A Comparative Appraisal & Assay of Parametric Models for Enrichment of Software Effort Estimation

A Dissertation submitted

By

Raksha Pandey

to

Department of Computer Science and Engineering

In partial fulfillment of the Requirement for the

Award of the Degree of

Master of Technology in Computer Science & Engineering

Under the guidance of

Er. Dalwinder Singh

(Asst. Professor)

(May 2015)

ABSTRACT

Accurate estimation of cost for software project is one of the prime requisite in software development organization. Accurate cost estimates help the customer to make successful investments as well as assist software project manager with appropriate plans for the project and making reasonable decisions during the project execution. An estimate is not really a prediction, it is a management goal. While assessing of the proposed system consists of evaluating whether the required functionality can be achieved with current affordable technologies. Measure of work involved in completing a project is called size of the project. Effort and time required to develop software can be computed by estimating the project size. The proposal is to sensibly hybrid parametric model with size estimation model to allow us to determine a set of homogeneous projects by using a technique derived from estimation by analogy. Two models have been selected to sensibly hybrid and provide more strength to estimation model. Size estimation model known as Function Point and SLIM parametric model are chosen for hybridization.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my thesis guide Prof. Dalwinder Singh for the continuous support of my M.Tech study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my M.Tech study.

Besides my thesis guide, I would like to thank the rest of my teaching faculty for their encouragement, insightful comments, and hard questions.

Last but not the least; I would like to thank almighty God and my family for supporting me spiritually throughout my study.

CERTIFICATE

This is to certify that Raksha Pandey has completed M.Tech dissertation titled **"A Comparative Appraisal & Assay of Parametric Models for Enrichment of Software Effort Estimation"** under my guidance and supervision. To the best of my knowledge, the present work is the result of her original investigation and study. No part of the dissertation has ever been for any other degree or diploma.

The dissertation is fit for the submission and partial fulfillment of the conditions for the award of M.Tech Computer science & Engg.

Date:________________

Signature of Advisor

Name: Er. Dalwinder Singh

UID:

DECLARATION

I hereby declare that the dissertation entitled, **“A Comparative Appraisal & Assay of Parametric Models for Enrichment of Software Effort Estimation”** submitted for the M.Tech Degree is entirely my original work and all ideas and references hace duly acknowledged. It does not contain any work for the award of any other degree or diploma.

Date: ______________________

Raksha Pandey

Reg. No. 11300032

TABLE OF CONTENT

CONTENT **PAGE NO.**

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Even though demand of software system is increasing in all aspects of human life, development of software projects are hysterically noted with delays, high cost and errors. Inaccurate estimation of resources is major factor of failure in software project [17]. Software estimation draws enormous attention from academicians and specialists.

Software estimation is the mechanism of predicting cost, effort and duration that are required to develop software [7]. Estimator often depends on number of pragmatism to generate software estimations [6]. Exceeded budgets, function that are not developed completely, low quality and partial completion of the project are some of the major factors that results in underestimation or overestimation of the software cost [7]. Estimation of cost and size of the software project are the biggest challenge. A project budget, schedule and size of development team are directly bank on the estimation. The process of software development effort estimation in which estimator predict the amount of effort in term of person-hour or monthly to maintain software are based on uncertain, noisy input and incomplete project plan, budgets, pricing processes and investment[2].

1.1Goals of Software Engineering

The four fundamental goal of the software engineering are modifiability, efficiency, reliability and understandability are as follows:

- Modifiability: are about cost of change.
- Efficiency: meeting the specified target without wastage of resources, processor utilization, space, network bandwidth, money, time etc.
- Reliability: consistency among computer program for a specified time under given environment.
- Understandability: It indicates that design, documentation that is done by user must be clearly written. It specifies the attribute of software that specifies the logical concepts and its applicability.

1.2 Basic Activities of Software Engineering

The four basic process activities are:

- Specification: It is the process that defines what services are desired from the system and identify its necessity on the system operation and development.

- Software design and implementation: The conversion of system specification into an executable system is course of implementation stage. It also involves the process of system design and programming.

  A software design describes the structure of the software to be implemented between system components and algorithm used.

- Software Validation: it defines formats and input criteria.
- Software evolution: it specifies maintenance of the system.

1.3 Some Myths Related To Software: There are many myths that are related to the software. In the software development many myths are used:

- Easy to change: One myth is that the software is easily modified. Editing is possible up to some extent but these modifications are not easy. To make changes without introduction of errors are tedious job to do as source code can be easily changed if errors are properly introduced. Verification is required to make the changes.
- The replaceable device has the less reliability than the computer system: The other in the software development process is that they are more reliable than the devices that are needs to replace.
- Software testing removes all the errors: Testing of the system means all the errors bugs are removed which in nothing but mere a myth. Testing only reveals number of error present in the system.
- Software safety can be increased with reusability: if the software is used again and again doesn’t give guarantee of hundred percent secure systems. Before code is being reused it should be properly analyzed and tested by software engineers.

- Software can work properly on its first time: Complete myth to believe that software can only work during starting days. If they are properly maintained, tested and errors or bugs are removed time to time they can give lifelong performance.
- Software with more features is better software: Unbelievable to believe that adding extra feature enhances the performance of the software. It will only make the system more complex and tedious job to do.

1.4 Software Engineering Sub-discipline

Sub-discipline has ten parts [19]:

- Software Requirement: It includes citation, assay, stipulation and validation.
- Software Design: It defines the architecture, characteristics of a system, interfaces and components.
- Software Construction: The precise creation of work.
- Software Testing: It verifies the behavior of a program based on test cases.
- Software Maintenance: It provides cost-effective platform to the software.
- Software Configuration Management: Consistently controlling changes in the configuration, and preserves its integrity.
- Software engineering process: The measurement, improvement, evaluation, definition, change, and implementation of the software life cycle process.
- Software engineering tools and methods: Tools support the software life cycle processes and the methods make the activities of software engineering more efficient through its structure.
- Software quality management: A set of essential characteristics that fulfills requirements.

The above mentioned sub-discipline play vital role in software engineering. As mentioned in different parts of the sub-discipline requirement are used for assay stipulation, design for the purpose of architecture and its characteristics, construction used for creation for work where software configuration management used for keeping eyes on any changes in the configuration and it also maintain its integrity. Different tools that are used in software engineering are basically used to support the software life cycle processes to make the system and its structure more efficient. The steps of the software engineering are as follows :

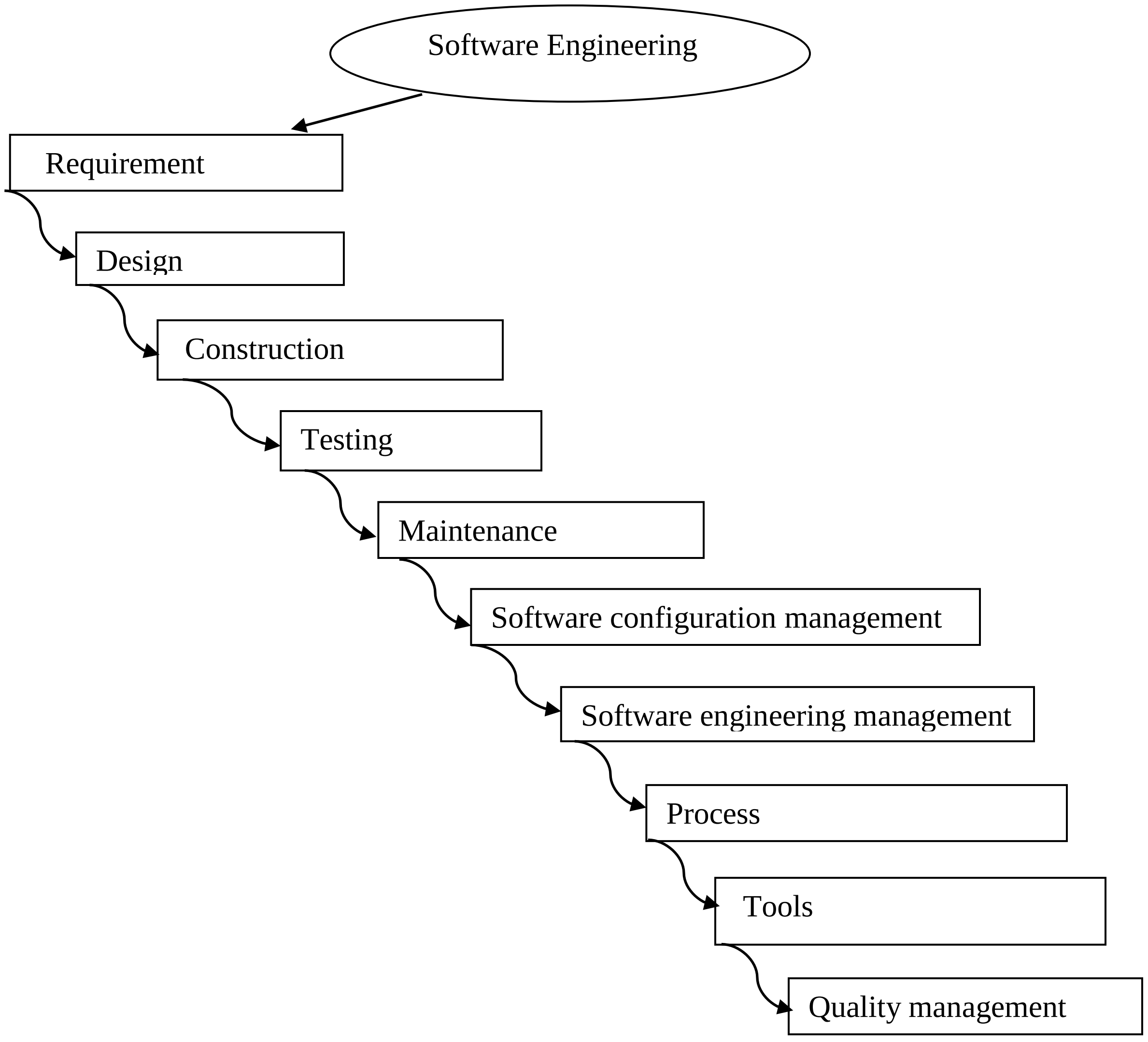


Figure 1.1: Software Engineering Steps

1.5 Software Cost Estimation

Software cost estimation plays a vital role in software engineering as the success or failure of project entirely depends on it. Cost estimation's deliverables like staff requirements, schedule and effort are important chunk of information for formation and execution of project. They provide inputs for project request and proposal, project planning, control, budget, progress monitoring & staff allocation. Illogical and uncertain estimates are the root cause of project failure. So, the capability of the system to find out correct time and cost of the software is very crucial for the progress of the system.

The software engineering community puts enormous effort while buildings models in order to comfort estimators to provide accurate cost estimates for software projects. COCOMO, SLIM, SEER-SEM and Price- S are some of the estimation models that have been proposed and used in the last three decades.

1.5.1. Models for Cost Estimation

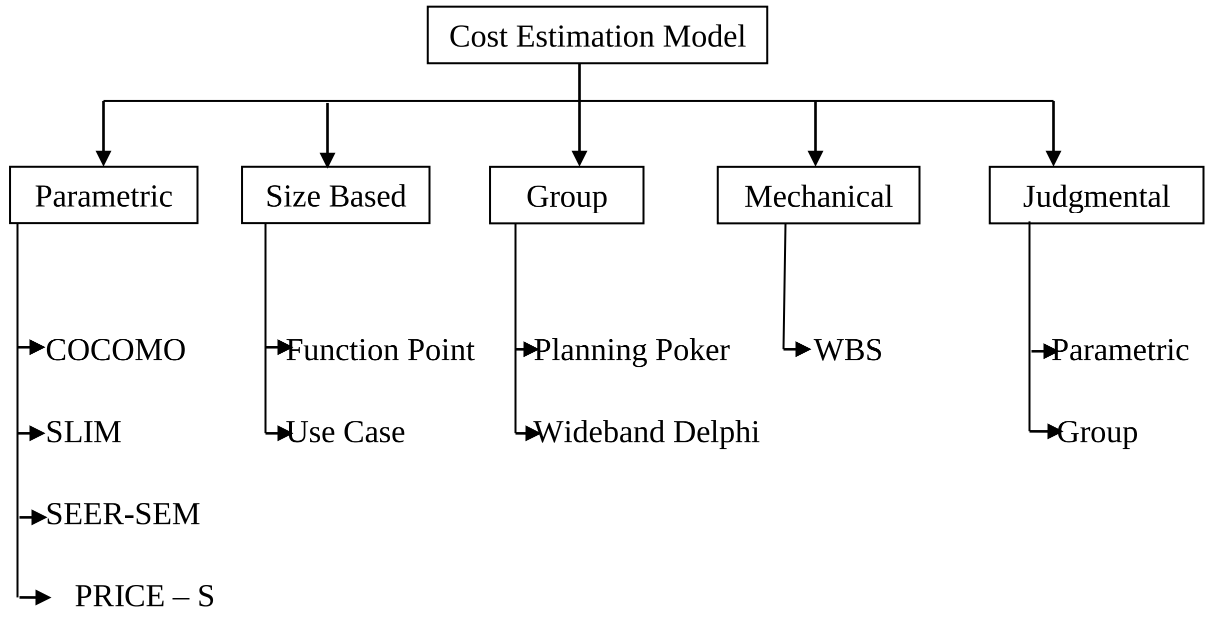


Figure 1.2: Cost estimation approaches

1.5.1.1 Parametric Model [12]: It is an estimation technique that applies on one or more cost estimating affair and combines mathematical relationships and logic. It describes variables that provide numerical estimates for vital input variables that influence the effort or time used in development.

COCOMO: It is a mechanism for assessing the cost of software. Its three levels are.

➢ Basic COCOMO**:** measures effort and cost as a function of program size. Its three mode are:

- ➢ Organic Mode: It is simple & affects small experienced teams. The DSI are typically in a small amount that is under 50,000 and planned software is not considered innovative.

    - Semidetached Mode: are much difficult than the organic mode and team members have mixed levels of experience. Approximately, 300,000 DSI are required by the software. It combines the characteristics of both modes.
    - Embedded Mode: highly complex form of hardware, software, regulations and operating procedures are dealt by this mode.
- Intermediate COCOMO: a set of “cost drivers” are used to extend the basic model for measuring the effort. Valuation of personnel or hardware is assessed through this mode.
- Detailed COCOMO: an extension of the Intermediate model that adds effort multipliers to determine the effect of cost drivers on each phase.

SLIM: Software Life Cycle Model was developed by Larry Putnam. SLIM obtain the probabilistic principle which are known as Rayleigh distribution between effort and time. SLIM is essentially applicable for large projects which usually have more than 70,000 lines of code [16]. It is first algorithmic cost model. It is based on the Rayleigh function also known as a macro estimation model. Functions of SLIM are [2]:

- Calibration: It translates a historical database of past projects.
- Build: It collects software characteristics, personal and computer attributes to build an information model.
- Software sizing: mechanized version of the line of code costing technique.

The software equation to estimate the effort for a software task is [21]:

$$K= (LOC/(C*t^{4/3}))*3$$

Where,

K is the total life cycle effort in working years.

t is development time.

C is the technology constant.

The value of technology constant varies from 610 to 57314.

The SLIM model assumes follows a form of Rayleigh probability model. The shapes and sizes of the Rayleigh curve reflect: Size, manpower buildup index, and other productivity parameters. The Rayleigh staffing level at time this presented as

$$P\ (t) = K/\ t_d^2\ te^{(-\ t2\ /2t2d)}$$

Where,

k is total lifecycle effort.

t_d is schedule to the peak of the staffing curve.

The quality: $D=K/t^2d$ consider staffing complexity of the project.

Merit of SLIM

- Uses linear programming to measure both cost and effort.
- SLIM need fewer parameter for estimation while comparing it with COCOMO.

Demerits of SLIM

- Factor that is responsible for making this model sensitive is technology.
- It is considered to be suitable only for the large project as SLIM is not suitable for small project.

