required Software Reliability	Low	Analyst Capability	Nominal	Time Constraint	Nominal
Data Base Size	Nominal	Programmer Capability	High	Storage Constraint	High
Product Complexity	Low	Personnel Continuity	Nominal	Platform Volatility	Low
Developed for Reusability	High	Application Experience	Nominal	**Project**	
Documentation Match to Lifecycle Needs	High	Platform Experience	Low	Use of Software Tools	Low
		Language and Toolset Experience	Low	Multisite Development	Nominal
				Required Development Schedule	Nominal

Maintenance Off

Software Labor Rates

Cost per Person-Month (Dollars): 2000

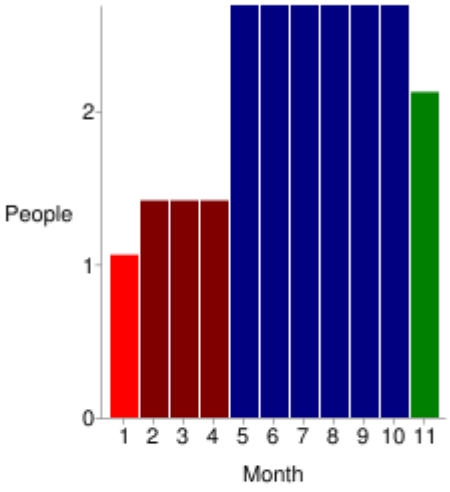
Results

Software Development (Elaboration and Construction)

Effort = 22.8 Person-months
Schedule = 10.3 Months
Cost = $45509

Total Equivalent Size = 6762 SLOC

Staffing Profile



Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.4	1.3	1.1	$2731
Elaboration	5.5	3.9	1.4	$10922
Construction	17.3	6.4	2.7	$34588
Transition	2.7	1.3	2.1	$5461

Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.2	0.7	1.7	0.4
Environment/CM	0.1	0.4	0.9	0.1
Requirements	0.5	1.0	1.4	0.1
Design	0.3	2.0	2.8	0.1
Implementation	0.1	0.7	5.9	0.5
Assessment	0.1	0.5	4.2	0.7
Deployment	0.0	0.2	0.5	0.8

4 Conclusion

Here are listed the real hours of works, included in Development Time Document, spent for MyTaxiApp project.

- **Requirements Analysis and Specifications Document:**
  - Alessandro Macchi: ~18h
  - Caterina Finetti: ~16h
  - Simone Manzoli:~22h
- **Design Document:**
  - Alessandro Macchi: ~13h
  - Caterina Finetti: ~16h
  - Simone Manzoli:~14h
- **Code Inspection:**
  - Alessandro Macchi: ~5h
  - Caterina Finetti: ~4h
  - Simone Manzoli:~4h
- **Integration Test Plan:**
  - Alessandro Macchi: ~2h
  - Caterina Finetti: ~2h
  - Simone Manzoli:~1h
- **Final Presentation:**
  - Alessandro Macchi: ~2h
  - Caterina Finetti: ~3h
  - Simone Manzoli:~3h

The total hours of work spent during all phases of the project are 125 hours.

*125 hours/(40 * 4) hours = 0.78 Person/Month*

We suppose that a man can work 40 hours in a week so 40*4 is a number of hours that man works in a month.

We suppose that we can implement the myTaxiApp in 375 hours so 2.34 Person/Month.

In conclusion the total Person/Month required to complete the application is 3.12 that is very little time compared with the theoretical COCOMO II estimation (24.05 Person/Month).