A Tool For Cost Estimation Using Cocomo Ii And Object Oriented Metrics

Rahul Chaudhary, Anuj Kumar, Amrendra Pratap
1. M.Tech(CSE),ASET,Amity University Noida,
2. M.Tech(IT),ASET,Amity University Noida,
3. M.Tech(CSE),ASET,Amity University Noida,

Abstract

The Software Cost Estimation Tools are becomes prominent in use. Although there are many more tools present, but there is not a single tool that allow comparison of cost and effort of two Real Time Project on a single window (in parallel), by using Grouping Method, Object Oriented Metrics and COCOMO II.

This paper going to demonstrate my Software, on which three modules are designed

First module to calculate line of code using Object Oriented Metrics (Class Diagram, Sequence Diagram and Coding Files),

Second module which is dedicated to execute first module to number of times(i.e. both project in parallel or only first project or only second project) and the execution depend on groups(combination of Class Diagram, Sequence Diagram and Coding Files) that are made for real time projects separately using Grouping Methodology,

Third module is the implementation of COCOMO II over the calculated Lines of Code in second module with the help of their four models i.e. Early Design Model(EDM), Post Architecture Model(PAM), Application Composition Model(ACM) and Reuse Model(RM).

This way easily able to Estimate and Compare the cost and effort more accurately by taking help/using three technologies which are very prominent; they are Grouping Methodologies, Object Oriented Metrics and COCOMO II. All this work helps a manager or Estimator or User of Software:

a) *To use the previous work (project) in new Real Time Project i.e. there are many references available for continuing to new Real Time Project, this Tool identify which reference is more suitable to use(effective to Cost and Effort).*
b) *When same goal project is developed by two different logics, then this Tool helps to compare between both Real Time Project in a single Dynamic window on the basic of Object Oriented Metric*

IJERT

Keywords

Grouping Methodologies, Object Oriented Metrics, COCOMO II, Class Diagram, Sequence Diagram, Coding

1. Introduction

This paper mainly focuses on the development of tool or comparison of two past work (Real Time Project) in order to select best among them, to implement the result in new project(Real Time Project) development and the comparison is done with the help of Object Oriented Metrics(i.e. Case Diagram, Sequence Diagram etc).

Although there are many other Work/Tools are already constructed in market and also available easily, but there is no software present in market which help a manager to compare Real Time Projects that are done using different method for a unique goal, by the help of Object Oriented Metrics.

Various given point gives step by step light over the concept behind this development:

- Login Module for Authentication.
- Sequential (one after another) analysis of Groups.
- SLOC (single line of code) calculation using Case Diagram.
- SLOC (single line of code) calculation using Sequence Diagram.
- SLOC (single line of code) calculation using Coding Files.
- Total SLOC (single line of code) calculation.

- Implementation of various COCOMO II sub-models.
- Calculation of Cost and Effort.
- Comparison of Result using various Techniques (tables, values, graphs etc.).

Steps (2, 3, 4, 5, and 6) are explained next in the section 3. Rest of other steps in illustrated through example in Section-4.

This tool helps a manager for estimating in different aspect, as follow:-

When a manager want to use the previous work (project) in new Real Time Project i.e. there are many references available for continuing to new Real Time Project, this Tool identify which reference is more suitable to use as reference to Cost and Effort.

When same goal project is developed by two different logics, then this Tool helps to compare between both Real Time Project in a single Dynamic window.

From 1960s, there are many software project models, such as – COCOMO (Constructive cost model) by B.W.Bohem[1], Albrecht's FPA (functional point analysis), and Putman Model by Putman [2]. Out of which FPA is very representative [3], use of various sub-model Methods for Cost Estimation will decrease the Error margin in between ± 20%. Whereas after the development realized that impact of this error is still large. Therefore use COCOMO II model, to set the forecast accuracy of our software as our first goal, and then discuss Cost Estimation accuracy from the Practice and Innovative views.

IJERT

1. Technologies used for Estimation:

1.1. COCOMO II:

As COCOMO (constructive cost model) is one which gives manager a path to estimate project Duration and Cost. It was developed by Barry Boehm in the early Eighties [7].