SEER-SEM: Powerful estimation is the key feature of this model. For performing estimation, it has been developed with a combination of estimation function. The SEER-SEM model was influenced by the frameworks of SLIM [10].

Demerits of SEER-SEM [18]:

- It has around 50 input parameters which increases the complexity of SEER-SEM, mainly in case of the unpredictability from outputs.
- Its specific details increase the difficulty of identifying the nonlinear relationship between the parameter inputs and its corresponding outputs.

PRICE- S [18]**:** Its purpose is to estimate acquisition and development of hardware systems whereas its first commercial software estimation model with the name of True S was released in 1970's which is the result of multiple extensions, change of ownership and upgrades.

PRICE-S is an activity based estimation model which is used to estimate the effort of activity. An activity can be the equipment, facilities, technologies and the work that people perform.

1.5.1.2 Size Based Estimation Model: The focus of estimation is on identifying the "parameters" that provides the size which are key focus for the tasks in the project.

FUNCTION POINT [21]: This is a top down method devised by Allan Albrecht. Albrecht was investigating programming productivity and needed to quantify the functional size of programs independently of their programming languages. The basis of function point analysis is that information system comprises five major component or 'external user types' in Albrecht terminology that are benefited to the user.

- External input types are input transactions which update internal computer files.
- External output types are transactions where data is output to the user.
- External inquiry types are transactions initiated by the user which provide information but don't update the internal files.
- Logical internal file types are standing files used by the system.
- External interface file types allow output and input that may pass to and from other computer.

Function point analysis recognizes that the effort required to implement a computer based information system relates not just to the number and complexity of the features but also to the operational environment of the system.

Unadjusted function point: The sum of all occurrences and all the occurrences are computed by multiplying each function count with a weighing and then adding all the values.

$$UFP = FC_1 * W_1 + FC_2 * W_2 + \dots\dots\dots + FC_n * W_n$$

Value Adjustment factor: degree of influence from 0 to 5

"0" no influence

"1" incidental

"2" moderate

"3" average

"4" significant

"5" essential

The constant value in it the equation and the weighing factor are determined empirically:

$$VAF = \text{total degree of influence} * 1\% + .65$$

On the whole,

$$FP = UFP * VAF$$

Function point computes the following important metrics:

Productivity: FP/person-month

Quality: Defects/FP

Cost: Rupees/FP

Documentation: Pages of documentation per FP.

Table 1.1: Albrecht complexity multipliers

External user types	Low	Average	High
External input type	3	4	6
External output type	4	5	7
External inquiry type	3	4	6
LIF type	7	10	15
EIF types	5	7	10

Input: Screen, dialog-box, form from which can be edited or modified by end user in certain ways.

Output: Messages, graphs, screen that are generated. Basically, these can be combine, summarize, process the complex data.

Inquiries: When input are converted into output as result. It is used to retrieve data from the database directly.

Logical Internal Files: It is a flat or table file in a relational database.

External Interface Files: File that are managed by other application having interaction with this application too.

Purpose of Function Point Analysis:

The technique which measures the size of the system according to user are called Function Point analysis. It is varies language to language used. It can only be used during later stage of development life cycle not in early stage.

Components of Function Point Analysis [20]:

FPA is used to measures the size of an application in two areas:

1. Specific User Specification: Under this measurement of the functionality initiated by the user of the application. There are 5 function types which are external input, external output, external enquiries, internal logical files and external interface files.

Function Point = (User Functionality) X (System Characteristics)

2. System Characteristics: The functionality affected by the some characteristics of the system. General characteristics are used to identify functionality**.**

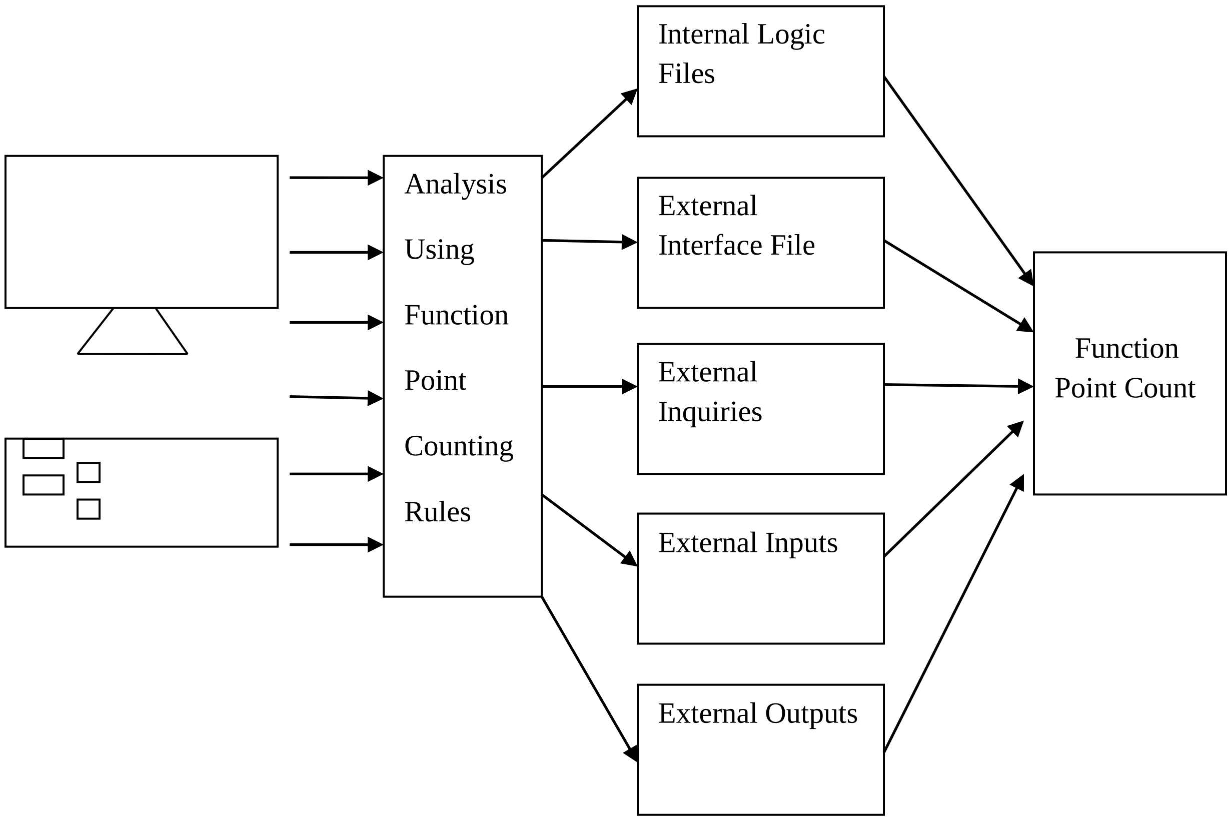


Figure 1.3: Function Point Analysis Estimation

USE CASE ANALYSIS [2]: if efforts are to be estimated during early stage of the project in that scenario Use Cases are used in which problem domain, size and its architecture should be known in advance. It is a software sizing and estimation method which is based on Use Case Points. It is calculated by analyzing the use case of the system. Its steps are:

- Classify in actors as simple, average or complex.

- A simple actor defined Application Programming Interface (API).
- An average actor interacts through a protocol such as TCP/IP.
- A complex actor may be a person interacting via GUI or a Web page.

Functions of Use Case Estimation: The number of use case points in a project is a function of the following [16]:

- Number and complexity of use case.
- Number and complexity of the actors.
- Non-functional requirements such as portability, performance, maintainability that are not written as use cases.

Advantages of Use Case Estimation

- Process can be automated when estimating through use case points which save estimating time.
- For Forecasting future schedule set managerial average implementation time per use case point.
- Use case points are pure measure of size

Disadvantages of Use Case Estimation:

- Estimation can be arrived when use cases are written.
- Proper planning is desired due to large unit of work.
- Some of the Technical Factors do not have much impact on overall project.

1.5.1.3 GROUP BASED ESTIMATION [16]**:** It is as accurate as model-based effort estimation. Unstable relationships and information that are important are excluded from this model and for such cases model may use of expert estimation are suggested. It requires experts with relevant experience.

PLANNING POKER [16]**:** it is an estimation technique based on consensus commonly known as scrum poker. Members of the group make assessment by playing numbered cards whose face are hidden. The cards are acknowledged to all group members and then estimates are discussed. By doing this, biasness or partiality can be removed.

Planning poker is commonly used in agile software development with the Scrum and Extreme Programming methodologies.

WIDEBAND DELPHI [16]: it is an estimation technique based on consensus. It is used to estimate various kinds of tasks from statistical data collection results to sales and marketing forecasts.

1.5.1.4 MECHANICAL COMBINATION: This includes average of an analogy based and a Work breakdown structure based effort estimate.

WORK BREAKDOWN STRUCTURE [11]**:** it decomposes project into smaller components.

The element of WBS may be a product, data, service, or any one among them. It provides the framework for estimating the cost and control along with guidance for software development.

1.5.1.5 JUDGEMENTAL ESTIMATION [11]: This is combination of parametric model and group estimation model.

Chapter 2

REVIEW OF LITERATURE

Misha Kakkar (2014) reviewed techniques and models of effort estimation. Comparison among several approaches is being done and the technique that produces the most accurate result serves as a measure of selection. They specified in there paper that every technique has its own merits and demerits. They suggested in their paper that there is no single technique that can run away from all the shortcomings and can be globally accepted, so the future work suggested in her paper is hybridization of several approaches as an alternative to produce realistic estimates.

Adanma C. Eberendu (2014) proposed hybrid model to enhance the accuracy of the estimation technique. As stated by the author that no single technique are sufficient that can do away with all the shortcomings. Author specified in their paper that is important to determine how much effort is required to complete the software project on-time. Hence, hybridization of more than one technique can give us more accurate estimate that can be helpful to avoid over-estimation or under-estimation of effort.

Derya Toka and Oktay Turetken (2013) analyzed the accuracy of parametric software estimation models. In their paper, they compared four parametric software estimation models in term of their effort and duration prediction accuracy. 51 real project data are used to analyze abilities of the models which compare with the actual effort and duration values. The results of the models that are investigated are par on accuracy. The future work suggested by them can be considered as incorporating historical data for adjustment purpose to have more insight into capabilities and strength of these methods and tools.

Aihua Ren and Chen Yun (2013) advised how to choose estimation methods after conducting research on effort estimation methods and they analyzed the advantages and disadvantages of parametric model. Different parametric model has been estimated under author research paper like Wideband, COCOMO, SLIM, SEER-SEM. Among which different models are being compared on the basis of their advantages and disadvantages.

D. Manikavelan and Dr. R. Ponnusamy (2013) described differential evolutionary algorithm while finding accurate software cost estimation which are implemented using expert judgment to find the accuracy of software cost estimation. They enhanced the expert judgment technique as the name implies that the expert are used in solving problems by judging previously developed projects with the proposed one and for that they used DE algorithm in order to get accurate results. As specified in there paper, that the project is implemented as an open source tool which are used for estimating the cost of the software in future.

Poonam Pandey (2013) analyzed the algorithmic techniques. She tried to analyze the merits and demerits of every technique. As stated in her paper that there are no formal rule of thumb for determining the actual effort which is required for completing a project. The most important thing in the estimation is the datasets of the current and future projects which are required during evaluation of estimation method. Not even a single of the factor that affects the project cost and development should be ignored while estimating the software development cost.

Mohammad Azeh (2013) investigates how Use Case Point estimation model are applicable model to global software project development. He analyzed the potential of use case point estimation model for global projects and used it as a basis to discuss three factors that are proposed by him which are Global team trust, Global team composition and culture value. As stated in the paper it help in managing the global software project development.

Manpreet Kaur and Inderpreet Walia (2013) propose combination of COCOMO model and Function point to produce less effort than COCOMO. Estimating efforts accurately determines whether the development of software is failure or success. As stated in paper that among all the models effort provided by the COCOMO are close to actual effort. In their paper, they are proposing a hybrid model of COCOMO and function point that produces estimated efforts less than COCOMO and function point alone.

Tripo Matijevic, Ivana Ognjanovic and Ramo Sendelj (2012) authors presented in their paper, the possible extension of function point analysis, which is widely, used Functional

Scaling Methods when we consider enhancing the processing of users non-functional requests. The first attempt of this paper is to allow users to define different kinds of input parameter for estimation of complexity of software products. They also analyzed variety of prioritization methods. The limitation of their approach is that as the number of requirements increases, it becomes harder for the user to select best tactics with inability to handle conditional requirements.

Chetan Nagar and Anurag Dixit (2012) combine the Use Case point and COCOMO. They forecast the Line of Code with the help of Use Cases. Explained by them that Use Case used in the method must be more specific not more generalized .The Use Cases gained wide popularity in software effort estimation. Results obtained using use cases are widely applicable. A strong monitoring policy is always required to make estimation as a success. They have to make a check list with the date of completion and must follow the checklist. If work is not done on the time some necessary action must be taken to compensate the deviation. To estimate the KLOC divide the project into module and module into the sub module until we are able to estimate the KLOC. Use Case Point shows the functional requirement of the system .So it is one of the good way of estimation. They also tried to illustrate that how we can combine Use Case and KLOC.

Juha Koskenkylä(2012) focuses towards cost factor as evaluation of software cost estimation techniques are used in some techniques. Cost estimation techniques are suitable and applied for the estimation of the software but they need more effort to be apply by software engineers in which experience of the engineers matters a lot. All technique requires data from previous projects to make analysis for their new project.

H. Azath and R.S.D Wahidabanu (2011) studied basis for the improvement of software effort estimation research through a quality attributes with COCOMO. The classification of software system for which the effort estimation is to be calculated based on COCOMO classes. They used quality assurance ISO 9126 quality factors for this purpose and function point metric are used as weighing factor. COCOMO model is an estimation model of software development. The model uses regression formula to estimate cost using historical data with present and future characteristics. COCOMO starts estimate from the design phase to integration phase of cost and schedule of the project. But separate estimation model should be required for remaining phase. It is not a realistic perfect model because assumptions made at early phase may vary with time progress. A new estimate may show over budget or under budget for the project when cost of the project is revised. This may lead to a limited development of the system. So COCOMO model is not accurate.

G. Stark (2011) According to the author, objective and requirement of business can be achieved through proper measure of size of product. Size basically used to estimate quality of the product, its duration and cost of the software product. Proper technical and business estimates can fulfill the business objective. Timely completion of the project is highly desirable. It is important to measure the effort, duration and quality of the product prior to its release.

Wei Lin Du, Danny Ho (2010) Accurate effort estimation is a essential part of software projects. Accurate effort estimation is base of effective development. Several techniques and models have been introduced by researchers still effort estimation is challenging to do. Earlier researchers developed computing framework. Author aim is to predict the performance of neuro-fuzzy model for the estimation of the software using another model SEER-SEM. Hybridization of fuzzy-logic with SEER-SEM has been proposed by author in his paper. According to author the proposed model has advantage of integration of knowledge, reduced sensitivity, efficient generalization. Its performance evaluated on the basis of the data gathered through several projects.

H Leung and Z Fan (2002) Effort estimation is the process of forecasting the cost which is required to develop the software. Several estimation models have proposed over decades. Research paper by author introduces cost estimation model of recent advances in the software engineering field. Most of the models rely on input based on size of the software. Author provide some of the size metrics then focuses on the estimation model that have been proposed by him and model that are used successfully. According to the author, Models have two main categories that are: algorithmic and non-algorithmic of which both have their advantages and disadvantages. Accuracy is the prime concern while selecting the effort estimation model. Paper describes performance and description of several new models.

Jørgensen, M. Shepperd, M (2007) author provides overview of how to improve software estimation process through a systematic review of the previous paper. 304 cost estimation paper and 76 journals are reviewed by the author and classify then paper according to research topic, approach study context and data set. Library based on web of cost estimation paper are also reviewed and analyzed.

