TAXI SERVICE

PP

Project Plan

version 1.1

Rota Diego, 841344, 6 hours

Montalto Simone, 841359, 6 hours

Politecnico di Milano, A.A. 2015-2016

Software Engineering 2 – Prof.ssa Mirandola Raffaela

Table of contents

1. Introduction

In this document are evaluated the time and the resource necessary for the Taxi Service System. To accomplish this, Functional Point analysis and COCOMO are used. It is also listed the number of hours spent by each resource to accomplish each task and the possible risk with the relative solution.

2. Function Points

The Functional Point approach is used to evaluate what is necessary in order to accomplish the design and the implementation of a project. The functionalities of the application are used to make this evaluation. The RASD document is used to extract all the functionalities that will be used in this document.

In the following table are listed the number assigned to the Functional Point. The numbers are calculated with respect to the functionality and their complexity.

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

2.1 Internal Logic Files

Taxi Service has some Internal Logic Files that are used to store different kind of information, useful in order to manage all the request of the users and the response of the drivers. To work, Taxi Service needs of all the users that are registered to the system. For each user, the system must to save all the request (immediate or booked) that he or she accomplish. Also the zones must be accessible. In the table below there are all the Internal Logic Files of the system.

ILF	**Complexity**	**FP**
User	Simple	7
Request (Immediate and Booked)	Medium	10
Zone	Complex	15
Total		32

2.2 External Logic Files

Taxi Service has to manage external service based on Google Maps API. The data obtained are returned in JSON format. They must be integrated in the mobile app. Maps are used both in Request and Zone management.

ELF	Complexity	FP
Maps	Simple	5
Total		5

2.3 External Inputs

In this section are listed all the interaction of the users with the system in order to accomplish all the action that the system offer.

EI	Complexity	FP
Login/Logout/Register	Simple	3x3
Create/Delete User Request	Medium	2x4
Driver Availability	Simple	1x3
Driver Accept/Decline Request	Complex	1x6
Total		26

2.4 External Inquiries

Taxi Service allows to the users to access their personal page and, in particular, to the external developer to access and request the API of the system.

EQ	Complexity	FP
User Personal Page	Simple	1x3
Request History	Simple	1x3
External Developer API	Medium	1x4
Total		10

2.5 External Outputs

Taxi Service sends notification to the Customers and Drivers when some action are performed.

EO	Complexity	FP
Notification to Customer for Booked Request	Simple	1x4
Notification to Driver for a new request	Simple	1x4
Total		8

2.6 Resume

In order to have a complete vision of all the functional points, in the table proposed below are listed all the internal and external points with the relative Function Point value.

Function Type	Value
Internal Logic Files	32

External Logic Files	5
External Inputs	26
External Inquiries	10
External Outputs	8
Total	81

2.7 Final Evaluation

Using the total value of the Functional Point, we can estimate the total number of lines of code (SLOC). To do this estimation, we use as AVC value (language-dependent factor) for J2EE 46, according to this webpage:
http://csse.usc.edu/csse/research/COCOMOII/cocomo2000.0/CII_modelman2000.0.pdf

$$SLOC = AVC * FP = 46 * 81 = 3726$$

3. COCOMO Approach

COCOMO compute software development effort and cost as a function of program size. Program size is expressed in estimated thousand of lines of code SLOC calculated in section [1.7].

The first value to calculate is the effort.

$$\text{effort} = 2.94 * \text{EAF} * (\text{KSLOC})^{\text{E}}$$

with EAF Effort Adjustment Factor derived from cost drivers and E Exponent derived from Scale Drivers. We'll use a value of 1.0 for EAF and a value of 1.0997 for E.

$$\text{effort} = 2.94 * \text{EAF} * (\text{SLOC})^{\text{E}} = 2.94 * 1.0 * (3.726)^{1.0997} = 12.48944366$$

The second value to calculate is the number of month necessary to complete the entire project. To do it we use this formula:

