MeteoCal Project

Project Reporting

Author:

Francesco Angelo {francesco.angelo@mail.polimi.it}

Valentina Ceriani {valentina.ceriani@mail.polimi.it}

Umberto Di Fabrizio {umberto.difabrizio@mail.polimi.it}

Prof: Elisabetta Di Nitto

Cost Estimation

1.1 Function Point

We have evaluated the requested effort to develop the software with the Function Point method.
This is based on the the functionalities of the application. The total is calculated as the sum of the function points of each functionality, weighted with its complexity
These are the Weighting function points:

Tipi di Funzioni	Complessità		
	Simple	Medium	Complex
Inputs	3	4	6
Outputs	4	5	7
Inquiry	3	4	6
Internal Logic Files	7	10	15
External Logic Files	5	7	10

Inputs:

The Inputs with a simple complexity are the logging in, signing up, editing account information, editing calendar privacy and adding a new calendar. The inputs that have a medium complexity are edit, create and delete an event and delete a user. Delete and import a calendar are considered the most complex functionalities.

FP: 3 * 5 + 4 * 4 + 6 * 2 = 43

Outputs:

The outputs that we have are the Notification (Email etc) and the Export of a calendar.
We have consider the two functionalities as medium complex.

FP: 5 * 2 = 10

Inquiry

We have two medium complex inquiry for retrieve and display informations: searching a user or an event and show a calendar.

FP: 4 * 2 = 8

Internal Logical Files

We have divided all the ILFs by number of fields and degree of managing difficulty. As simple ILFs: User, Invitation, Notification, Public Joins. As medium ILF : Event. As complex ILFs is the Calendar

FP: 7 * 4 + 10 * 1 + 15 * 1 = 53

External Logical Files

We have identified as simple ELFs the Google Geolocation (present in the entity Event) and a complex ELFs the OpenWeatherMap forecast.

FP: 5 * 1 + 10 * 1 = 15

TOTAL FP: 43 + 12 + 8 + 53 + 10 = 125

The conversion for an high language software can be carry out with the formula: LoC = FP * 53
(Where 53 is the average number of LoC for the Java Language)
And in our case:

LoC = 125 * 53 = 6625

Finally we have analyzed our project with a counter of lines of code to have a general idea on the estimate.

Analysis with Simple Code Metrics (netbeans plugin) [link]

```
------------------
LOC
------------------
Total LOC: 12422

Classes LOC:
ForecastWeatherData: 880
ManageEventBacking: 865
DailyForecastData: 800
AbstractWeatherData: 761
CurrentWeatherData: 724

Packages LOC:
java: 12422
main: 12422
src: 12422
weatherLib: 3904
bakingBeans: 3239
------------------
Lines with imports
------------------
Total imports: 678
------------------
Blank lines
------------------
Total blank lines: 1811
------------------
```

The number of lines of code that we have effectively written are:

Total	12422 -
Lines of an open source library	3904 -
Blank lines	1811 -
Lines with imports	678 =
Net LOC	6029

This number of lines is quite comparable to the numbers of lines of code evaluated with the FP:
6625 vs 6029.

1.2 COCOMO 1

S: Project Dimension
M: Effort in men months
T: Time to develop in months
N: Number of persons in the team

Type of application	a	b	c	d
Organic Mode	2.4	1.05	2.5	0.38
Semi-Detached Mode	3	1.12	2.5	0.35
Embedded Mode	3.6	1.2	2.5	0.32

We considered the complexity of the application of Semi-Detached level, because there are largely used Primefaces components and everything else is build from zero.
Going on with the estimate started with the FP, we can evaluate the total time to develop and the number of involved people.

