1. Introduction

1.1. Description of the used techniques:

The following Project Reporting document aims to analyze the overall effort made in the design phase through the techniques of Unadjusted Functional Points and COCOMO.

Function Points: In this first analysis we propose to consider the time and resources spent on the basis of the features of interest of which were realized. It is a comprehensive assessment that wants to move away from the actual implementation of the program, not based for example on the lines of application code or the memory used, but placing itself in a more understandable from the point of view of the customer, however, that accurately reflects the overall efforts that would later be made in the course of implementation.

COCOMO: A second analysis using COCOMO (Constructive Cost Model) is used to refine the estimate using Function Point through a rigorous technique and with precise rules but at the same time retaining the generality necessary to make the document intelligible and easy to understand. A proper use of COCOMO assumes a style of development of type Waterfall (which we adopt in progress) and the generation of appropriate documentation to the project during construction. In this part we will consider the phases of Design, Implementation and Testing.

1.2. Using the document:

At the end of this analysis will be possible to compare the estimated effort using these techniques and the actual implementation on the basis of the theory studied in the course of Software Engineering 2, for a detailed description of the steps the reader is referred to the documentation specification generated previously.

2. Effort analysis through Function Point

2.1 Notation used:

The analysis presented in this chapter is based on a very precise notation that describes the various functions of the program by dividing them into five groups fixed. Below is a description of them to make it easier to understand the analysis itself.

Types of functions:

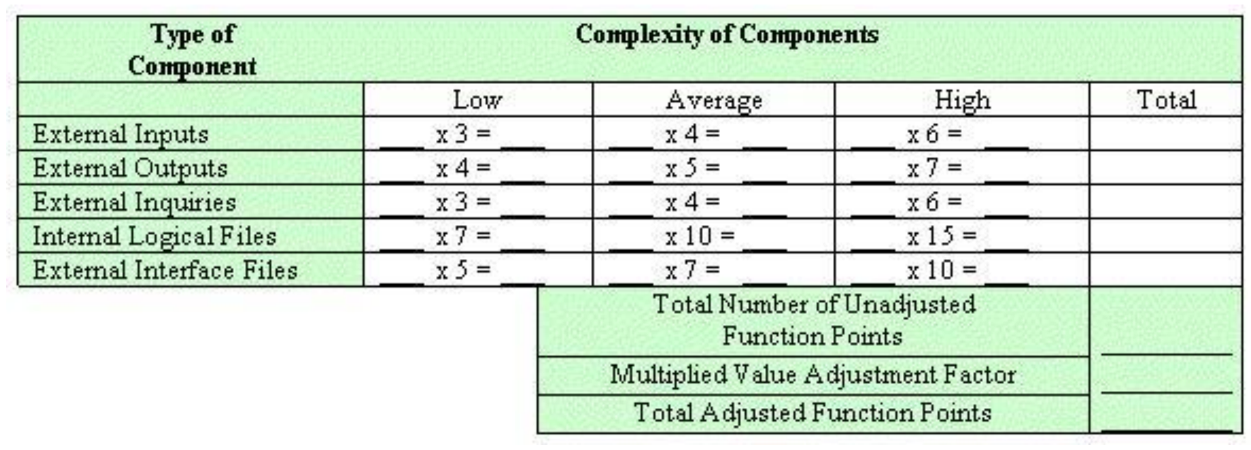
1. ILF (Internal Logic Files): ILF is a set of homogeneous data used and

managed by the application, part of the group "Data".
EIF (External Interface Files): EIF is a set of homogeneous data used by the application but generated and administered by others, belong to the group "Data".
External Input: Elementary operations that use of data from the external environment, is part of the group "Operations".

External Output: Elementary operations that generates data for the external environment, is part of the group "Operations".

External Inquiry: elementary operation that involves both input and output, with a full interaction between user and application. Belong to the group "Operations".

Below is the table of the standard evaluation of the various types of functions.