Juan J.Cuardrado Gallego (2007) parametric effort is based on mathematical relationship. Poor adjustment and tailored accuracy are result of data from multiple software projects. To get rid from this problem one major solution is setting mathematical equation that is obtained from historical project dataset. Clusters used as a tool that provide more accurate models. Case study has been used by the author to describe the tool, result and process that are available as historical dataset ISBSG. Results reveal that the single estimation model is much better than the extension of existing model.

Briand, L. C. and I. Wieczorek (2002) categorizes estimation approaches under three main categories as Expert estimation, Formal estimation and combination based estimation. Author further explain number of different models that fall under these categories. As he explained expert estimation include Project Management Software, Planning Poker, Wideband Delphi. ANGEL, Weighted Micro Function Point, COCOMO, SLIM, SEER-SEM, Function Point Analysis, Use case analysis fall under the category of Formal Estimation Model whereas Average of an analogy based and a Work breakdown structure are fall under combination Based Estimation.

Putnam, Lawrence H. (1978) empirical models works on effort, size which are fitted into a curve based on the data provided**.** Prediction of effort is made on the basis of size and effort for which equations are used. If software project need to be finished on time and effort are prerequisite factor. To meet this requirement Putnam introduces algorithmic model popularly known as SLIM (Software Lifecycle Management). It is considered as one of the earliest parametric model that is widely used in software industry. SLIM based on linear regression technique by collecting data from past projects. The Rayleigh curve function is used to describe the time and effort required to complete a software project of specified size. It is used to plot effort as a function of time. It represents the estimated total effort to complete the project at some time.

Chapter 3

PRESENT WORK

Effort estimation means to estimate the efforts according to prospects stakeholders before project is being implemented. With the size estimation we use to measure the project size, usually in lines of code or equivalent. Software is a product without physical existence and its main cost is the design and development of the product.

The effort is measured in term of man-month or year. There are many technique that estimate the efforts of software development but no single technique is sufficient that rum away all the shortcomings. However, it is suggested that the models should be used in pair to estimate the efforts accurately.

3.1 PROBLEM FORMULATION

Measure of work involved in completing a project is called the size of the project [19]. The size of a project is obviously not the number of bytes that the source code occupies, neither is it the size of the executable code. The project size is a measure of the problem complexity in term of the effort and time required to develop the product. Two popular metrics used to measure the size are Source Lines of Code and Function Point.

The **Source Lines of Code** (SLOC) count is usually for executable statement [20]. It is a count of instruction statements. Because the SLOC counts represent the program size and complexity, it is not a surprise that the more lines of code there are in a program; the more defect rate is expected. The SLOC measure suffers from various types of disadvantage like [19]:

- No Precise Definition.
- Difficult to estimate at start of the project.
- Only a code measure.
- Programmer dependent.
- Does not consider code complexity.

Hence, these are to a great extent corrected in the Function Point measure.

Function point is most accepted technique for effort estimation and is based on the functional size. The lines of code are replaced by Allan Albrecht Function Points (FP) as unit for measuring the project size. The Function Points measure the size of the software independently of the technology and the language used to code the programs. It involves the transition from the size oriented metrics to the functionality oriented metrics. The productivity to develop will be countered as FP per man-month.

The important aspect is to convert the size estimate into cost estimate using at least two of the existing technique.

Software Lifecycle Management (SLIM), the base of the software cost estimation was established by Lawrence H. Putnam and Ann Fitzsimmons. SLIM based on linear regression technique by collecting data from past projects. The Rayleigh curve function is used to describe the time and effort required to complete a software project of specified size [13]. It is used to plot effort as a function of time. It represents the estimated total effort to complete the project at some time.

COCOMO model is an estimation model of software development. The model uses regression formula to estimate cost using historical data with present and future characteristics. COCOMO starts estimate from the design phase to integration phase of cost and schedule of the project. But separate estimation model should be required for remaining phase. It is not a realistic perfect model because assumptions made at early phase may vary with time progress. A new estimate may show over budget or under budget for the project when cost of the project is revised. This may lead to a limited development of the system. So COCOMO model is not accurate.

The Efforts are directly proportional to size of the Project i.e. SLOC. Equation for SLIM is [21]: $K = (LOC/(C*t^{4/3}))*3.$

Where,

K is the total life cycle in the working year.

t is development time.

C is the technology constant.

The value of technology constant range from 610 to 57314.

While at the same time COCOMO, involves with the equation:

Effort Applied (E) = $a_b(KLOC)^{b_b}$ [person-months].

As mentioned above, we estimate the effort of the software with the help of source line of code because effort and size both are directly proportional to each other. If project is large, KLOC properly work with Function point analysis. Hence, Hybridization is accomplished by the use of a set of Unadjusted Function Points UFP to Source Line of Code SLOC.

To formulate problem in a dignified manner NASA93 dataset has been selected in which three models will be analyzed on the basis of their Magnitude Relative Error (MRE). Comparison and Analysis of all the models are done of which two models are COCOMO II & SLIM and third one is proposed model called Hybrid estimation which is combination of SLIM and Function Point. Dataset consist of 20 individual projects with the values of cost drivers and KLOC. We have randomly selected 10 projects of which one project is selected at a time to calculate effort and graphically plot them. Performance of all 20 projects is also analyzed accumulatively.

3.2 OBJECTIVE

The main objectives of the study are:

- **Proposing a new hybrid estimation technique**

  To make good software management decision, it is important to determine how much effort is required to complete the software project on-time. There are many software estimation techniques available but not even a single estimation technique is sufficient that can do away with all shortcomings. Hence, hybridization of more than one technique can give us more accurate estimate that can be helpful to avoid over-estimation or under-estimation of effort.

- **Implementation of proposed model to reduce error and enhance accuracy**

  Accuracy is prime concern of software project management. A software project is considered to be accurate if it provides consistent result. The proposed estimation technique will try to reduce the Mean Magnitude of Relative Error (MMRE) and Magnitude of Relative Error (MRE) of the project to enhance the accuracy. Therefore, the reliable estimation technique provides failure free operation and reduces error that will enhance the accuracy.

3.3 METHODOLOGY

Method is depicted through the following figure:

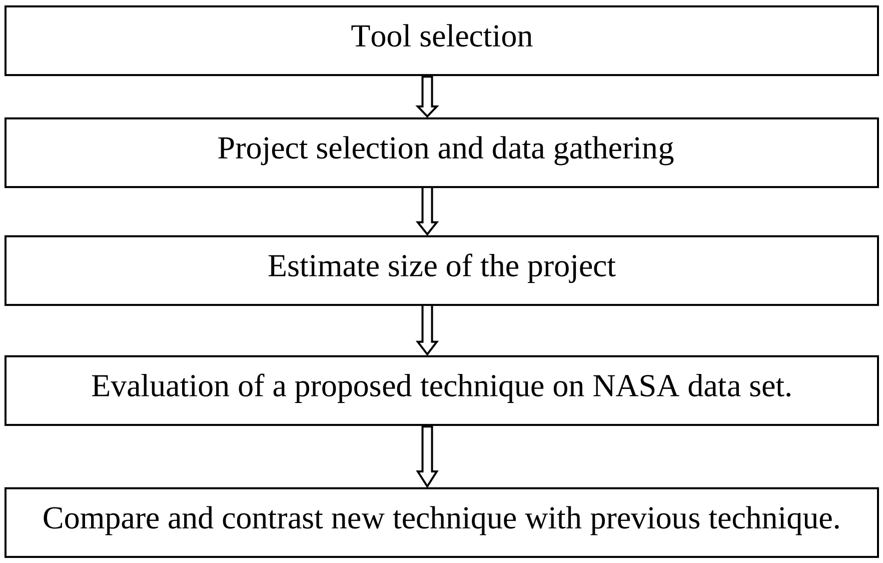

Tool selection
Project selection and data gathering
Estimate size of the project
Evaluation of a proposed technique on NASA data set.
Compare and contrast new technique with previous technique.

Figure 3.1: Research Methodology

I. Tool Selection: After analyzing several parametric models, the following two models have been selected for the comparative analysis: COCOMO II, SLIM. These two models are selected due to the widely use and approval in the software industry for the development of business application in which SLIM is used for commercial purpose whereas COCOMO II is a free online tool for estimation.

- **COCOMO II:** This model preserves the originality of COCOMO model i.e. openness of the COCOMO. The three stages of COCOMO II are:

  Firstly, it follows the prototyping model with the help of application model capability. Secondly, it occupies investigation of architectural alternatives or incremental development strategies. Last, when project is ready to develop then it should have life-cycle architecture, which provide more exact information on cost driver inputs and gives more accurate cost estimates[10].
- **SLIM:** Software Lifecycle Management, the base of the software cost estimation was established by Lawrence H. Putnam and Ann Fitzsimmons. SLIM based on linear regression technique by collecting data from past projects. The Rayleigh curve function is used to describe the time and effort required to complete a software project of specified size. It is used to plot effort as a function of time. It represents the estimated total effort to complete the project at some time.
- **MATLAB:** is a numerical tool for manipulating matrix, implementing algorithms etc. It is the most widely used tool by the programmers in software industry as it can interface with any other language. It is a fourth generation programming language which is developed by MathWorks. With the help of the MATLAB we can perform the functionalities such as algorithm implementation, can create user interface and interfacing with programs written in other languages.

  MATLAB is a MATRIX Laboratory. It is an interactive program which provides numerical computation and visualization of data. With the help of its programming capabilities it provides tool which is useful for all areas of science and engineering.

  Image Processing Toolbox provides a comprehensive set of reference-standard algorithms, functions, and apps for image processing, analysis, visualization, and algorithm development. We can perform image enhancement, image deploring, feature detection, noise reduction, image segmentation, geometric transformations, and image registration. Many toolbox functions are multithreaded to take advantage of multicore and multiprocessor computers. Image Processing Toolbox chains a various set of image and their types, together with high forceful range, embedded ICC profile, topographic and gig pixel resolution. Visualization functions permit you searches an image, inspect a region of pixels, create histograms, adjust the contrast and influence regions of interest (ROIs). With toolbox algorithms you can restore ruined images, explore shapes, analyze textures, adjust color balance and detect and measure features.

Image Processing Toolbox is included in MATLAB and Simulink Student Version. Its key features are as follows:

- Importing and Exporting Images
- Displaying and Exploring Images
- Post processing Images and Preprocessing
- Analyzing Images
- Image Registration and Geometric Transformations
- Working with Large Images

II. Project Selection & Data Gathering:

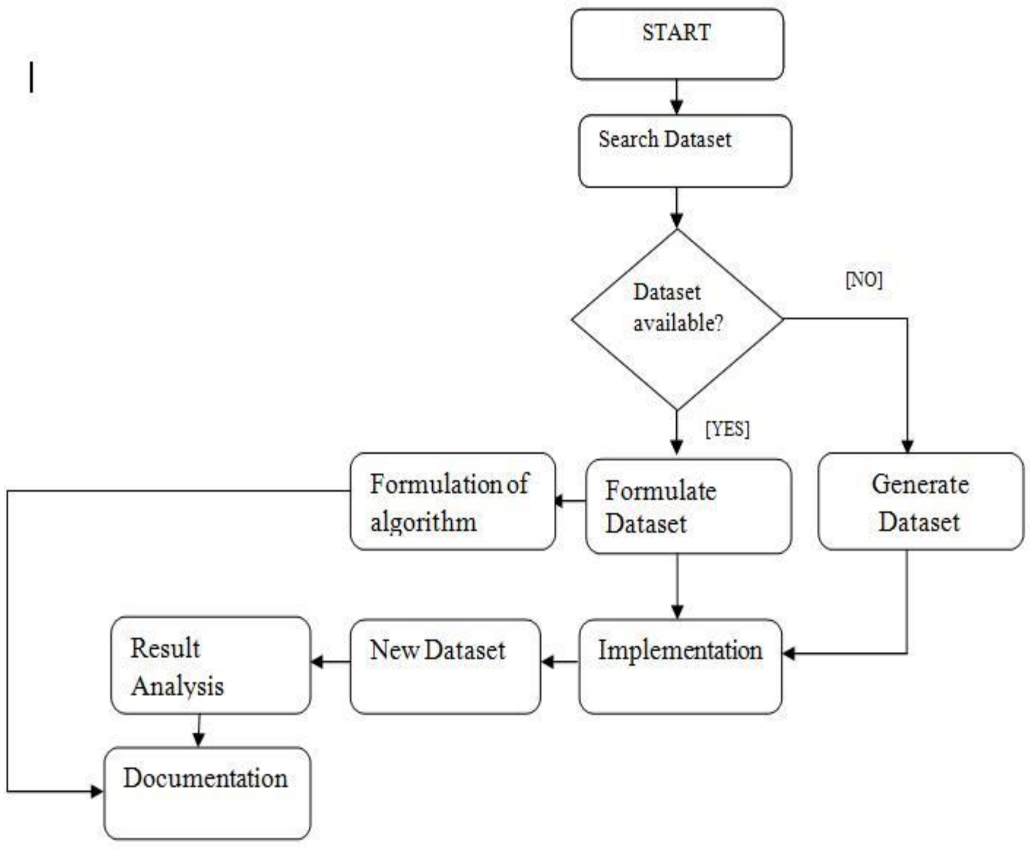


Figure 3.2: Selection of Dataset

III. Estimate size of the Project: SLIM model is based upon the line of code. As complexity increases, it is difficult to estimate the effort with more complexity. To get rid from this difficulty a new method for estimating the effort is being proposed that estimate the effort more accurately. A new proposed technique is being developed to enhance the accuracy of the SLIM by adjusting the value of the cost driver.

For adjusting the cost driver, they are classified into three groups:

- Pessimistic group: In this group, the value of cost driver is directly proportional to the efforts. As the value of the cost driver increases, effort also increases and so the errors. This group consists of seven cost drivers among total of fifteen from CD8 to CD14.
  Pessimistic Group: {ACAP, PCAP, AEXP, MODP, TOOL, VEXP, LEXP}
- Independent group: This group consists of the cost driver in which the effort does not depend upon the values of cost driver neither have it affected the efforts which lead to increase or decrease of its values. There is only one independent cost driver.
  Cost driver: {SCED}
- Optimistic group: In this group, the values of cost driver are inversely proportional to the efforts calculated. As the cost drivers increases, the effort decreases and so the errors. It consist of 7 cost drivers:
  Optimistic group: {DATA, TURN, VIRT, STOR, TIME, RELY, CPLEX}

Algorithm

- In this algorithm, we are proposing enhancement in the SLIM model for effort estimation. The new algorithm is useful for selecting the most relevant features of effort estimation and cost drivers which has significant influence on the SLIM model for effort estimation where cost drivers are the effort multiplier.
- The LOC of SLIM model is directly proportional to effort estimation which will affect the effort estimation directly, some time it will be overestimated or underestimated due to vagueness in the cost drivers .
- It is important to stress that uncertainty at the input level of the SLIM model yields uncertainty at the output too. This becomes obvious and, more importantly, according to the effort, Cost drivers are separated into three groups such as Optimistic, Pessimistic and independent group.

- **Pessimistic Group:** The group consists of the cost drivers in which the effort is directly proportional to the values of cost drivers. I.e. the corresponding cost drivers support the system by reducing the required effort by increasing their values.

  PG = {CD1, CD2……CD7)

  Pessimistic Group are as follows:

  ACAP - Analyst Capability

  PCAP - Programmers capability

  AEXP - Application Experience

  MODP -Modern Programming Practices

  TOOL -Use of software tools

  VEXP -Virtual machine experience

  LEXP -Language experience

- **Independent Group:** The group consists of the cost drivers in which the effort is independent to the values of cost drivers. I.e. the corresponding cost drivers do not affect the effort.

  Cost drivers**:** SCED (Independent Group) **-** Schedule constraint

- **Optimistic Group:** The group consists of the cost drivers in which the effort is inversely proportional to the values of cost drivers. i.e. the corresponding cost drivers supports the system by reducing the required effort by increasing their values.