The COCOMO II [8] Model is update of COCOMO Model in order to address software development during 1990 & 2000s. The model is now invigorative software engineering artefact that has, customer perspective, with following features:

I. Simple and well tested model.
II. Provide accuracy about 70% in time and 20% in cost.

In general COCOMOII is used to estimate project cost, directly from person-months efforts, just assuming that project file size expressed in (SLOC) i.e. thousands single lines of code. Using estimation formula:

$$effort(in\_person\_month) = a * KSLOC^{b}$$

Where a (coefficient) is about 2.94 and b (scaling factor) is about 1.0997.

In practice COCOMO parameters are very different from there typical values, COCOMO II provides classification of various factors that have their influence on Project cost estimation.

While comparing COCOMO and COCOMO II is the introduction of five scaling factors, out of while three directly related to project manager's activities, and help to raise the role of management of project in reducing costs related to project.

- CMM levels-Process maturity is accounted.
- Accounting of degree to which project arch exists and stabilized before phase of construction.
- Accounting relationship perspective i.e. Team Cohesion, Stockholders and Relations.

1.2. SLOC (size in single line of code) calculation measure:

The line of code generated by software activities and the source program code which is used to measuring of linage (most commonly), also instruction number of object code [4] (2.2), delivered system documentation phases. The calculation of single line of code includes design and analysis, written program, instruments tests and documentation of system in person-month.

1.3. Measures in Object Oriented concepts:

Focus on C&K[5] and MOOD[6] Measures to identity weather a design is reasonable or not reasonable. C&K gives six class level Metrics: Each class weighted method, inheritance tree depth, total children; object classes coupling, responses for a class, methods having lack of cohesion.

MOOD Measures allusion to four attributes i.e. Polymorphism, Inheritance, Encapsulation and Message transferring of object oriented. Two most important measures i.e. methods of attributes are main characteristics of MOOD.

2. Software project Cost Estimation:

Divide the software into components. This is called splitting or grouping of software which not require unnecessary internal knowledge and his helps to configured to complete word as required, hence by this way, it is easy and convenient for developer and it provides great convenient for estimation.

After grouping, application having two relationships:

Integer relationship b/w classes with in an individual group.

The relationship b/w various groups are previously created.

Now while see the individual group then realize that large-scale work/application is decomposed into small work, which is much easy to analyses and solve. This leads to improve accuracy of estimation results. To the relation b/w groups, because of several feature details of grouping are hidden, the application granularity get smaller, and dramatically complexity reduced.

Now, for individual group, often details design completion each group will have.

- Complete class diagram.
- Complete sequence diagram.

Now, use both of above complete class and sequence diagram, i.e. as follow

- Use number of class in class diagram.
- The number of methods in all class.

The relationship b/w class C i.e. aggregation, inheritance etc, to calculate total size i.e. single line of code.

While dealing with high complexity , weighted for reasonable value is required and then use COCOMO2 model for calculation of cost and effort for relation b/w various groups, where when a group call another is a relationship that usually exist or data transfer b/w different group. Whereas this complete process is mainly suitable for functional point analysis.

Here in this paper, demonstrating a new method through as my software describe in [7] to calculate the function point through these group diagram and use case figure/diagram.

Basic idea behind this development is UML class diagram and sequence diagram in order to calculate single line of code size of each group individually. Firstly use C&K [5] and MOOD [6] measure for measuring of each class in design groups and then from the output determine complexity of classes together with measurement of reasonable design.

Now, for reasonable designs, proceeds to size calculate on of code for each reasonable group, **this calculation done as follow.**

a) For Class Diagram:

In the class diagram, there several methods which require only one line of code to elaborate like get value, set value etc. All these are taken as single line of code and record to database, together with a record of method name in database where as several are much complex and needed more single line of code to elaborate hence to note those methods from class diagram such as insert, division and sequence diagram of those methods are needed to draw for calculating total single line of code corresponding to those method.