Type of Component	Complexity of Components			
	Low	Average	High	Total
External Inputs	___ x 3 = ___	___ x 4 = ___	___ x 6 = ___	
External Outputs	___ x 4 = ___	___ x 5 = ___	___ x 7 = ___	
External Inquiries	___ x 3 = ___	___ x 4 = ___	___ x 6 = ___	
Internal Logical Files	___ x 7 = ___	___ x 10 = ___	___ x 15 = ___	
External Interface Files	___ x 5 = ___	___ x 7 = ___	___ x 10 = ___	
		Total Number of Unadjusted Function Points		___
		Multiplied Value Adjustment Factor		___
		Total Adjusted Function Points		___

2.2 Distribution of Function Points:

Internal Logic Files (ILFs)

The application includes a number of ILFs that will be used to store information about

user account (system user)
user(seller and buyer)
transaction (bid on auction)

notification (Winner and payment)

These entities all have simple structures, hence we can use the simple weight :

4x7=28 FPs concerning Internal Logic Files

External Interface File (EIFS)

The application does not use special EIFS to carry out its functions, it does not need data generated by third party software to perform its functions. A future and hypothetical new version of the application could exploit data structures on external sites (such a those of a airline) to extract information from an external DB and insert them into their own. Although there are cues in the specification has not been reported the use of these functions.

FP: 0

External Inputs

You want to distinguish between the two groups in total External Input:

- External Inputs - Side User:
  - Login/Logout: simple operations 2x3=6 FPs
  - Select a product: still simple : 2x3=6 FPs
  - Total:6+6=12 FPs
- External Inputs - Side Buyer:
  - Login/Logout is included in the Client section.
  - Create/delete Bid: medium 4x4=16 FPs
  - Modify Bid :medium 2x4=8 FPs
  - Total:16+8=24
- External Inputs - Side seller:
  - Login/Logout is included in the Client section.
  - Create/delete auction: medium 4x4=16 FPs
  - Modify auction :medium 2x4=8 FPs
  - Total:16+8=24 FPs

- So we have 12+24+24=50 FPs concerning External Inputs

External Inquiry

The system allows buyer/seller to browse all the product and auctions, allows buyer to bid on available auctions . All these can be considered as medium complexity

3x4=12 FPs concerning external inquiries.

Conclusions: Total calculated FP: 90

The analysis has led to the accumulation of 90 identified Function Points, which represent roughly the complexity of the application that has gone to develop. The analysis will continue through COCOMO in the following chapter.

3. Refinement of the analysis using COCOMO

3.1 Notation used:

In this chapter, we refer to the standard formulas for calculating the effort based on the theory of COCOMO. For ease of understanding we'll make a brief summary metrics and formulas used.

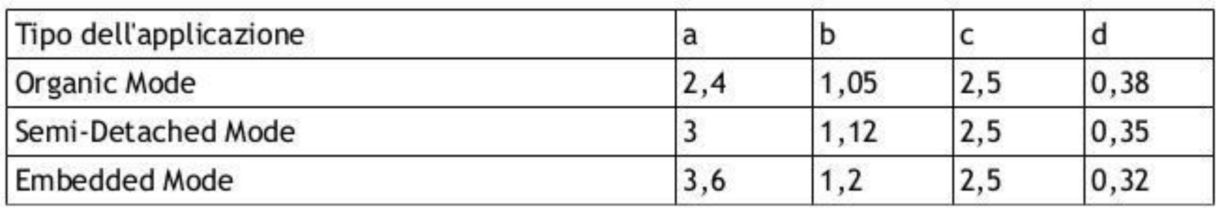
S =Size of the Program

M =Effort expressed in person months {M = aS ^ b}

T = Total time of development expressed in months {T = cS ^ d}

N= number of people involved in the project

The coefficients we are referring to in the above formulas can be viewed in the following table:



Tipo dell'applicazione	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 level of complexity of the application will be considered for the analysis will be

an "Organic Mode" ie we will consider all stages of the development process to estimate the actual effort that would require the development of the program.