  OG = {CD9, CD10……CD15)

  Optimistic Group=

  DATA- Database Size

  TURN- Turnaround Time

  VIRT- Machine Volatility

  STOR- Main memory constraints

  TIME- Time Constraints for CPU

  RELY- Required software reliability

  CPLEX- Process complexity

- Effort derived from the proposed model calculates effort which is equal to:

[((m + n) / n * 0.1 + 2.5) * Size * 0.99] 1.266.

Where,

"M" is the sum of cost drives in the set PG i.e. cost drivers which are present in the Pessimistic Group.

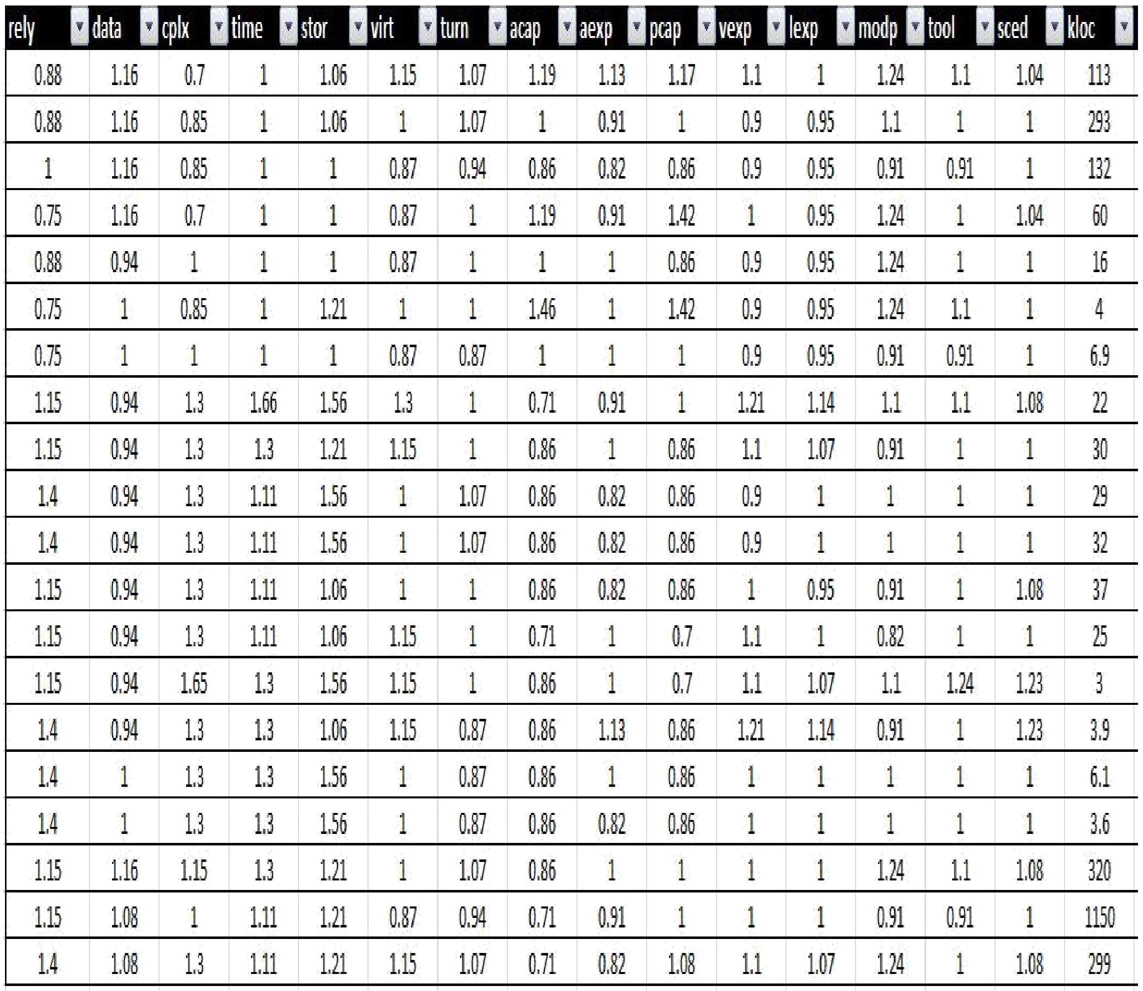
"N" is the sum of cost drives in the set OG i.e. the cost drivers which is present in the Optimistic Group.

1.266 is Empirical Exponential Constant.

0.99 is Empirical Domain Constant.

0.1 And 2.5 is Empirical Adjustment Factor.

The following are the cost driver values in NASA93 dataset on which formula is generated:



rely	data	cplx	time	stor	virt	turn	acap	aexp	pcap	vexp	lexp	modp	tool	sced	kloc
0.88	1.16	0.7	1	1.06	1.15	1.07	1.19	1.13	1.17	1.1	1	1.24	1.1	1.04	113
0.88	1.16	0.85	1	1.06	1	1.07	1	0.91	1	0.9	0.95	1.1	1	1	293
1	1.16	0.85	1	1	0.87	0.94	0.86	0.82	0.86	0.9	0.95	0.91	0.91	1	132
0.75	1.16	0.7	1	1	0.87	1	1.19	0.91	1.42	1	0.95	1.24	1	1.04	60
0.88	0.94	1	1	1	0.87	1	1	1	0.86	0.9	0.95	1.24	1	1	16
0.75	1	0.85	1	1.21	1	1	1.46	1	1.42	0.9	0.95	1.24	1.1	1	4
0.75	1	1	1	1	0.87	0.87	1	1	1	0.9	0.95	0.91	0.91	1	6.9
1.15	0.94	1.3	1.66	1.56	1.3	1	0.71	0.91	1	1.21	1.14	1.1	1.1	1.08	22
1.15	0.94	1.3	1.3	1.21	1.15	1	0.86	1	0.86	1.1	1.07	0.91	1	1	30
1.4	0.94	1.3	1.11	1.56	1	1.07	0.86	0.82	0.86	0.9	1	1	1	1	29
1.4	0.94	1.3	1.11	1.56	1	1.07	0.86	0.82	0.86	0.9	1	1	1	1	32
1.15	0.94	1.3	1.11	1.06	1	1	0.86	0.82	0.86	1	0.95	0.91	1	1.08	37
1.15	0.94	1.3	1.11	1.06	1.15	1	0.71	1	0.7	1.1	1	0.82	1	1	25
1.15	0.94	1.65	1.3	1.56	1.15	1	0.86	1	0.7	1.1	1.07	1.1	1.24	1.23	3
1.4	0.94	1.3	1.3	1.06	1.15	0.87	0.86	1.13	0.86	1.21	1.14	0.91	1	1.23	3.9
1.4	1	1.3	1.3	1.56	1	0.87	0.86	1	0.86	1	1	1	1	1	6.1
1.4	1	1.3	1.3	1.56	1	0.87	0.86	0.82	0.86	1	1	1	1	1	3.6
1.15	1.16	1.15	1.3	1.21	1	1.07	0.86	1	1	1	1	1.24	1.1	1.08	320
1.15	1.08	1	1.11	1.21	0.87	0.94	0.71	0.91	1	1	1	0.91	0.91	1	1150
1.4	1.08	1.3	1.11	1.21	1.15	1.07	0.71	0.82	1.08	1.1	1.07	1.24	1	1.08	299

Figure 4.1: Cost Drivers and their values in NASA93

increase these to decrease effort	acap:	analysts capability
	pcap:	programmers capability
	aexp:	application experience
	modp:	modern programming practices
	tool:	use of software tools
	vexp:	virtual machine experience
	lexp:	language experience
	sced:	schedule constraint
decrease these to decrease effort	data:	data base size
	turn:	turnaround time
	virt:	machine volatility
	stor:	main memory constraint
	time:	time constraint for cpu
	rely:	required software reliability
	cplx:	process complexity

Figure 4.2: Depiction of values according to generality

The above mentioned figure depicting the fifteen cost drivers according to their respective group as they are categories in three groups: Pessimistic, Independent and Optimistic group. In the Pessimistic group, attempt is make to increase the value of the cost driver so as to decrease the effort which leads to reduction in error. In the Optimistic group, as the values of the cost drivers are decreases with respect to that the effort also decreases significantly.

	very low	low	nominal	high	very high	extra high	productivity range
ACAP	1.46	1.19	1.00	0.86	0.71		2.06
PCAP	1.42.	1.17	1.00	0.86	0.70		2.03
AEXP	1.29	1.13	1.00	0.91	0.82		1.57
MODP	1.24.	1.10	1.00	0.91	0.82		1.51
TOOL	1.24	1.10	1.00	0.91	0.83		1.49
VEXP	1.21	1.10	1.00	0.90			1.34
LEXP	1.14	1.07	1.00	0.95			1.20
SCED	1.23	1.08	1.00	1.04	1.10		
DATA		0.94	1.00	1.08	1.16		-1.23
TURN		0.87	1.00	1.07	1.15		-1.32
VIRT		0.87	1.00	1.15	1.30		-1.49
STOR			1.00	1.06	1.21	1.56	-1.56
TIME			1.00	1.11	1.30	1.66	-1.66
RELY	0.75	0.88	1.00	1.15	1.40		-1.87
CPLX	0.70	0.85	1.00	1.15	1.30	1.65	-2.36

Figure 4.3: Productivity range of Cost Drivers

IV. Evaluation of the Proposed Technique on the NASA93 Dataset: A new formula is being derived which calculate the efforts of the individual project selected from the NASA93 dataset. The process of evaluating the effort is carried out by applying certain mathematical formulas. The following metrics are measured for each model and project:

- RE (Relative Error) = (Actual Effort – Estimated Effort) / Actual Effort.
- EE (Estimated Effort) = a * (KLOC) ^ b * EAF.

  Where, EAF is the multiplication of the 15 cost drivers.
- Error = (Actual Efforts – Estimated Efforts).
- MRE(Magnitude of Relative Error): which is used to check the reliability of an estimation :

  MRE = abs (RE).

MRE indicates how accurate estimation is, if the value of MRE is closer to 0% estimation is more accurate. Mean Magnitude of Relative Error will calculated by finding mean value of the Magnitude Relative Error.

V. Compare and Contrast New Technique with Previous Technique: After evaluation of the proposed technique, comparison among the previous technique COCOMO II & SLIM is being conducted with new technique that is called as hybrid estimation technique which is a combination of SLIM and Function Point. Comparison in term of graphical representation clearly demonstrate which of the model gives better effort estimation compared with others.

Chapter 4

RESULT AND DISCUSSIONS

Early estimation of the effort during software development is a key to success. The main activity of the effort estimation is to identify the project variations from the existing project and systematically considered during estimation of the new software. Significant modifications into the existing parameter's can drastically change result that will enhance the accuracy. The proposed software effort estimation technique will provide accurate outcome that enhances the accuracy in formal effort estimation.

Effort, duration and management of man power are prime sources of effort estimation. Over-estimation or under-estimation may frequently change the outcome. A proposed model that will enrich our ability to scientifically estimate software development effort and help to reduce the error injected in existing project. The proposal to hybrid parametric models are expected to deliver failure free estimates and within a specified period of time. This will in turn lead to a reduction of the number of IT-project failures and make better use of scarce financial and human resources.

4.1 Analysis of Outcome

The proposed hybrid parametric models are expected to deliver failure free estimates and within a specified period of time. This will in turn lead to a reduction of the number of IT-project failures and make better use of scarce financial and human resources.

NASA93 dataset has been selected in which three models will be analyzed on the basis of their Magnitude Relative Error (MRE). Comparison and Analysis of all the models are done of which two models are COCOMO II & SLIM and third one is proposed model called Hybrid estimation which is combination of SLIM and Function Point. Dataset consist of 20 individual projects with the values of cost drivers and KLOC. We have randomly selected 10 projects of which one project is selected at a time to calculate effort and graphically plot them. Performance of all 20 projects is also analyzed accumulatively.

At a time one Project is selected among total number of projects.

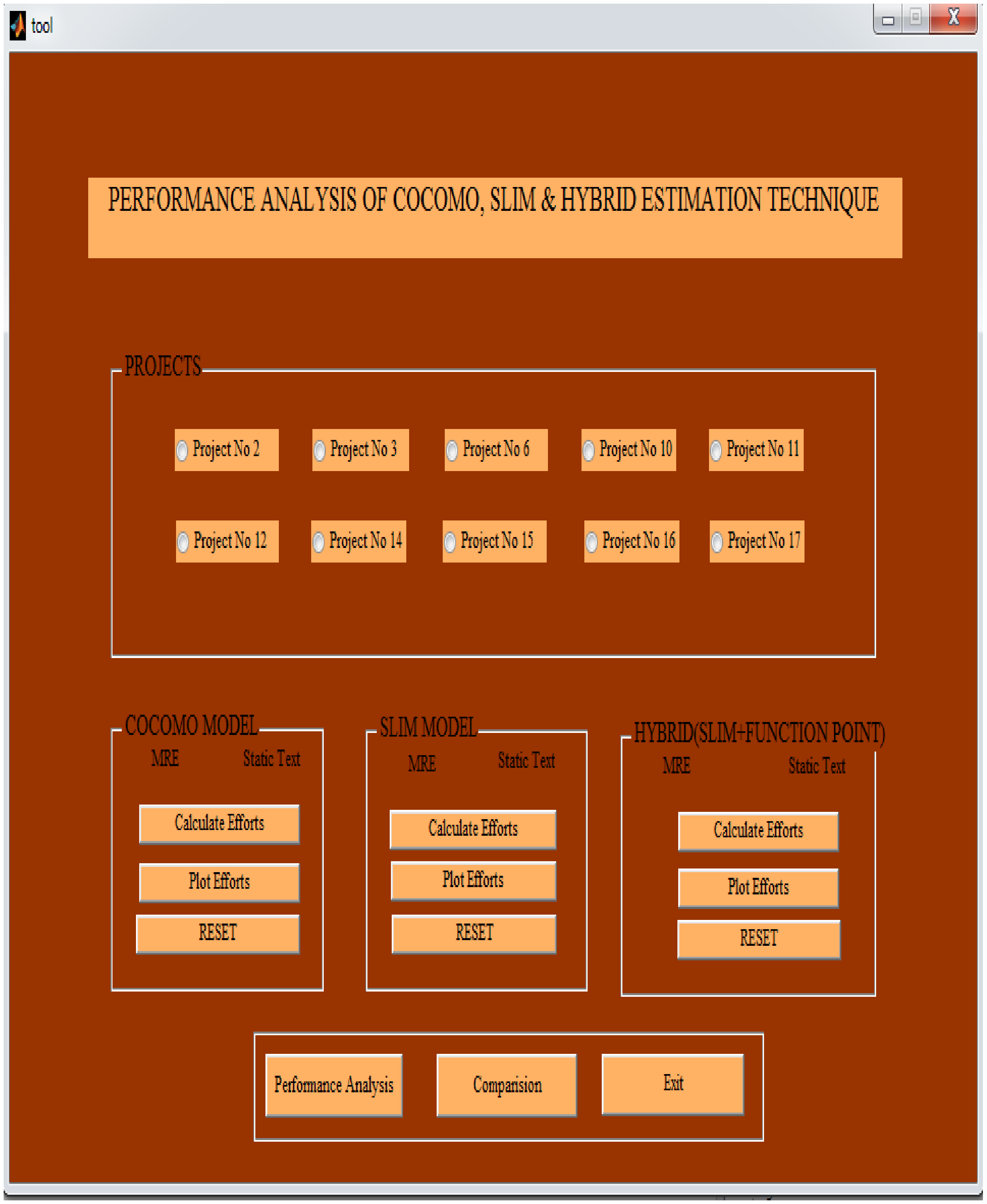


Figure 4.4: Tool Default View

Randomly Project 3 is selected to evaluate different models.