IJERT

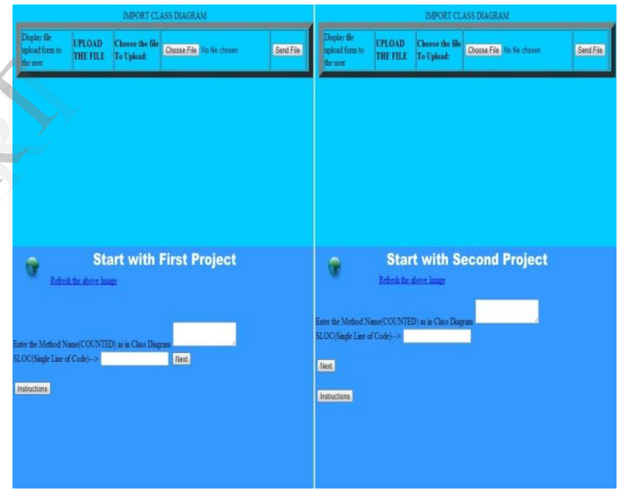

IMPORT CLASS DIAGRAM
Display file upload form to the user
UPLOAD THE FILE
Choose the file To Upload:
Choose File
No file chosen
Send File
Start with First Project
Refresh the above Image
Enter the Method Name(COUNTED) as in Class Diagram
SLOC(Single Line of Code)-->
Next
Instructions
IMPORT CLASS DIAGRAM
Display file upload form to the user
UPLOAD THE FILE
Choose the file To Upload:
Choose File
No file chosen
Send File
Start with Second Project
Refresh the above Image
Enter the Method Name(COUNTED) as in Class Diagram
SLOC(Single Line of Code)-->
Next
Instructions

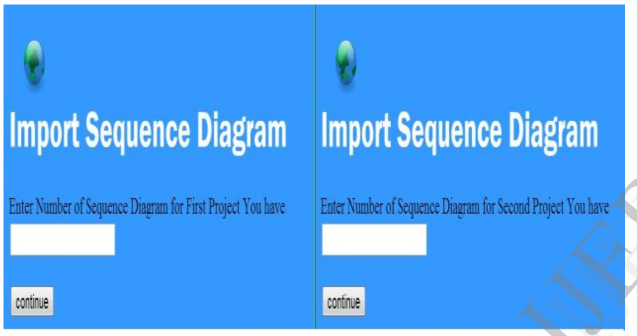
Figure 1 Class Diagram Analysis

NOTE: The methods analyzed, needed to input there name or record their name to database.

b) For Sequence Diagram:

First, you have to enter number of methods, that are with your corresponding group which is currently used process, whose sequence diagram are analyzed for calculating single line of code. And then our software window repeats that number of time for the analysis of all sequence diagrams. Hence in analysis of sequence diagrams are focus on real line arrow (each operation) as single line of code (one line), but if the method is more complex, then single operation can also noted as more than one line, and here to estimate source code size of sequence diagram very properly.

Although, if complexities of class in grouping are different, to the method of high complexity, use weighted value to multiply its source code by using Carrying out complexity analysis and weighted value estimation. Apply CBO method and COF method.



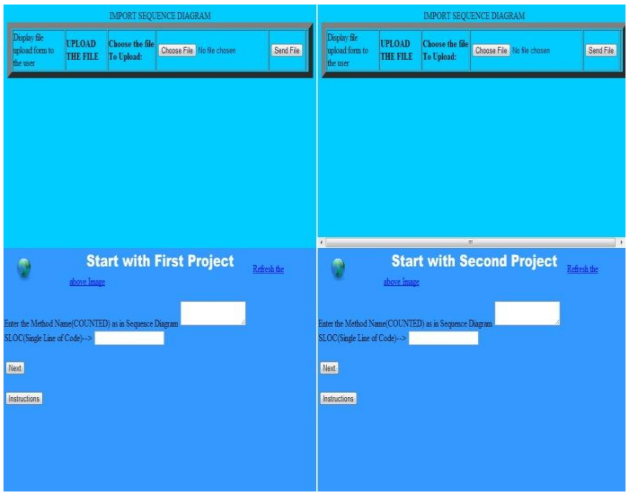
Import Sequence Diagram

Enter Number of Sequence Diagram for First Project You have

Import Sequence Diagram

Enter Number of Sequence Diagram for Second Project You have

Figure 2: Number of Sequence Diagram



IMPORT SEQUENCE DIAGRAM

Display file upload form to the user	UPLOAD THE FILE	Choose the file To Upload:	Choose File No file chosen	Send File

Start with First Project

Refresh the above Image

Enter the Method Name(COUNTED) as in Sequence Diagram

SLOC(Single Line of Code)-->

IMPORT SEQUENCE DIAGRAM

Display file upload form to the user	UPLOAD THE FILE	Choose the file To Upload:	Choose File No file chosen	Send File

Start with Second Project

Refresh the above Image

Enter the Method Name(COUNTED) as in Sequence Diagram

SLOC(Single Line of Code)-->

Figure 3 Sequence Diagram Analysis

NOTE: The methods analyzed, needed to input there name or record their name to database.