3.2 Analysis of the effort using COCOMO:

It is made here analysis using the parameters introduced in the previous chapter we have a size of the program equal to 3.162 KLOC. Using COCOMO applying the organic mode you get an estimate for the efforts = 8.04 and a total time = 3.87, with a number of people = 2,08. In reality the time span of the project was from April 2015 to July 2015 , which is 4 month, 2 team members , M=8 . The estimation is relatively close.

Much of the time was devoted to third phase of the development process, the implementation, which was the largest obstacle in the entire period of work. Moreover, the theory on which these techniques are based for the overall effort point out that the phase that requires more work is the implementation one and data obtained from direct experience have confirmed this assumption.

As regards the aspect of Verification & Testing the effort involved was much smaller than the actual implementation of the functionality, although the time invested in the earlier stages has allowed an easier development of the actual program.

3.3 Conclusion:

In conclusion, the forecasting techniques based on dell'effort basic COCOMO proved to be quite accurate, it does not provide a totally truthful but allowing a preliminary analysis of the resources to employ profit organization of work.

4 Automatic reporting:

For the sake of completeness is reported a result of a reporting automatically generated covering the source with the java code:

```
------------------
LOC
------------------
Total LOC: 3368

Classes LOC:

UseraccountController: 206
Auction: 202
BidController: 179
BuyerController: 170
AuctionController: 167

Packages LOC:

Bean: 1461
Entity: 1136
Controler: 320
util: 132
util: 71

------------------
Lines with imports
```

```
------------------
Total imports: 339

Classes imports:

UseraccountController: 21
Auction: 20
BidController: 19
BuyerController: 17
AuctionController: 17

Packages imports:

Bean: 148
Entity: 116
Controler: 34
util: 12
util: 6

------------------
Blank lines
------------------
Total blank lines: 556

Classes blank lines:

UseraccountController: 40
UserDAO: 33
BidController: 32
Auction: 30
BuyerController: 29

Packages blank lines:

Bean: 253
Entity: 162
Controler: 52
util: 23
util: 12

------------------
Classes count
------------------
Total classes: 32
```

```
Packages with the biggest number of classes:

Bean: 9
Controler: 9
Entity: 8
util: 2
util: 1

------------------
Methods count
------------------
Total methods: 288

Classes with the biggest number of methods:

Auction: 25
Buyer: 19
UseraccountController: 17
Goods: 17
BidController: 14

------------------
Cyclomatic complexity
------------------
Average cyclomatic complexity: 1.3333333333333333

Methods with the highest cyclomatic complexity:

AuctionController::persist: 5
UseraccountController::persist: 5
WinerController::persist: 5
GoodsController::persist: 5
BuyerController::persist: 5

------------------
LCOM
------------------

------------------
Average LCOM 1: 28
------------------

Classes with the highest LCOM 1:
```

```
Auction: 268
Buyer: 145
Goods: 112
Seller: 58
Bid: 58

Packages with the highest average LCOM 1:

Entity: 96
util: 28
util: 19
Bean: 7
Controler: 1

------------------
Average LCOM 2: 0.42895281629705007
------------------

Classes with the highest LCOM 2:

Auction: 0.92
Buyer: 0.8947368421052632
Goods: 0.8823529411764706
Seller: 0.8461538461538461
Bid: 0.8461538461538461

Packages with the highest average LCOM 2:

Entity: 0.8589893697633636
Bean: 0.4961873638344227
util: 0.4
Controler: 0.05555555555555555
util: 0.0

------------------
Average LCOM 3: 0.4722433274191086
------------------

Classes with the highest LCOM 3:

Auction: 0.9583333333333334
Buyer: 0.9444444444444444
Goods: 0.9375
```

```
Seller: 0.9166666666666666
Bid: 0.9166666666666666

Packages with the highest average LCOM 3:

Entity: 0.9237847222222223
util: 0.0
util: 0.5
Bean: 0.5333422364672364
Controler: 0.06349206349206349

------------------
Average LCOM 4: 3
------------------

Classes with the highest LCOM 4:

Auction: 12
Buyer: 9
Goods: 8
JsfUtil: 8
JsfUtil: 8

Packages with the highest average LCOM 4:

util: 8
Entity: 7
util: 6
Bean: 3
Controler: 1
```