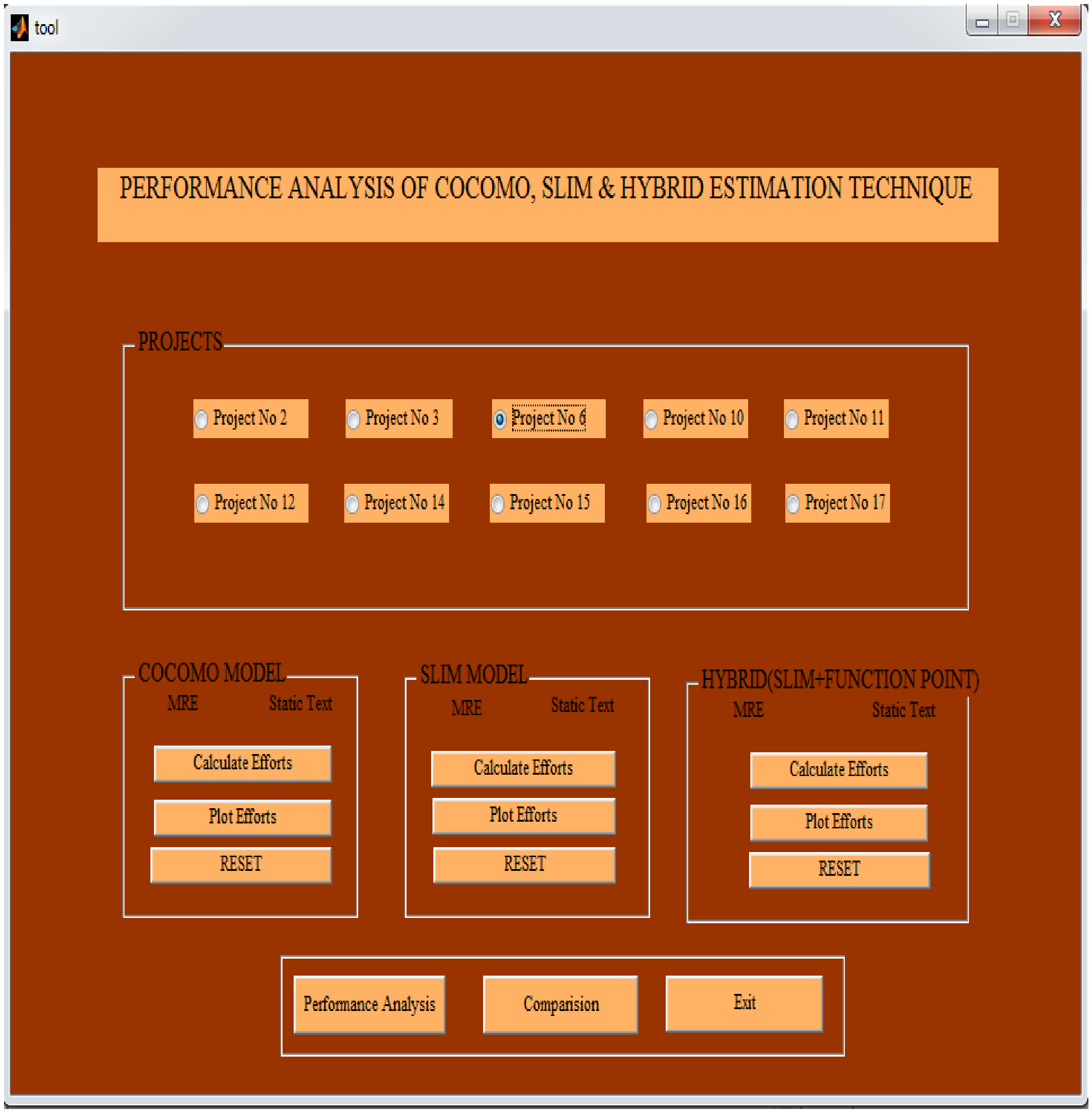


Figure 4.5: Selection of Project

Selection of individual project is demonstrated in above mentioned figure which indicate only one project at once. Here, Project 3 is selected to analyze individual model and to mark the comparison among rest of the models. COCOMO, SLIM and Hybrid estimation model are then individually selected to calculate the Magnitude of Relative Error.

After selecting the Project among the number of projects, choose model to evaluate.

Effort is calculated of selected Project 3 by initiating calculate effort button.

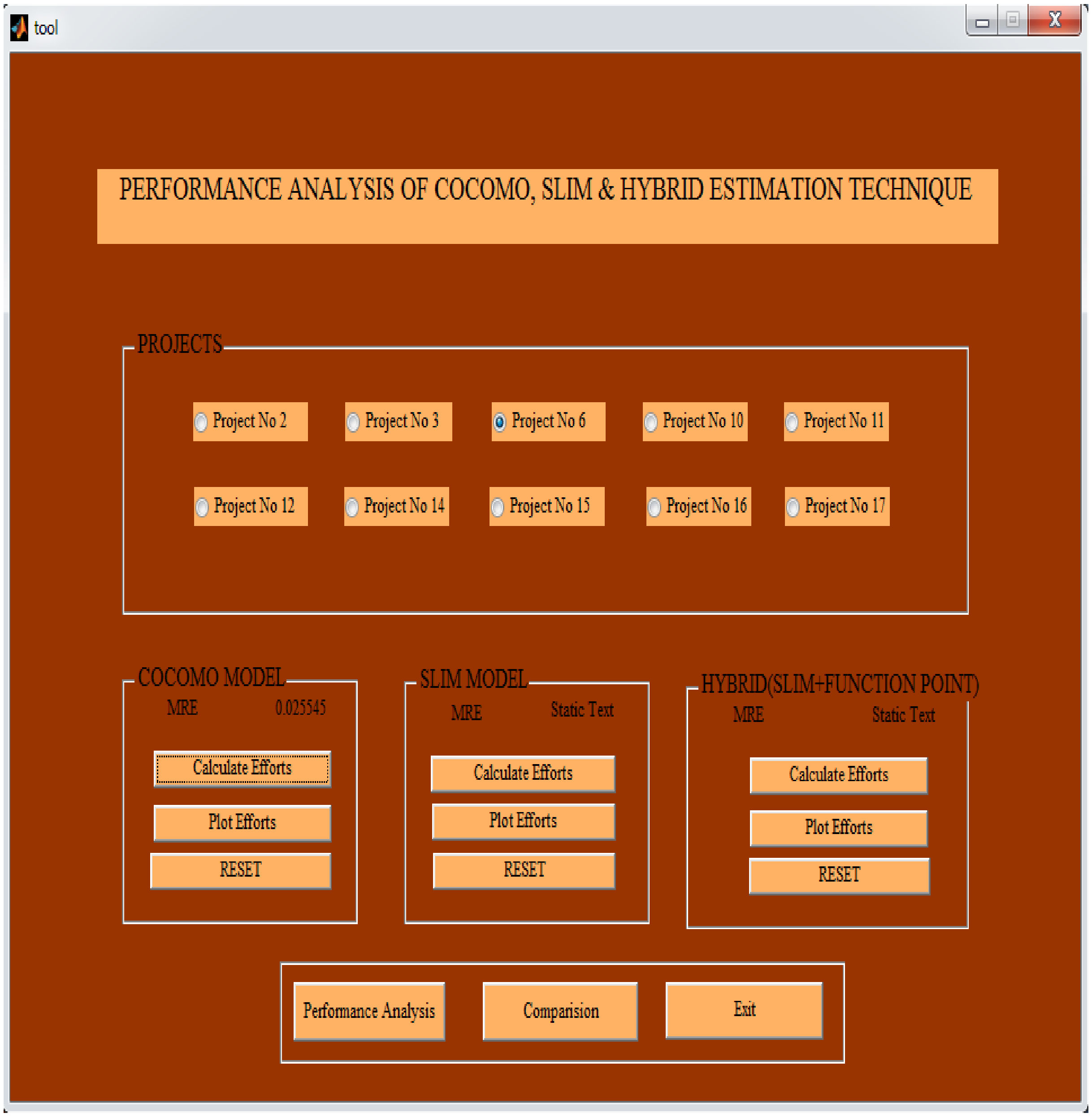


Figure 4.6: Selection of COCOMO Model

COCOMO Model is selected to evaluate the Magnitude Relative Error. As mentioned in the above screen, MRE of COCOMO shows static value as 0.025545 after pressing the button calculate effort. The Plot effort button is used to graphically present the MRE of COCOMO while RESET button is used to refresh the value of COCOMO again.

Moving further to graphically represents the calculated value of COCOMO model

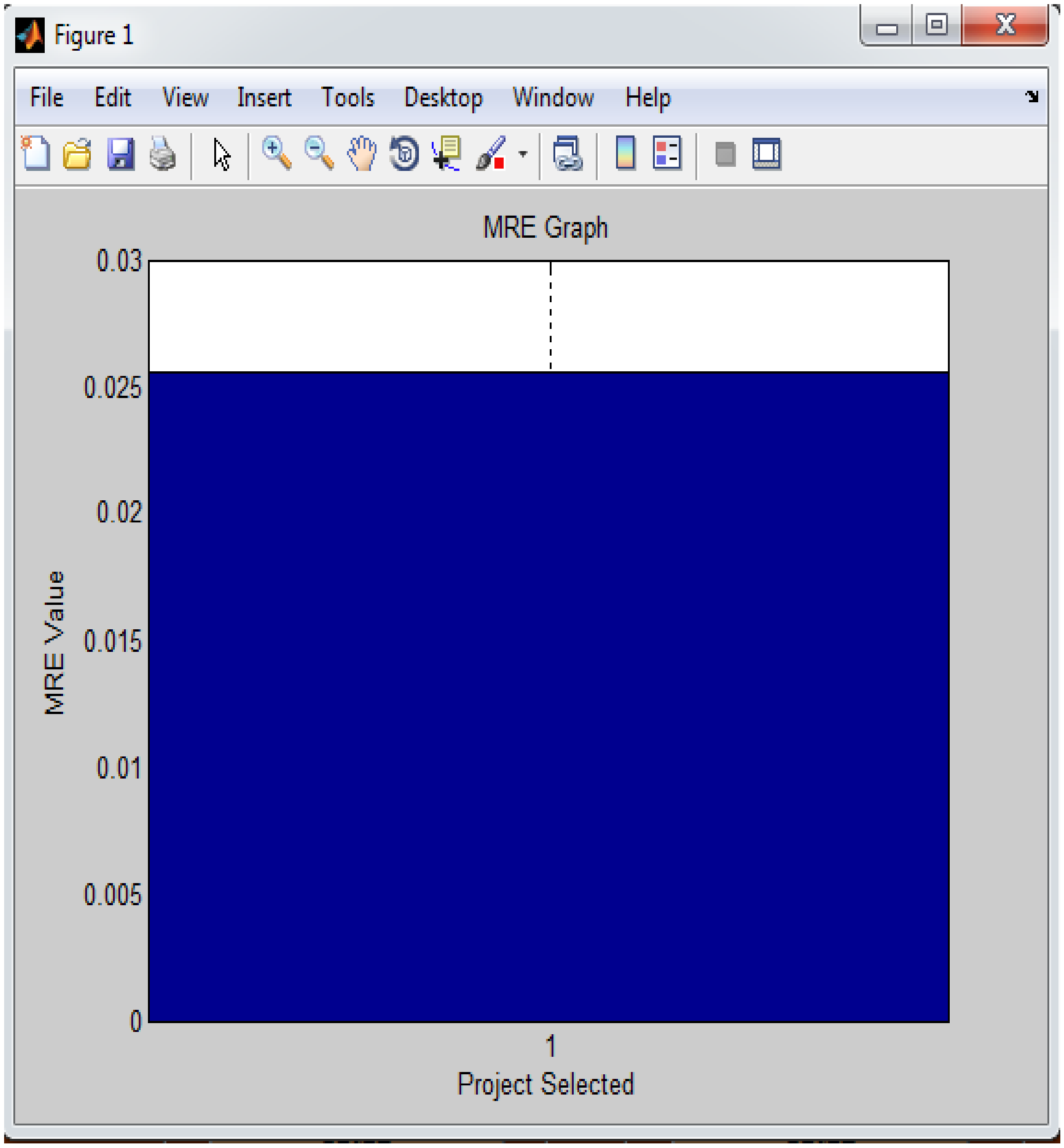
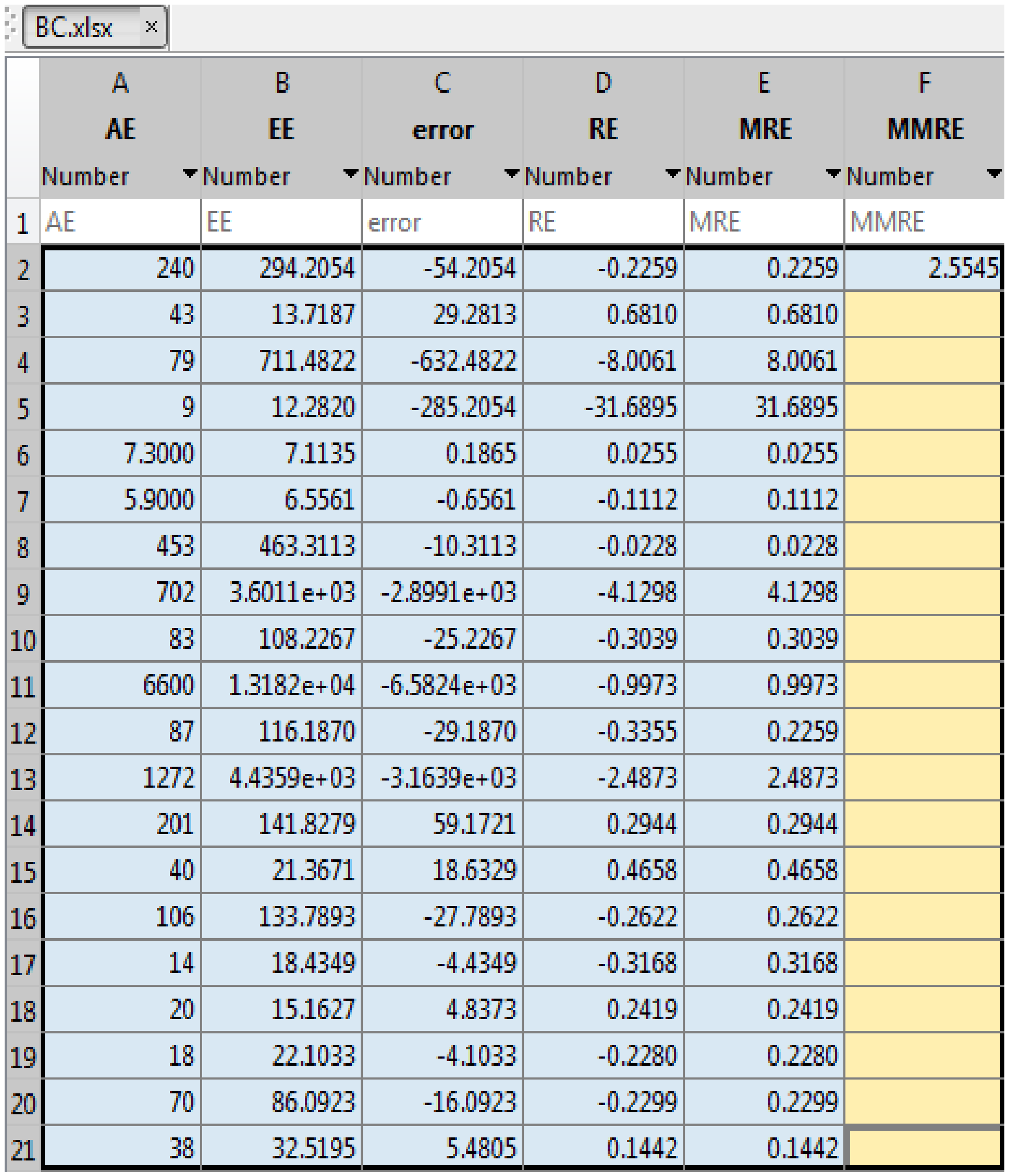


Figure 4.7: Graphically plotting COCOMO

Calculated value of COCOMO model are plotted graphically where X- axis shows selection of the project and Y-axis shows its respective magnitude relative error value. The values are plotted on the basis of the magnitude relative error.