After estimation of size of project code through class and sequence diagrams- proceeds to coding.

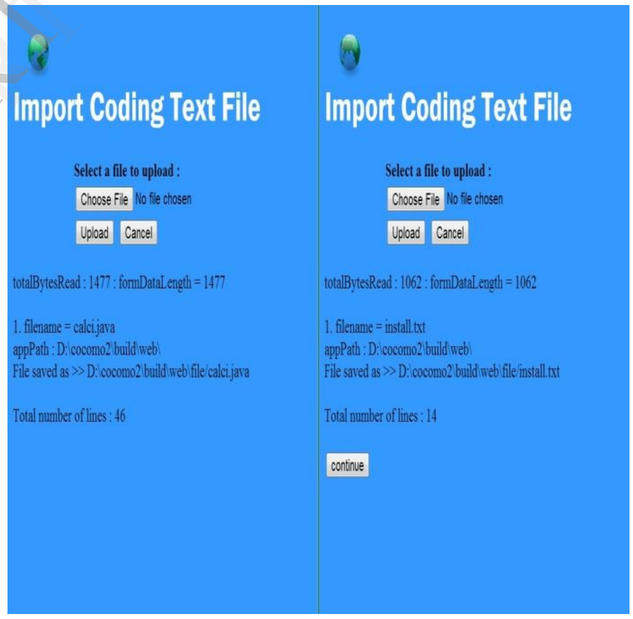
c) Coding File Analysis:

Browse Coding files to software for calculating total single line of code in file, this software automatically remove single line comments, multiline comments and white/ blank lines, It imports all files to their database and auto record all the results to database, including those SLOC which are already calculated using class and sequence diagrams analysis.

d) Total SLOC Calculation:

Now, have all method names that are analyze in class diagrams analysis & sequence diagram analysis and also total SLOC calculation in coding file analysis. This data is used to calculate Significant number of accurate SLOC, Software auto recognize all the methods by name and as it have all coding files loaded in it, Now it remove those SLOC which are already counted in the Class Diagram Analysis & Sequence Diagram Analysis by recognizing the module of concern methods name that saved in database during previous steps.

IJERT



Import Coding Text File

Select a file to upload :

```
totalBytesRead : 1477 : formDataLength = 1477

1. filename = calci.java
appPath : D:\cocomo2\build\web\
File saved as >> D:\cocomo2\build\web\file/calci.java

Total number of lines : 46
```

Import Coding Text File

Select a file to upload :

```
totalBytesRead : 1062 : formDataLength = 1062

1. filename = install.txt
appPath : D:\cocomo2\build\web\
File saved as >> D:\cocomo2\build\web\file/install.txt

Total number of lines : 14
```

Figure 4 Coding Files Analysis (allow any number of file select)

Hence this way, calculate total SLOC related to concern group, and also for every group in sequential fashion,

Repeat section 3 repeatedly for every group in order to calculate each group separately, this realistically increases the accuracy of estimating the source code size, and average margin is now bounded b/w ±5%.

3. Example to Illustrate ([4] (Sec-4)):

Example provides accuracy in result of size computation of source code. for user information components. This component based on database operation i.e. getting user's information, creating user (new), modify user detail, deleting user information & updating user's status.

Using Java language to develop the component, and JDBC used to operate databases, the class diagram in fig **given below:**