$$\text{Duration} = 3.67 * (\text{effort})^{\text{SE}}$$

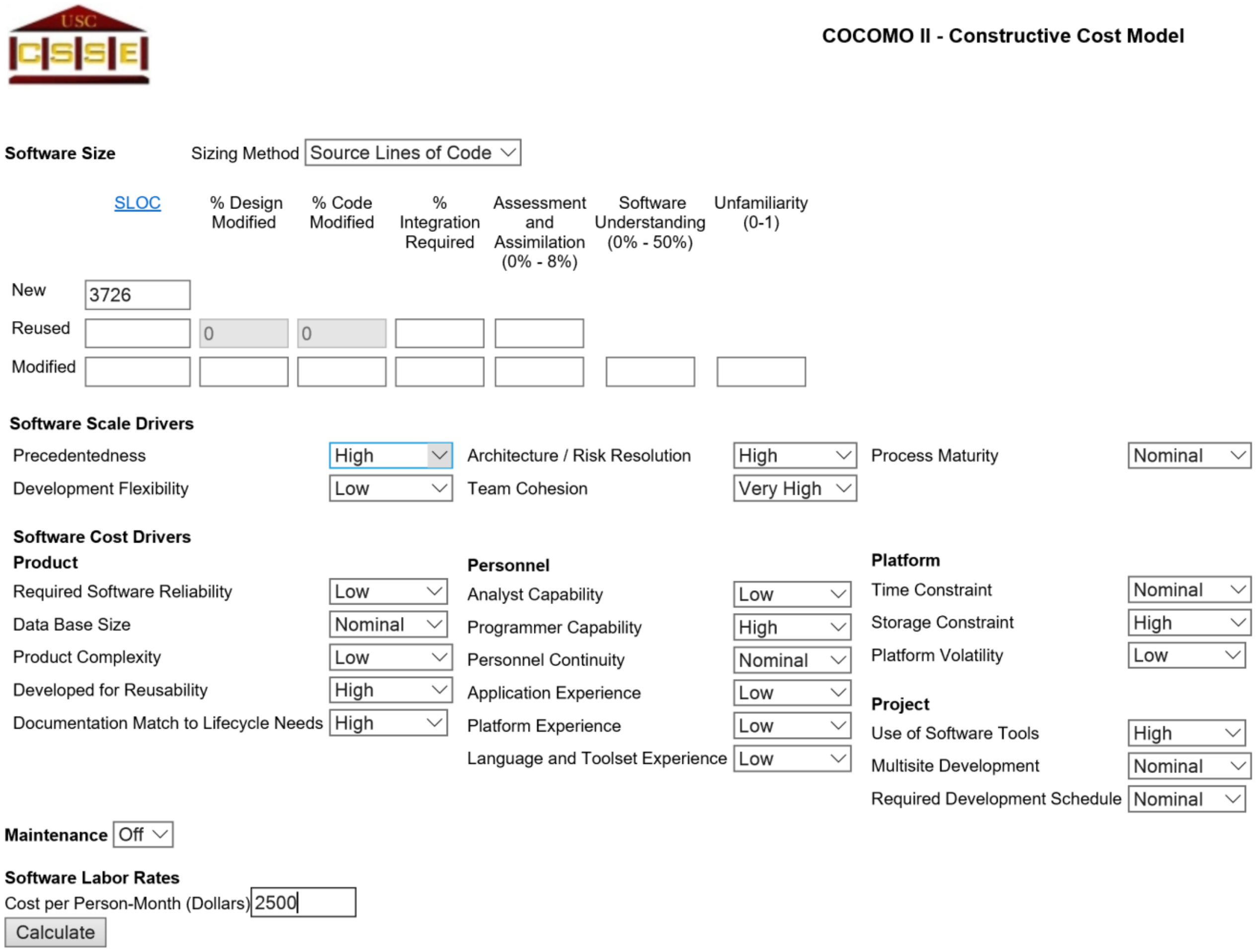
with effort the value calculated before and SE, that is the schedule equation exponent derived from the five scale drivers, equal to 0.3179.

$$\text{Duration} = 3.67 * (\text{effort})^{\text{SE}} = 3.67 * (12.48944366)^{0.3179} = 8.233135578$$

From the effort and the duration is possible to calculate the number of person that must work on the project.

$$\text{N}_{\text{people}} = \text{effort} / \text{Duration} = 12.48944366 / 8.233135578 = 1.5169 = 2 \text{ people}$$

We have also used a tool to evaluate the COCOMO II more precisely. This is the link where it is possible to found the tool: http://csse.usc.edu/tools/COCOMOII.php
Below there are two screenshot to show the results provided by the tool.



COCOMO II - Constructive Cost Model

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	3726						
Reused		0	0				
Modified							

Software Scale Drivers

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

Software Cost Drivers

Product

Required Software Reliability	Low
Data Base Size	Nominal
Product Complexity	Low
Developed for Reusability	High
Documentation Match to Lifecycle Needs	High

Personnel

Analyst Capability	Low
Programmer Capability	High
Personnel Continuity	Nominal
Application Experience	Low
Platform Experience	Low
Language and Toolset Experience	Low

Platform

Time Constraint	Nominal
Storage Constraint	High
Platform Volatility	Low

Project

Use of Software Tools	High
Multisite Development	Nominal
Required Development Schedule	Nominal

Maintenance Off

Software Labor Rates

Cost per Person-Month (Dollars) 2500

Results

Software Development (Elaboration and Construction)