BC.xlsx

	A AE Number	B EE Number	C error Number	D RE Number	E MRE Number	F MMRE Number
1	AE	EE	error	RE	MRE	MMRE
2	240	294.2054	-54.2054	-0.2259	0.2259	2.5545
3	43	13.7187	29.2813	0.6810	0.6810	
4	79	711.4822	-632.4822	-8.0061	8.0061	
5	9	12.2820	-285.2054	-31.6895	31.6895	
6	7.3000	7.1135	0.1865	0.0255	0.0255	
7	5.9000	6.5561	-0.6561	-0.1112	0.1112	
8	453	463.3113	-10.3113	-0.0228	0.0228	
9	702	3.6011e+03	-2.8991e+03	-4.1298	4.1298	
10	83	108.2267	-25.2267	-0.3039	0.3039	
11	6600	1.3182e+04	-6.5824e+03	-0.9973	0.9973	
12	87	116.1870	-29.1870	-0.3355	0.2259	
13	1272	4.4359e+03	-3.1639e+03	-2.4873	2.4873	
14	201	141.8279	59.1721	0.2944	0.2944	
15	40	21.3671	18.6329	0.4658	0.4658	
16	106	133.7893	-27.7893	-0.2622	0.2622	
17	14	18.4349	-4.4349	-0.3168	0.3168	
18	20	15.1627	4.8373	0.2419	0.2419	
19	18	22.1033	-4.1033	-0.2280	0.2280	
20	70	86.0923	-16.0923	-0.2299	0.2299	
21	38	32.5195	5.4805	0.1442	0.1442	

Figure 4.8: MRE & MMRE representation of 20 projects in COCOMO Model

Magnitude of Relative Error is calculated in COCOMO Model by following the evolution process from which we can calculate the Actual Effort, Estimated Effort, Error Relative Error and the Magnitude of Relative Error from which Mean value of Magnitude of Relative Error are termed as MMRE can be calculated.

SLIM Model is selected to evaluate the Magnitude Relative Error.

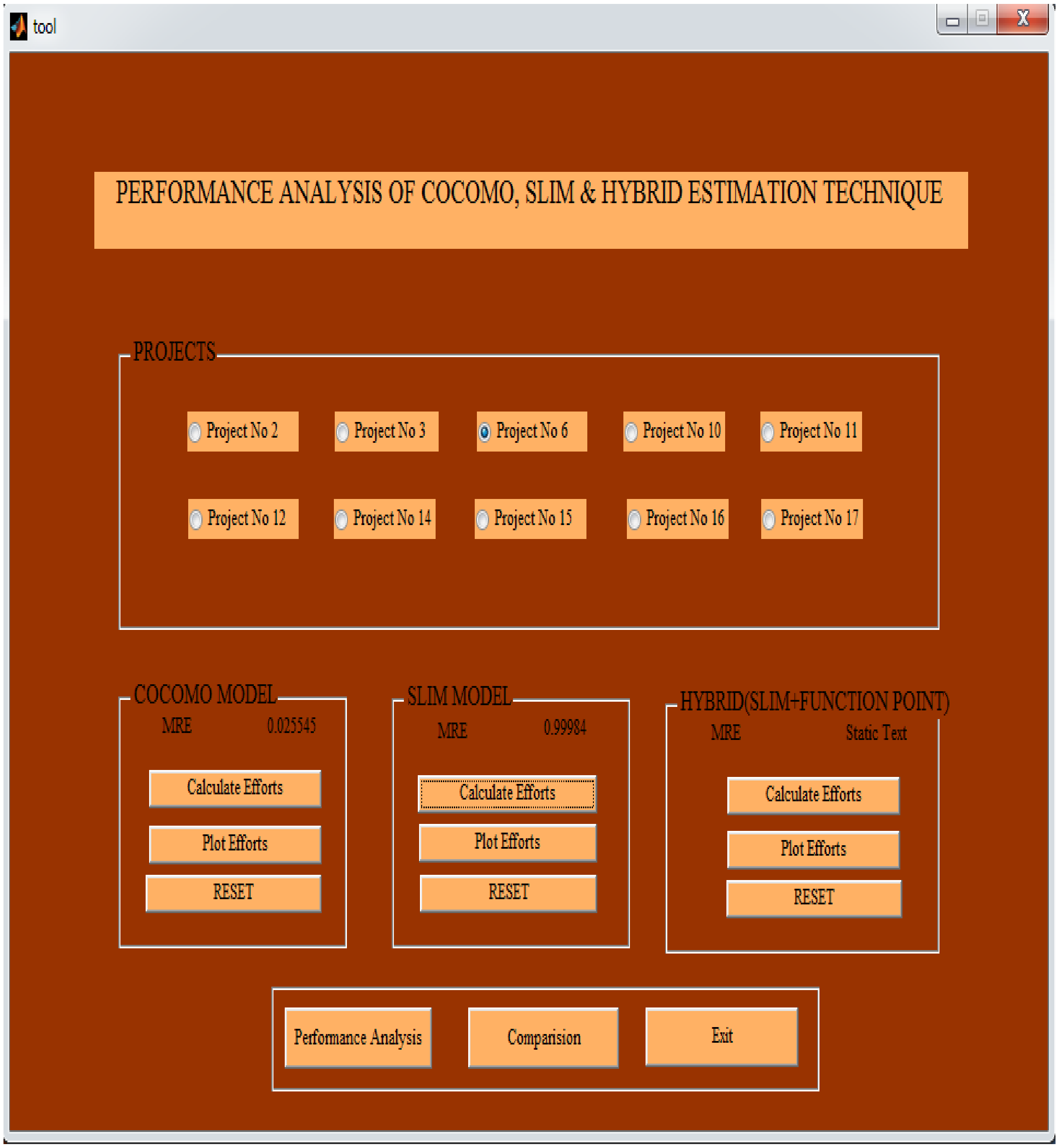


Figure 4.9: Selection of SLIM Model

SLIM Model is selected to evaluate the MRE by pressing the Calculate Effort button whose static value is 0.99995 as depicted in above mentioned figure after pressing the button calculate effort. The Plot effort button is used to graphically present the MRE of SLIM while RESET button is used to refresh the value of SLIM again.

Moving further to represent the SLIM model graphically.

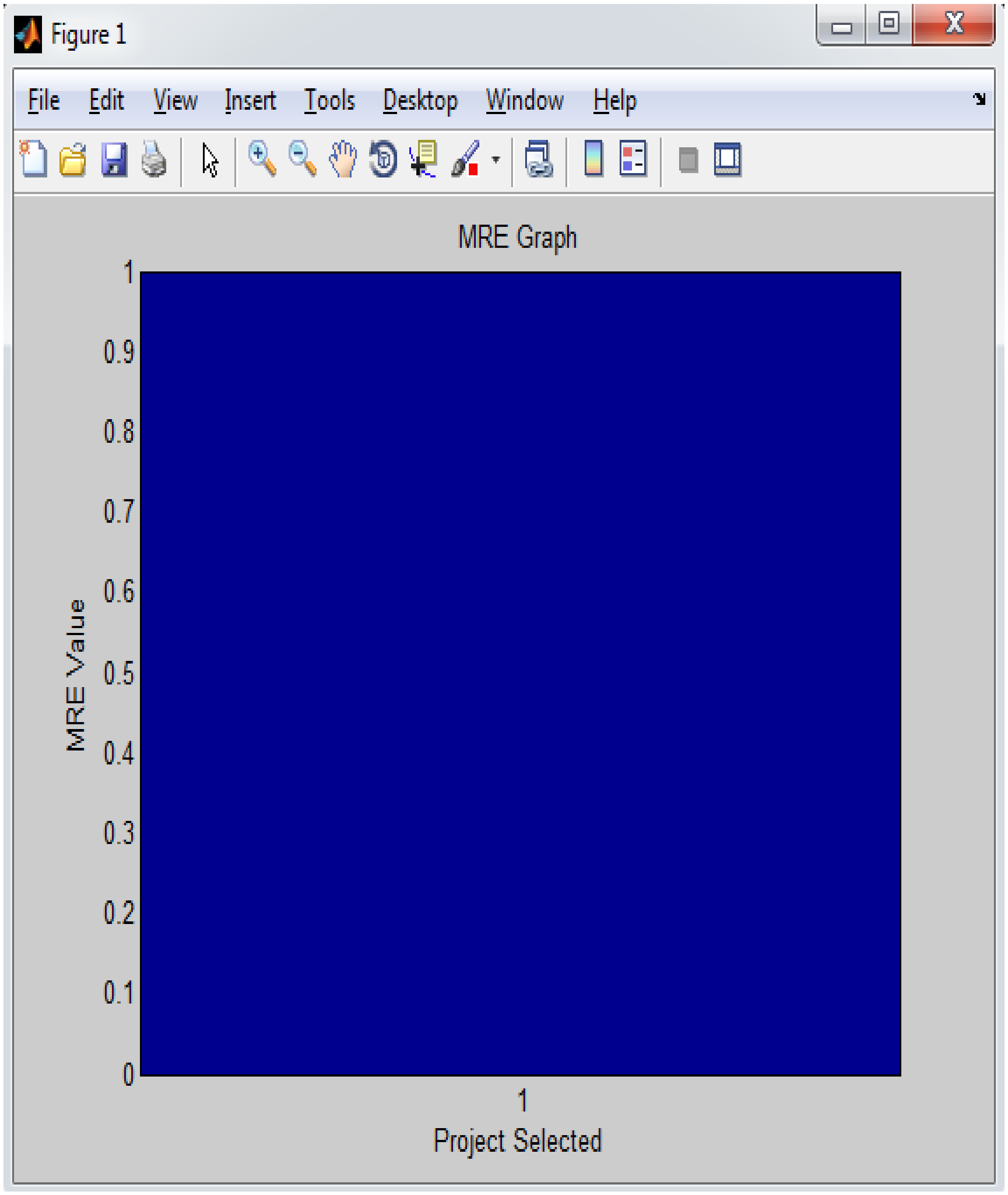
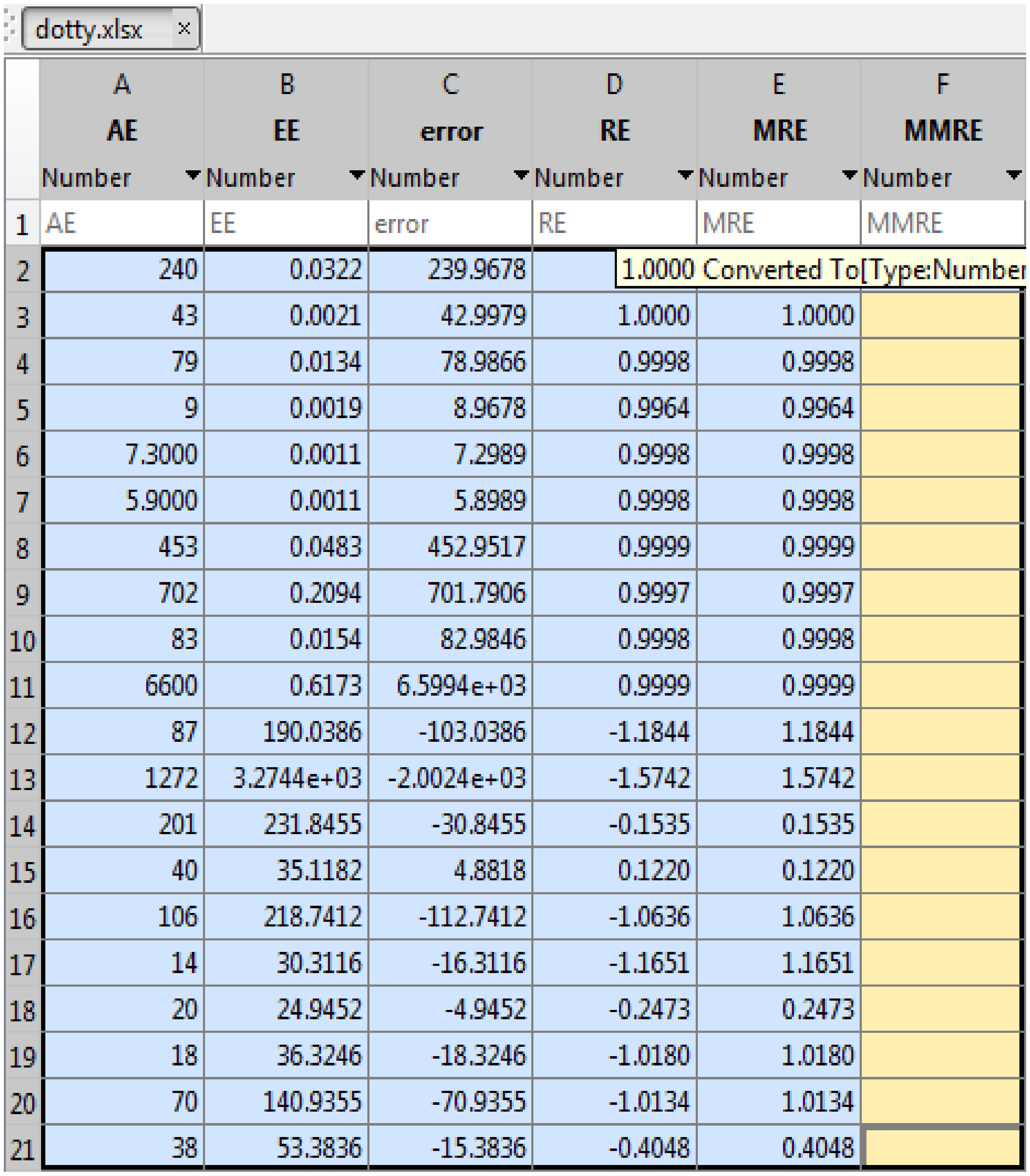


Figure 4.10: Graphically Plotting SLIM

Calculated value of SLIM model are plotted graphically where X- axis shows selection of the project and Y-axis shows its respective magnitude relative error value. . The values are plotted on the basis of the magnitude relative error.



dotty.xlsx

	A	B	C	D	E	F
	AE	EE	error	RE	MRE	MMRE
	Number	Number	Number	Number	Number	Number
1	AE	EE	error	RE	MRE	MMRE
2	240	0.0322	239.9678		1.0000 Converted To[Type:Number	
3	43	0.0021	42.9979	1.0000	1.0000	
4	79	0.0134	78.9866	0.9998	0.9998	
5	9	0.0019	8.9678	0.9964	0.9964	
6	7.3000	0.0011	7.2989	0.9998	0.9998	
7	5.9000	0.0011	5.8989	0.9998	0.9998	
8	453	0.0483	452.9517	0.9999	0.9999	
9	702	0.2094	701.7906	0.9997	0.9997	
10	83	0.0154	82.9846	0.9998	0.9998	
11	6600	0.6173	6.5994e+03	0.9999	0.9999	
12	87	190.0386	-103.0386	-1.1844	1.1844	
13	1272	3.2744e+03	-2.0024e+03	-1.5742	1.5742	
14	201	231.8455	-30.8455	-0.1535	0.1535	
15	40	35.1182	4.8818	0.1220	0.1220	
16	106	218.7412	-112.7412	-1.0636	1.0636	
17	14	30.3116	-16.3116	-1.1651	1.1651	
18	20	24.9452	-4.9452	-0.2473	0.2473	
19	18	36.3246	-18.3246	-1.0180	1.0180	
20	70	140.9355	-70.9355	-1.0134	1.0134	
21	38	53.3836	-15.3836	-0.4048	0.4048	

Figure 4.11: MRE & MMRE representation of 20 projects in SLIM Model

Magnitude of Relative Error is calculated in SLIM Model by following the evaluation process from which we can calculate the Actual Effort, Estimated Effort, Error Relative Error and the Magnitude of Relative Error from which Mean value of Magnitude of Relative Error are termed as MMRE can be calculated.