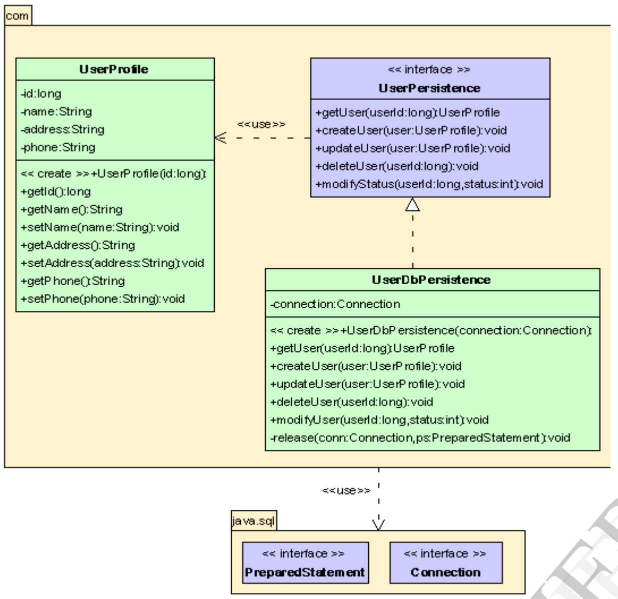

com
UserProfile
-id:long
-name:String
-address:String
-phone:String
<< create >>+UserProfile(id:long)
+getId():long
+getName():String
+setName(name:String):void
+getAddress():String
+setAddress(address:String):void
+getPhone():String
+setPhone(phone:String):void
<<use>>
<< interface >>
UserPersistence
+getUser(userId:long):UserProfile
+createUser(user:UserProfile):void
+updateUser(user:UserProfile):void
+deleteUser(userId:long):void
+modifyStatus(userId:long,status:int):void
UserDbPersistence
-connection:Connection
<< create >>+UserDbPersistence(connection:Connection)
+getUser(userId:long):UserProfile
+createUser(user:UserProfile):void
+updateUser(user:UserProfile):void
+deleteUser(userId:long):void
+modifyUser(userId:long,status:int):void
-release(conn:Connection,ps:PreparedStatement):void
<<use>>
java.sql
<< interface >>
PreparedStatement
<< interface >>
Connection

Figure 5 [9] Class Diagram of user info.

Here User Profile is an entity class, used to store information that is user required. To make this component more flexible, define interface called User_Persistence, and five methods on it to achieve functions that are above mentioned. User_Persistence implements User_Db_Persistence and information of user stored in database if the class diagram is reasonable (reasonability explained in section-3) then calculate Source Code Size of Component. According to method explained in analysis of Class Diagram, those methods in User_Db_Persistence are more complex & required to have sequence diagram. For example let deleting user's Information Method.

IJERT IJERT

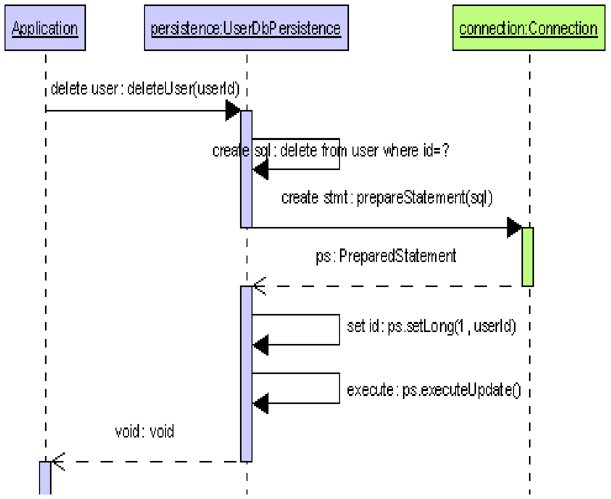
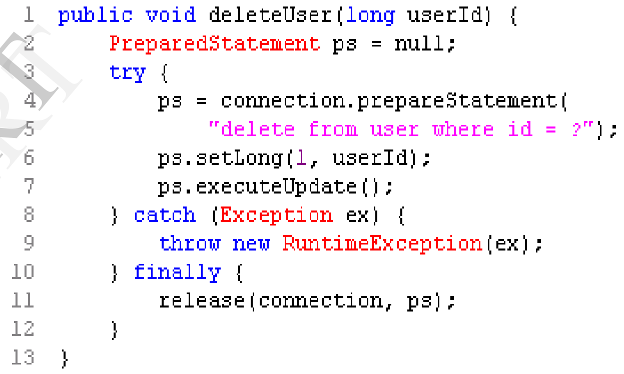

Application
persistence:UserDbPersistence
connection:Connection
delete user : deleteUser(userId)
create sql : delete from user where id=?
create stmt : prepareStatement(sql)
ps : PreparedStatement
set id : ps.setLong(1 , userId)
execute : ps.executeUpdate()
void : void

Figure 6 [9] Sequence diagram for deleting Method

Get the estimated SLOC is 5 and figure 7 shows implementation of method to delete user



```
public void deleteUser(long userId) {
    PreparedStatement ps = null;
    try {
        ps = connection.prepareStatement(
            "delete from user where id = ?");
        ps.setLong(1, userId);
        ps.executeUpdate();
    } catch (Exception ex) {
        throw new RuntimeException(ex);
    } finally {
        release(connection, ps);
    }
}
```

Figure 7 [9] Coding for deleting User

Get the estimated SLOC is 10.