$a = 3\ b = 1,12\ c = 2,5\ d = 0,35$

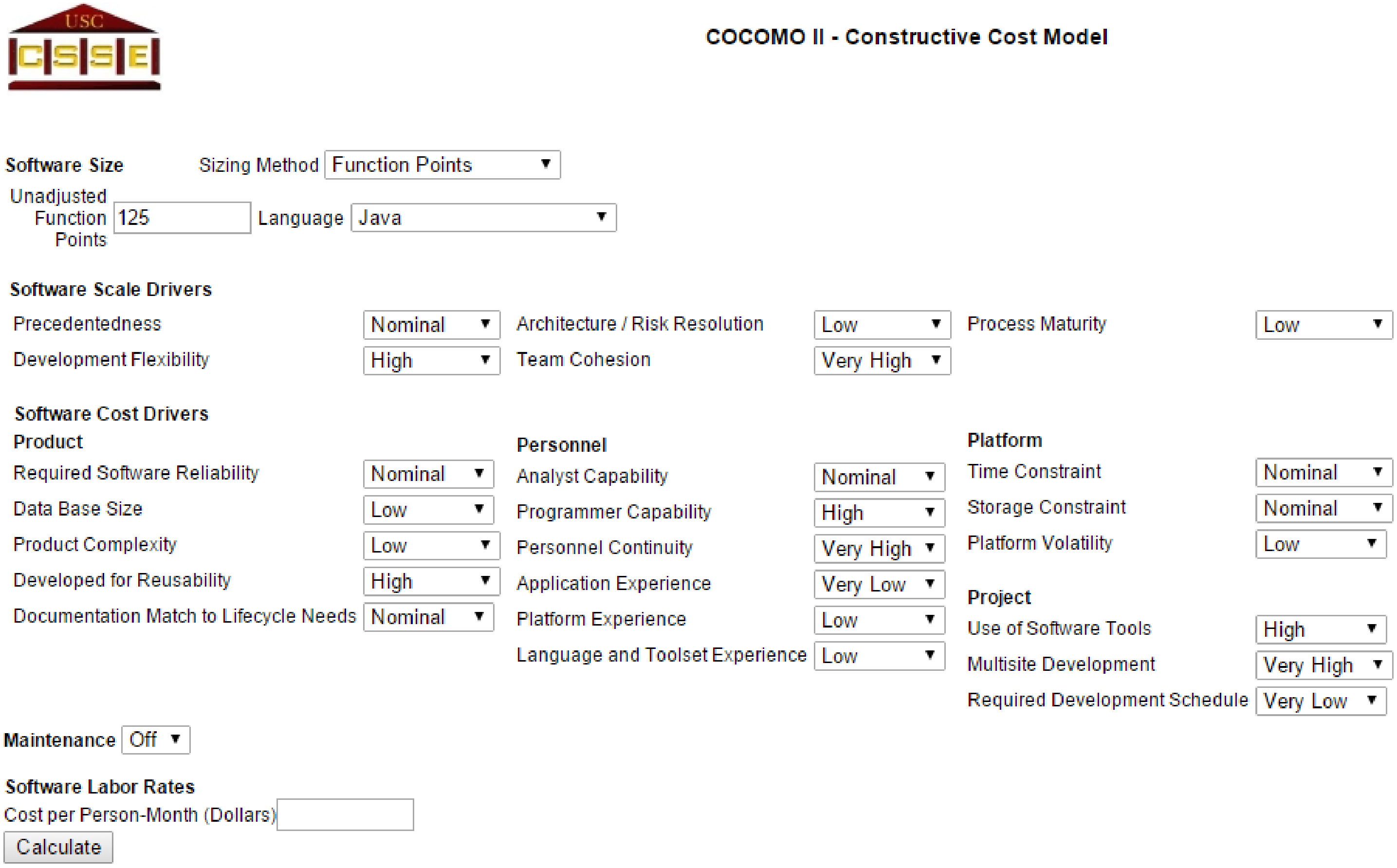
$$M = 3 * (6.62^{1.12}) = 24.91$$

$$T = 2.5 * (6.62^{0.35}) = 4.82$$

$$N = \frac{M}{T} = \frac{24.91}{4.82} = 5.16$$

1.3 Cocomo 2

We used the tool find here to better analyze the cost estimation with COCOMO II algorithm. Scale Drivers and Cost Drivers has been chosen as in *Fig.1*.
Results are shown in *Fig.2.*