Hybrid Estimation Technique is evaluated which is a combination of SLIM and Function Point.

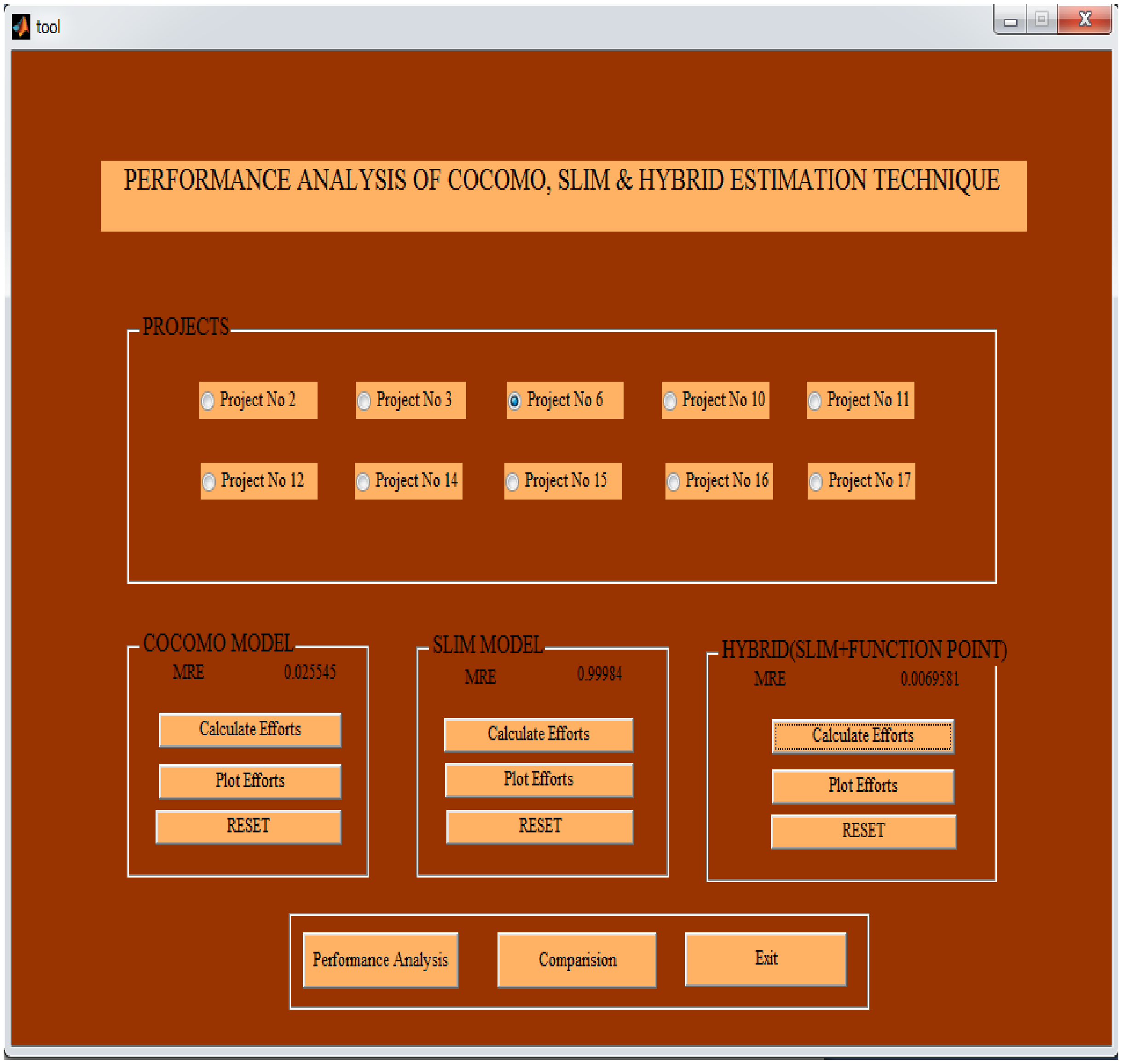


Figure 4.12: Selection of Hybrid estimation Model

Hybrid Estimation Model(SLIM + Function Point) is selected to evaluate the MRE by pressing the Calculate Effort button whose static value is 0.0069581 as depicted in above mentioned figure after pressing the button calculate effort. The Plot effort button is used to graphically present the MRE of SLIM while RESET button is used to refresh the value of SLIM again.

Graphical representation of Hybrid Estimation model.

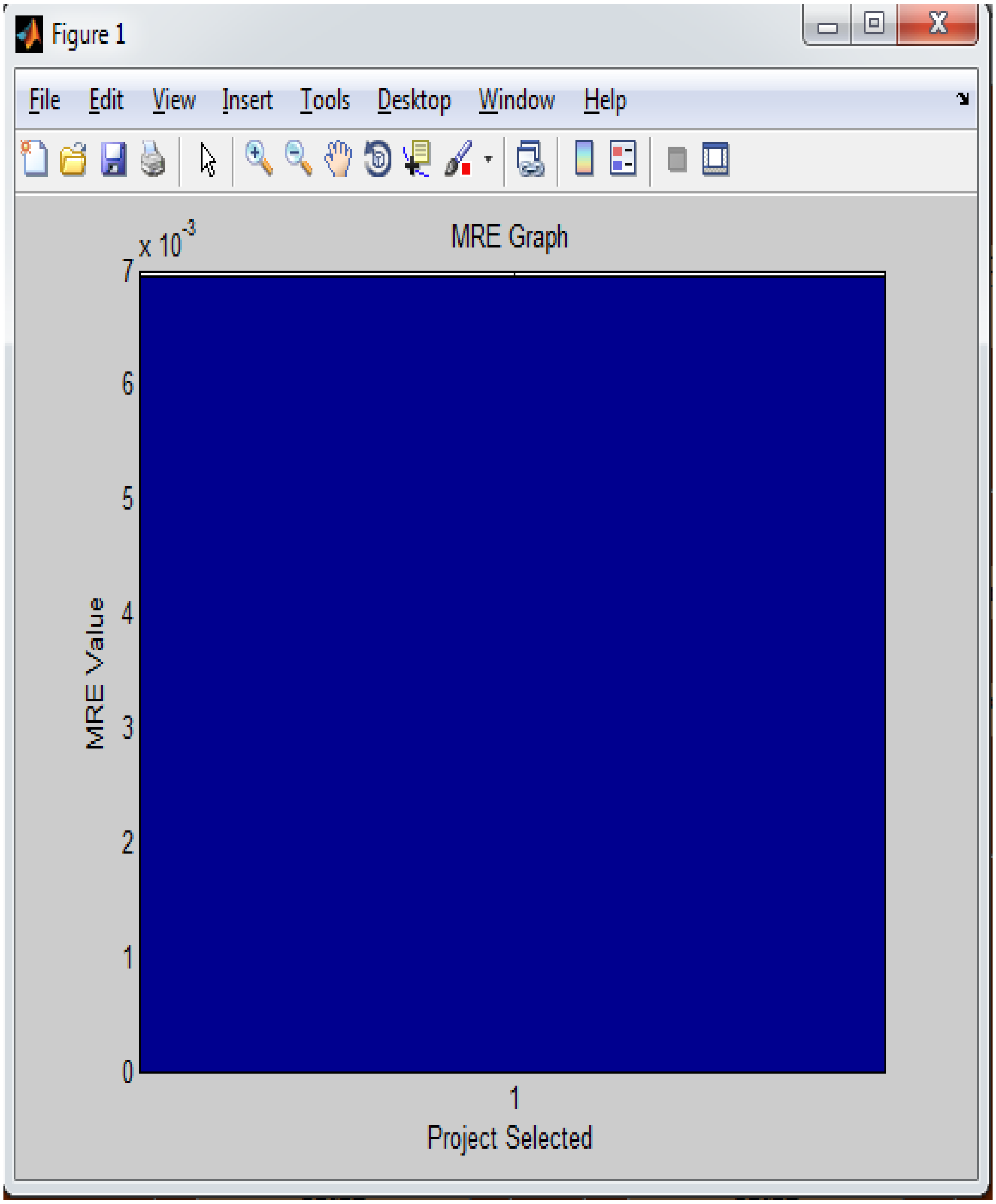
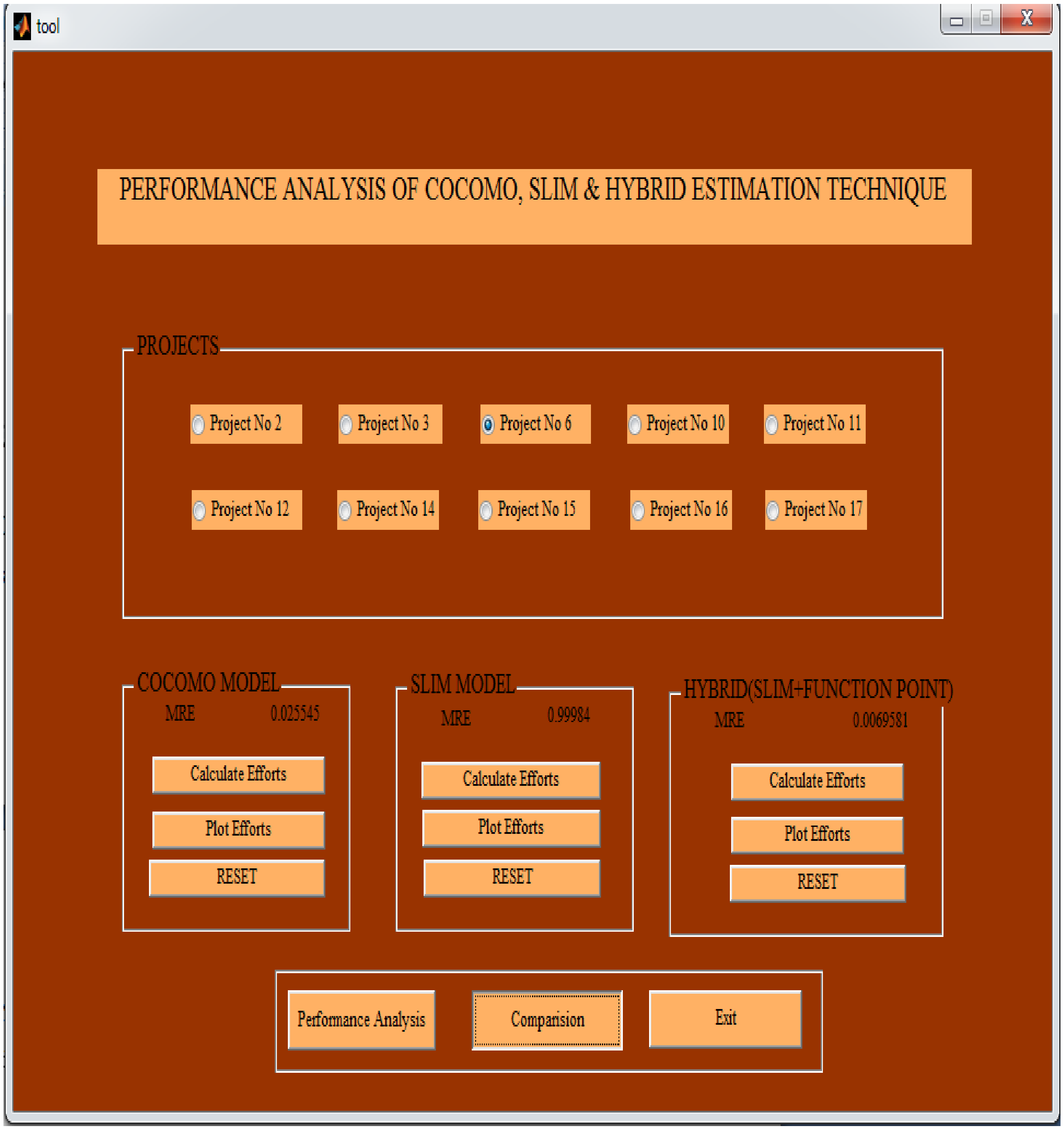


Figure 4.13: Hybrid Estimation Model

Calculated value of Hybrid model are plotted graphically where X- axis shows selection of the project and Y-axis shows its respective magnitude relative error value. . The values are plotted on the basis of the magnitude relative error.

Now, Comparing these three models on the basis of their Magnitude Relative Error which is done by pressing the Comparison button.



Selected project: Project No 6 [✓]

Model	MRE
COCOMO MODEL	0.025545
SLIM MODEL	0.99984
HYBRID(SLIM+FUNCTION POINT)	0.0069581

Figure 4.14: Comparison among three different Estimation Model

By pressing the comparison button as indicated in the above figure, Comparison of COCOMO, SLIM and Hybrid estimation model which is a combination of SLIM and Function Point are done. Comparison of models gives us clear picture as which model is more efficient so that the particular model can be used during effort estimation.

Plotting of Three estimation model

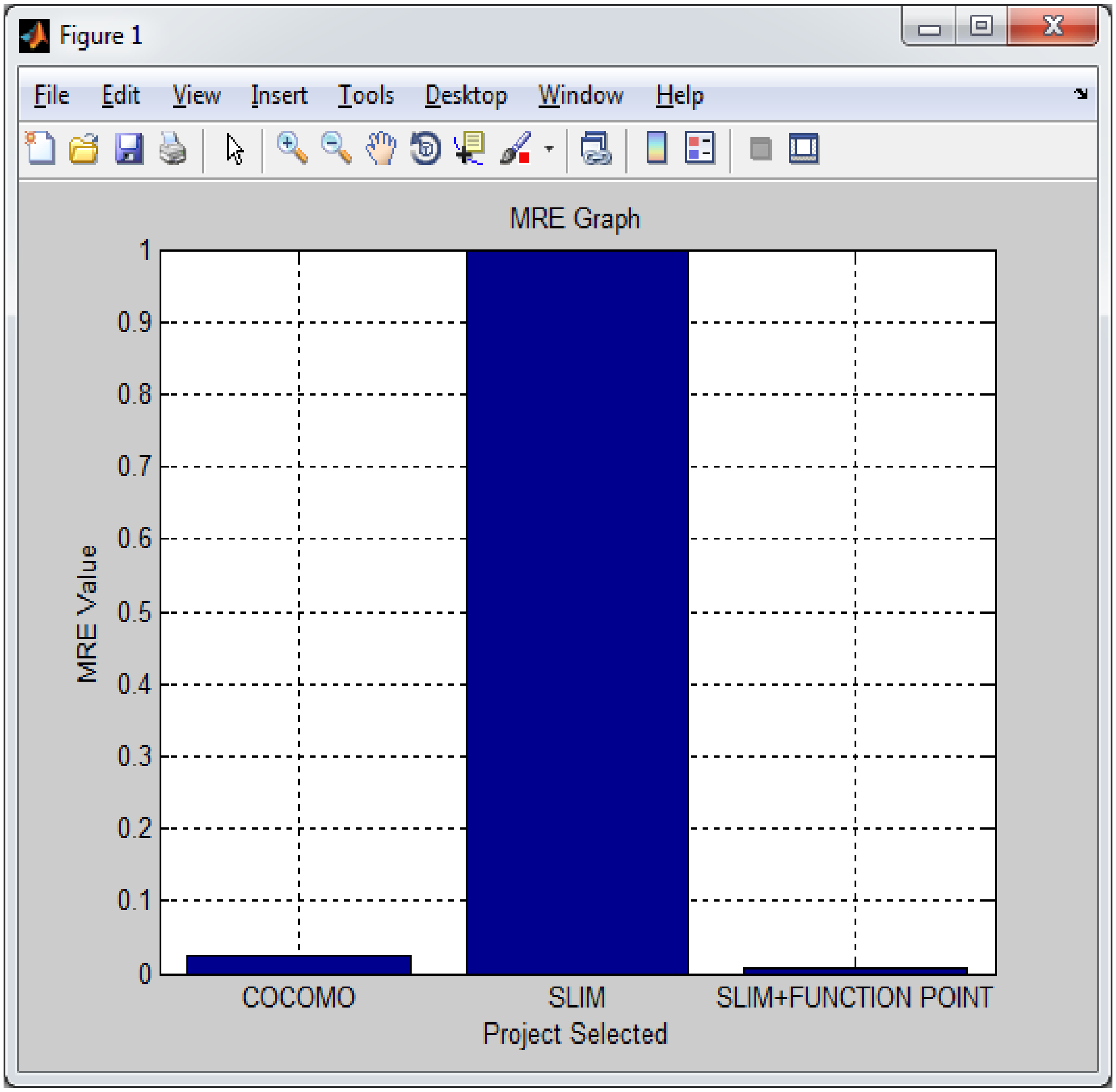


Figure 4.15: Comparison of Estimation Models

Comparison among three estimation model are plotted graphically where X-axis shows the selection of three models COCOMO, SLIM and new model Hybrid Estimation which is combination of SLIM & Function Point. The bar graph clearly depict COCOMO MRE is far lower than SLIM but greater than Hybrid estimation technique which proves that new proposed estimation model gives less error as compared to other two models SLIM and COCOMO.