While calculating size of source code, using following rules:

- If there is semicolon, it is new line.
- Logical operation counted as one line i.e. if_else, do_while, for, try…catch…finally etc.
- The right bracket not recognized as single line.
- Breaking of one statement in more than one line, then it can be counted as on line.

Using the above rules count the Size of coding file i.e. as shown in (figure 7).

The last step is to use Cost Estimation Model COCOMOII for Cost Estimation, Which take the calculation size of source code as input & used for achieving the result but there is little difference, i.e. no direct implementation of COCOMO II is possible but in its place use its 4 sub models named as:

- Early design model
- Post-architecture model
- Application composition model
- Reuse model

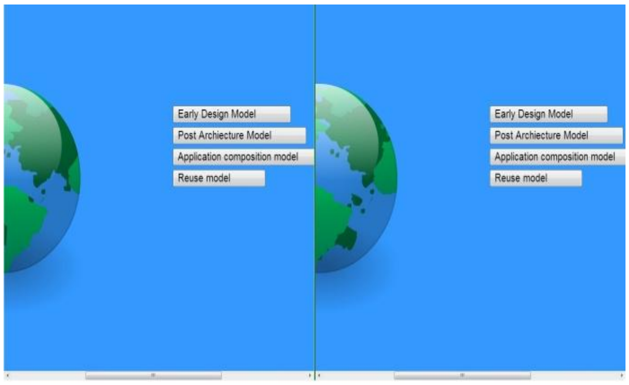
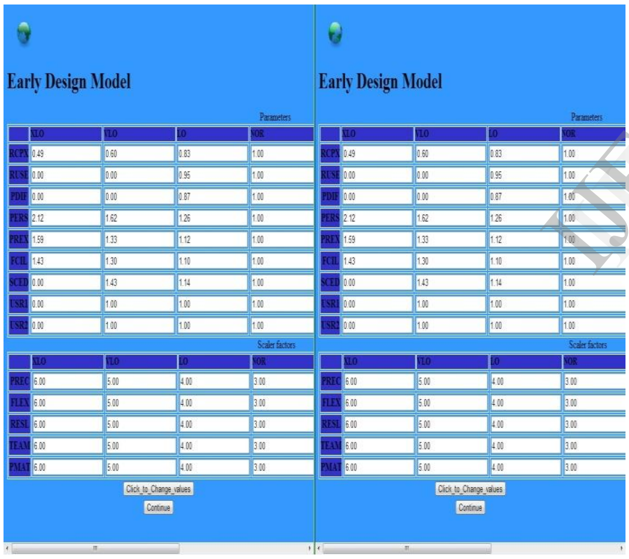


Figure 8 Model Selection Snapshoot



Early Design Model

Parameters

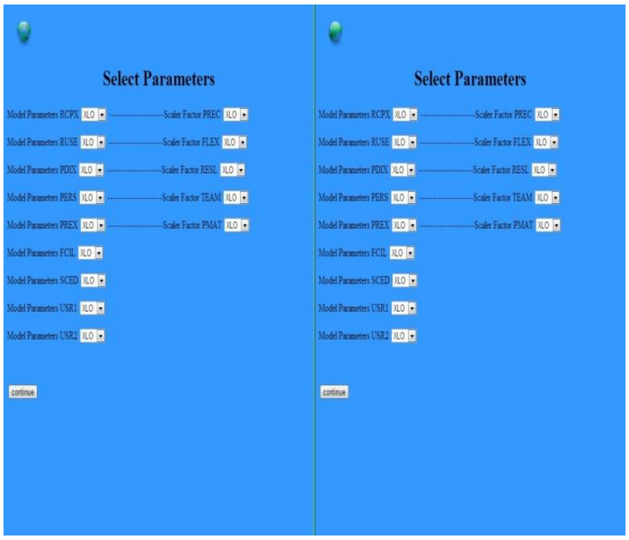
	XLO	VLO	LO	NOR
RCPX	0.49	0.60	0.83	1.00
RUSE	0.00	0.00	0.95	1.00
PDIF	0.00	0.00	0.87	1.00
PERS	2.12	1.62	1.26	1.00
PREX	1.59	1.33	1.12	1.00
FCIL	1.43	1.30	1.10	1.00
SCED	0.00	1.43	1.14	1.00
USR1	0.00	1.00	1.00	1.00
USR2	0.00	1.00	1.00	1.00