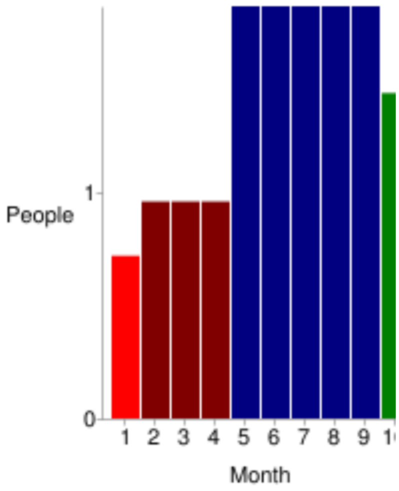
Effort = 12.7 Person-months
Schedule = 8.5 Months
Cost = $31757

Total Equivalent Size = 3726 SLOC

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	0.8	1.1	0.7	$1905
Elaboration	3.0	3.2	1.0	$7622
Construction	9.7	5.3	1.8	$24136
Transition	1.5	1.1	1.4	$3811

Staffing Profile



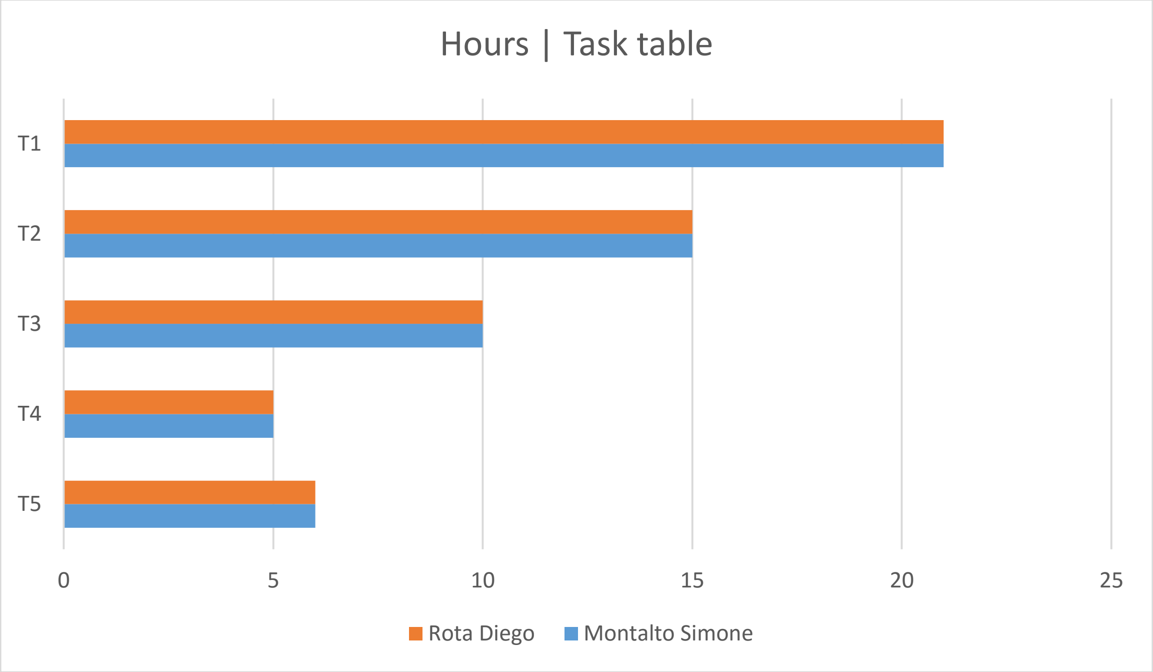
Software Effort Distribution for RUP/MBASE (Person-Months)

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.1	0.4	1.0	0.2
Environment/CM	0.1	0.2	0.5	0.1
Requirements	0.3	0.5	0.8	0.1
Design	0.1	1.1	1.5	0.1
Implementation	0.1	0.4	3.3	0.3
Assessment	0.1	0.3	2.3	0.4
Deployment	0.0	0.1	0.3	0.5

4. Allocation Resources

In this section there are some tables that show on which tasks each resource has worked during the project. It is listed also the number of hours spent for each task.

T1: RASD
T2: DD
T3: Code inspection
T4: Integration test
T5: Project Plan



The total amount of hours spent for the project by each member is 57h. We estimate that, for the implementation task, we could spent more or less 120h for each member, for a total amount of time of 240h. This lead to a total of 177h for each member and 354h in total. Taking into account that each member could work 20h per week, this lead to a duration of the project of 8.85 weeks per member (the equivalent of 2.21 month per member).

5. Risk Management

In the table proposed below are defined the risk for the project and the associated recovery actions.

Risk	Probability	Effects	Resolution Strategy
Key staff are ill and is absent for a lot of time.	Medium	Serious	Reorganization of the team tasks.
Change in External API	Low	Serious	Reorganization of the team. Some people must update the code where the API are used.
Requirements Changes	Medium	Serious	High-level abstraction design to minimize the impact between different modules of the system.
Database Performance (database cannot process all the transactions)	Medium	Very Serious	Buy an high-scalable database infrastructure.