MeteoCal

Software Enginnering 2 Project

RASD

D. Pensa
M. Pini
A. Pintus

Contents

1 Function Points

Using the method of Function Points, the effort required to complete the application depends on the functionalities of the application itself.
The total Function Points is calculated from the weighted sum of the
different application functions.

Functions	Complexity		
	Simple	Medium	High
Inputs	3	4	6
Outputs	4	5	7
Inquiry	3	4	6
Internal Files	7	10	15
External Files	5	7	10

1.1 Internal Logic Files(ILF)

The different ILF in the application can be divided by complexity: the first, simpler since they contain fewer fields, are the *Contact*, *Calendar* and *Token*; the remaining have medium complexity as they contain more attributes: *User*, *Update* and *Weather Condition*. Only the *Event* can be considered as having high complexity.
FP : 3*7 + 3*4 + 1*7 = 40

1.2 External Logic Files(ELF)

The fundamental ELF is the one related to th *weather forecasts*, retrieved through Yahoo APIs. Its management is very complex! Instead, retrieving data from Google Maps to get a precise *location* for the event is more and more simple.
FP : 1*10 + 1*5 = 15

1.3 Output

As output, the application must send *email* notification to the registered users; it has been used as much as possible, to keep users always up-to-date, but its implementation is very simple. Using email to perform password recovery is a medium complexity operation; instead, the *export* operation of the calendar, is very complex and it provides 4 different file formats
FP : 1*4 + 1*5 + 1*7 = 16

1.4 Input

The four simplest inputs are *login*, *logout*, *registration* and *un-registration*; other simple inputs are adding a *contact*, *change password* and *privacy*. The only two input operation that have medium complexity are creation/modification of an event. In the end, there is also the possibility to *import* a calendar previously exported from the application and it has an high complexity.
FP : 7*3 + 2*4 + 1*6 = 35

1.5 External Inquiry

The inquiries to get and display data are medium complexity operations in general; they are performed on *events*, *contacts*, *users*, *updates*, *weather conditions*.
FP : 5*4 = 20

TOTAL FP : 40 + 15 + 16 + 35 + 20 = 126

Conversion from FP to lines of high-level language code is calculated with the formula

LOC = FP*53

In this case **LOC = 126*53 = 6678**

2 COCOMO

Let's calculate the effort of the application using the method of *COCOMO*.

S : Application dimension
M : Effort (Month/Person) M = a*S^b
T : Development time (Months) T = c*S^d
N : Number of people needed for the project

The coefficients a,b,c,d are given by this table

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

The application can be considered as Semi-Detached level, as it have been used *Primefaces* components whereas the rest of the application has been deployed from scratch.
Thanks to the conversion of Function Points made earlier, we can approximate the number of lines of code of the application to 6,7 KLOC.
Using this data and the coefficients in the table, we can calculate the effort, the total time of development and the number of people involved.

$$a = 3, b = 1.12, c = 2.5, d = 0.35 \tag{1}$$

$$M = 3 * 6.7^{1.12} = 25.253 \tag{2}$$

$$T = 2.5 * 6.7^{0.35} = 4.865 \tag{3}$$

$$N = M/T = 2 = 5.191 \tag{4}$$

In conclusion, the estimates based on Function Points and *Cocomo* have proved fairly accurate, except for the estimate of the number of people involved, in fact *Cocomo* indicates a number of people involved in the application approximately equal to 5, while in reality the program was developed by a group of three elements. About the lines of code calculated with Function Points instead we got a fairly accurate result: 6678 lines estimated against 6177.

3 Final Working Hours Schedule

TASK	Pensa	Pini	Pintus	h/sect
Problem Analysis	2	2	2	6
Non Functional Requirements	6	4	5	15
Functional Requirements	6	4	3	13
Alloy Specifications	4	9	8	21
Tot Requirements	18	19	18	55
General Description	1	1	1	3
Data Planning	2	3	10	15
UX Model	9	9	4	22
BCE Model	9	5	3	17
Sequence Diagrams	3	3	3	9
RASD Adjustment	0	1	1	2
Tot Design	24	22	22	68
Setup Tools	4	5	5	14
Data Layer Implementation	38	35	50	123
Business Logic Implementation	52	72	57	181
Web Tier Implementation	65	54	53	172
User Interface Implementation	7	5	12	24
Unit and Integration Testing	13	24	10	47
Setup, User Manual	2	2	0	4
Tot Implementation	181	197	187	565
Test Cases Identification	1	1	1	3
Tests Execution	1	4	3	8
Documentation	0	1	2	3
Tot Acceptance Testing	2	6	6	14
TOT PROJECT	225	244	233	702