COCOMO II - Constructive Cost Model

Software Size Sizing Method: Function Points

Unadjusted Function Points: 125 Language: Java

Software Scale Drivers

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

Software Cost Drivers

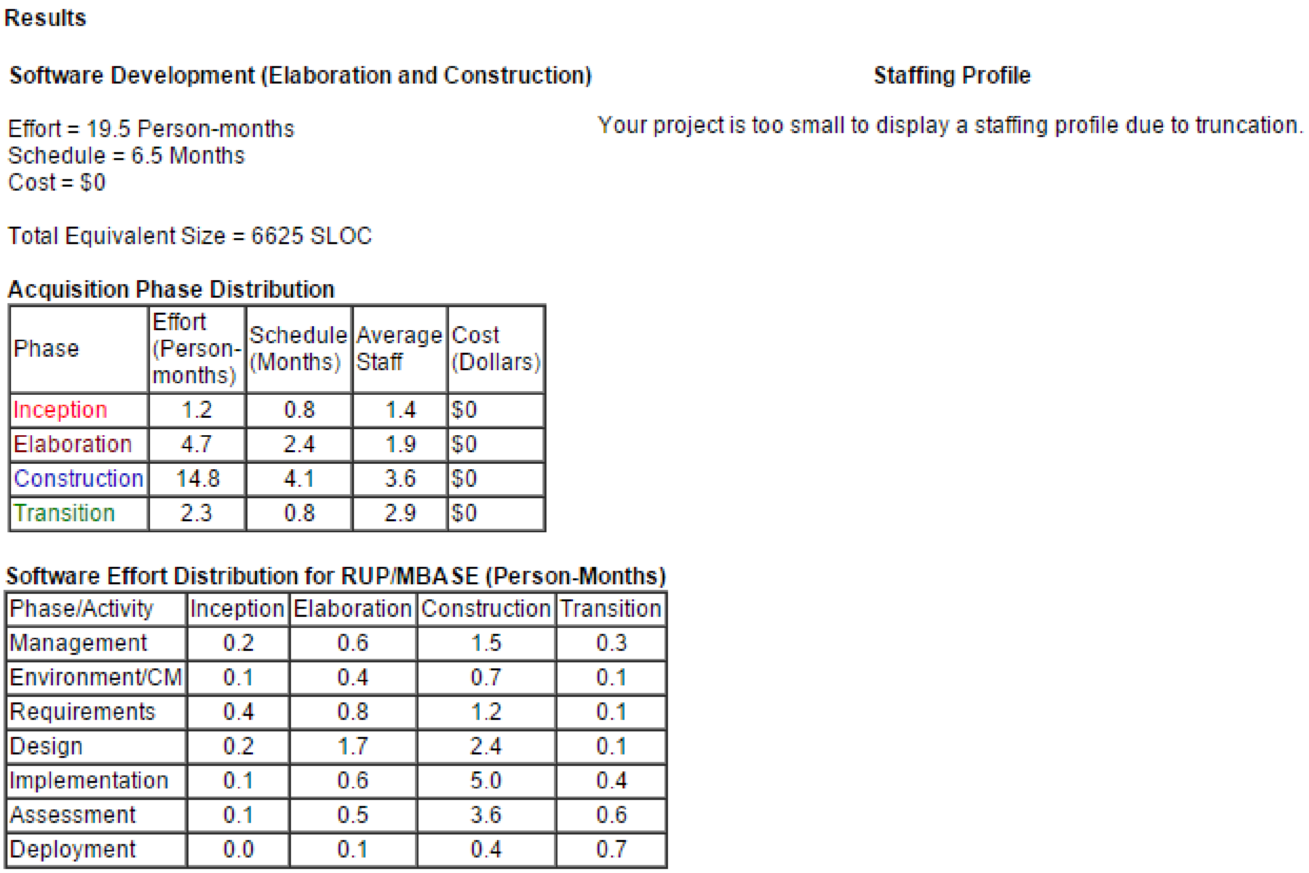
Product		**Personnel**		**Platform**	
Required Software Reliability	Nominal	Analyst Capability	Nominal	Time Constraint	Nominal
Data Base Size	Low	Programmer Capability	High	Storage Constraint	Nominal
Product Complexity	Low	Personnel Continuity	Very High	Platform Volatility	Low
Developed for Reusability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

Maintenance Off

Software Labor Rates

Cost per Person-Month (Dollars)

Fig.1 : Choosen Scale & Cost Drivers



Results

Software Development (Elaboration and Construction)

Effort = 19.5 Person-months
Schedule = 6.5 Months
Cost = $0

Total Equivalent Size = 6625 SLOC

Staffing Profile

Your project is too small to display a staffing profile due to truncation.

Acquisition Phase Distribution

Phase	Effort (Person-months)	Schedule (Months)	Average Staff	Cost (Dollars)
Inception	1.2	0.8	1.4	$0
Elaboration	4.7	2.4	1.9	$0
Construction	14.8	4.1	3.6	$0
Transition	2.3	0.8	2.9	$0

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

Phase/Activity	Inception	Elaboration	Construction	Transition
Management	0.2	0.6	1.5	0.3
Environment/CM	0.1	0.4	0.7	0.1
Requirements	0.4	0.8	1.2	0.1
Design	0.2	1.7	2.4	0.1
Implementation	0.1	0.6	5.0	0.4
Assessment	0.1	0.5	3.6	0.6
Deployment	0.0	0.1	0.4	0.7

Fig.2: Results

$$N_{people} = \frac{Effort}{Duration} = \frac{19.5}{6.5} = 3$$

1.4 Final Considerations

The estimated LOC using FP is slightly overestimated, but very close to the actual LOC.

The Cocomo I analysis forecast a man-month of 24.91 which is too high compared to the time we actually spent for this reason we deepen the study with the Cocomo II.
This one forecast both a value of the man-month and the group number which is closer to our situation.The number N of Team Members, using Cocomo II is the same as the real value.

Anyway the values of the Cocomo II are still higher than the real values because the analysis is used to evaluate a Company Process with fixed parameters (e.g. time,budget) but more available resources (e.g. people) and not a university project, moreover we did work much more than a standard employee in a company (w.r.t. time per day) since we produced code even during weekends and nights.

The overall estimation could have been nonetheless used as a valid support during the project planning.

Final Overview of Working Hours

Documentation - Working Hours			
	Francesco Angelo	Valentina Ceriani	Umberto Di Fabrizio
RASD	43	45	41
DD	25	25	26
Acceptance Test	5	5	5
Project Reporting	3	3	3
Implementation	184	168	176
Tot	260	244	249