Scaler factors

	XLO	VLO	LO	NOR
PREC	6.00	5.00	4.00	3.00
FLEX	6.00	5.00	4.00	3.00
RESL	6.00	5.00	4.00	3.00
TEAM	6.00	5.00	4.00	3.00
PMAT	6.00	5.00	4.00	3.00

Figure 8 Early Design Model (selected) Snapshot

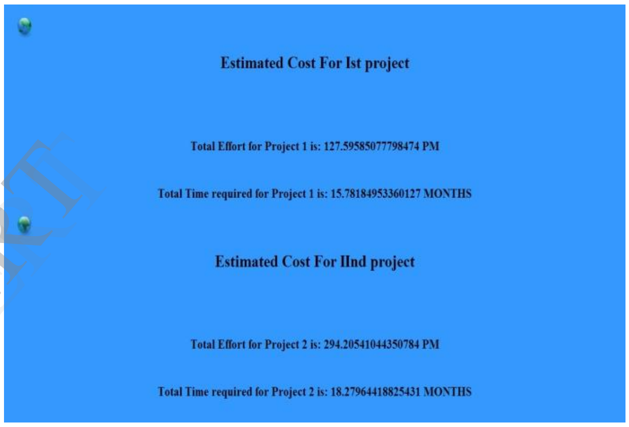
For calculation of Cost Estimation for both the Window in parallel, hence using this approach, i.e. the use of some model for task estimation increase our system flexibility to do their task and also provide more choice to the user.



Select Parameters

Model Parameters RCPX XLO ------------------Scaler Factor PREC XLO
Model Parameters RUSE XLO ------------------Scaler Factor FLEX XLO
Model Parameters PDIX XLO ------------------Scaler Factor RESL XLO
Model Parameters PERS XLO ------------------Scaler Factor TEAM XLO
Model Parameters PREX XLO ------------------Scaler Factor PMAT XLO
Model Parameters FCIL XLO
Model Parameters SCED XLO
Model Parameters USR1 XLO
Model Parameters USR2 XLO

Figure 9 : Early Design Model (parameters) Snapshot



Estimated Cost For Ist project

Total Effort for Project 1 is: 127.59585077798474 PM

Total Time required for Project 1 is: 15.78184953360127 MONTHS

Estimated Cost For IInd project

Total Effort for Project 2 is: 294.20541044350784 PM

Total Time required for Project 2 is: 18.27964418825431 MONTHS

IJERT

Figure 10 Early Design Model (Result) Snapshot

4. Conclusion:

Correct and accurate Estimation in cost of project (guarantee our work). Software project cost estimation methods based on grouping/groups as they are new methods that estimates software project cost accurately, which is then compared between both the window (or projects result) result and help to fetch out much accurate and correct comparison that helps user/manager to select best old work for their future project. All this is based on RUP (Rational Unified Model), and this is milestone in Software Development Cost Estimation together with it helps to reduce cost and time engaged in software development and estimations.

5. Future Work:

- Analysis of class diagram Image through DIP (Digital Image Processing) and AI (Artificial Intelligence).

- Analysis of Sequence diagram Image through DIP (Digital Image Processing).
- Making the solution online.

6. References

[1] Boehm B W. Software Engineering Economics. Englewood Clifs, NJ: PrenticeHall, 1981.

[2] H. Putnam. "A General Empirical Solution to the Macro Software Sizing and Estimating Problem". IEEE Transactions on Software Engineering,1978(7), 1(4).

[3] P. S. Pressman. “Software Engineering”. National defense industry Press. 1988.

[4] JIN Yongqin,LI Jianning, CHEN Qingzhang,”Software Project Cost Estimation Based On Groupware”, 2009

[5] Shyam R. Chidamber, Chris F. Kemerer. Towards A Metric Suite for Object-Oriented Design. IEEE Trans.Softw.Eng, 1994.

[6] Abreu, Fernando B. "The MOOD Metrics Set," Proc. ECOOP'95 Workshop on Metrics, 1995.

[7] “software Engineering economics ” by Barry Boehm, Prentice Hall, 1981.

[8] “Software Cost Estimation With COCOMO II”, Prentice HALL, 2000.

[9] Software Project Cost Estimation Based On Groupware by JIN Yongqin1, LI Jun, LIN Jianming,CHEN Qingzhang

IJERT