Pie chart representation of three estimation models

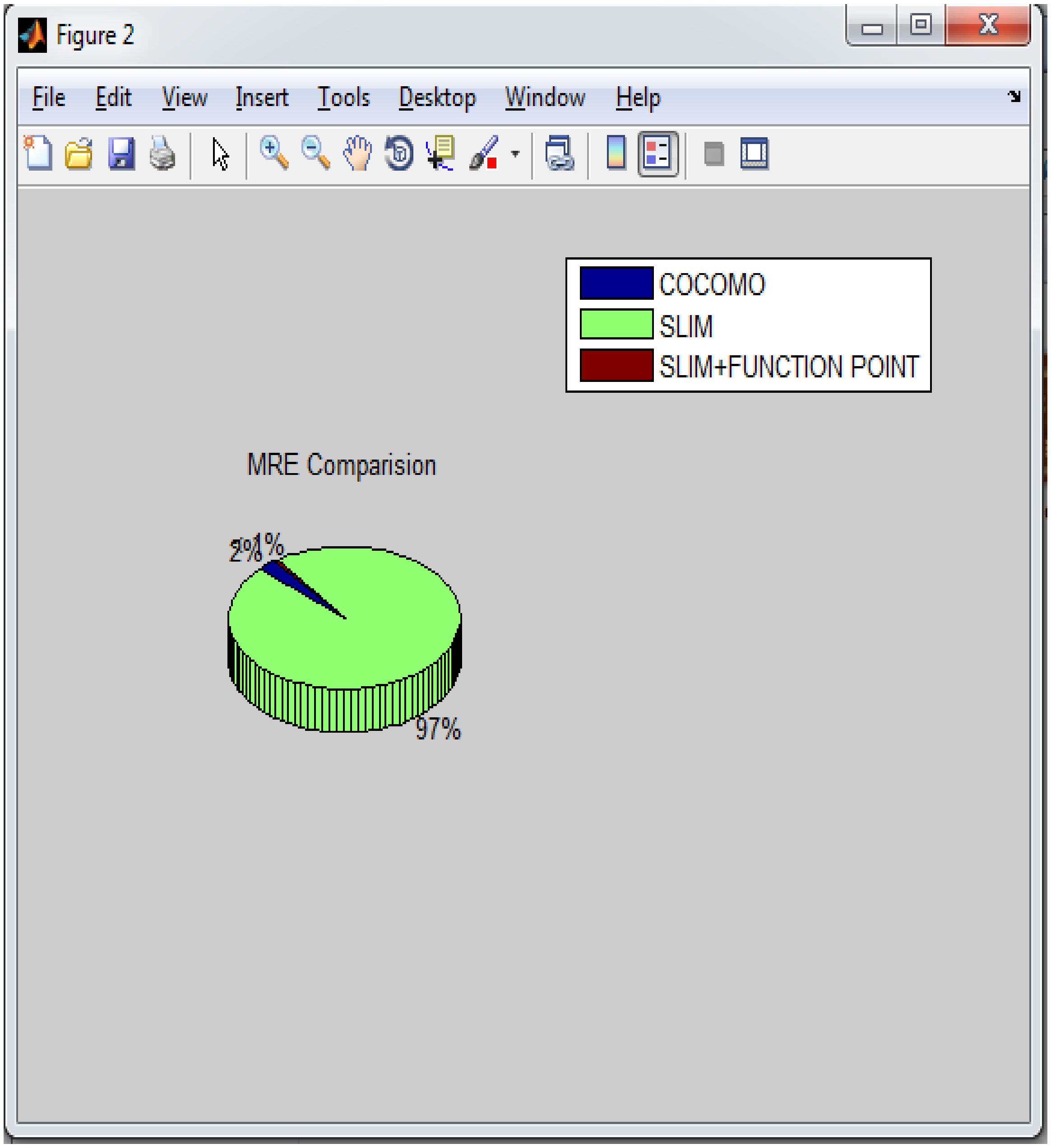
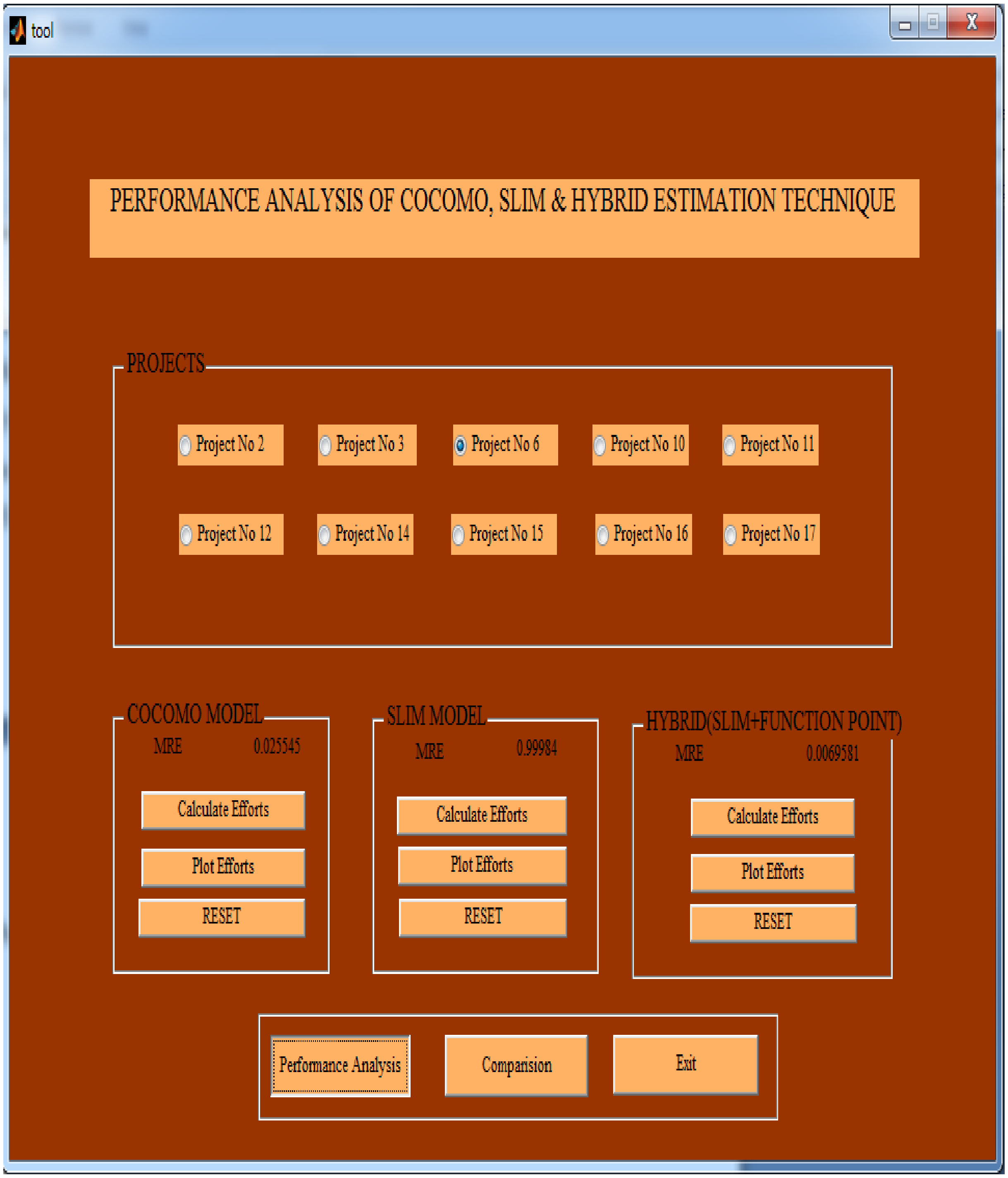


Figure 4.16: Pie chart representation of Estimation Models

Figure above, clearly represents that SLIM rate of error is much higher than that of COCOMO which focuses that COCOMO gives better result than SLIM but comparing COCOMO with Hybrid estimation model reveals that new model is better than that of COCOMO. Figure shows 97% error by SLIM, 2% by COCOMO and only 1% by Hybrid (SLIM + Function POINT) estimation.

Performance analysis of three Estimation Models



Project No 2 ( ) Project No 3 ( ) [✓ Project No 6] Project No 10 ( ) Project No 11 ( ) Project No 12 ( ) Project No 14 ( ) Project No 15 ( ) Project No 16 ( ) Project No 17 ( )

COCOMO MODEL MRE 0.025545

SLIM MODEL MRE 0.99984

HYBRID(SLIM+FUNCTION POINT) MRE 0.0069581

Figure 4.17: Performance Analysis

By pressing Performance analysis button accumulative values will be presented in form of graph.

Graphical representation of Estimation Models

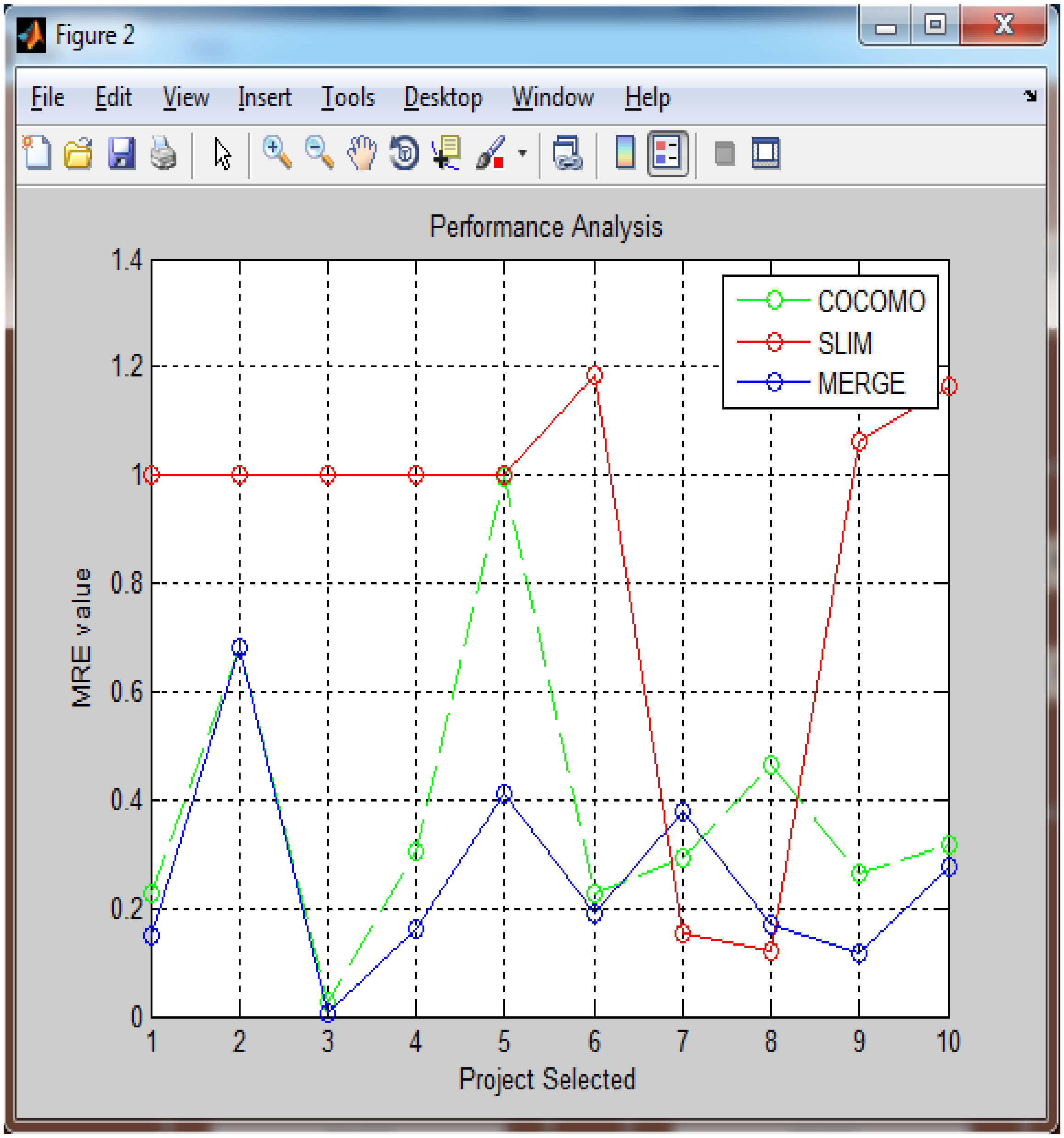


Figure 4.18: Performance Analysis of three models

Performance analysis of randomly selected 10 projects are depicted through Line graph where green line used for COCOMO, red for SLIM and blue for Hybrid estimation model. As graphically represented in the above figure, MRE value of Hybrid estimation model is lesser than other two estimation models. As MRE indicates how accurate estimation is, if the value of MRE is closer to 0% estimation is more accurate.

Effort, duration and management of man power are prime sources of effort estimation. If they are underestimated or overestimated they directly affect on the outcome. A proposed model that will enrich our ability to scientifically estimate software development effort and help to reduce the error injected in existing project. The proposal to hybrid parametric models are expected to deliver failure free estimates and within a specified period of time. This will in turn lead to a reduction of the number of IT-project failures and make better use of scarce financial and human resources.

As mentioned, while calculating the magnitude relative error values of COCOMO and SLIM they are on higher side than compared with the hybrid estimation model. MRE reaching towards 0% is an indication of error free result. By looking and comparing three different models the efficiency of Hybrid estimation model is significantly proved.

MMRE of COCOMO is 2.5545

MMRE of SLIM is 0.8971

MMRE of Hybrid estimation is 0.2641

Hence, looking into MMRE of three different models significantly prove the efficiency of new proposed model Hybrid estimation model.

Table 4.1: Comparison among three models on the basis of MRE & MMRE

PROJECTS	**COCOMO**	**SLIM**	**HYBRID(SLIM + FUNCTION POINT)**
Project 2	0.22586	0.99987	0.14805
Project 3	0.68096	0.99995	0.68305
Project 6	0.02554	0.99984	0.00695
Project 10	0.30394	0.99982	0.16279
Project 11	0.99734	0.99910	0.41309
Project 12	0.22586	1.1844	0.18071
Project 14	0.29439	0.15346	0.38061
Project 15	0.46582	0.12205	0.17131
Project 16	0.26216	1.0636	0.11624
Project 17	0.31678	1.1651	0.2754
MMRE(20)	2.5545	0.8971	0.2641

Randomly 10 projects are selected to calculate the value of Magnitude Relative Error. By looking into table we get a clear picture of which model is model efficient among three. MRE indicates Hybrid estimation model is much more efficient than the COCOMO and SLIM.

Chapter 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

Software development effort estimation is the process of amount of effort required to develop or maintain software based on incomplete, uncertain and noisy input. Software estimation is the mechanism of predicting cost, effort and duration that are required to develop software. Estimator often depends on number of pragmatism to generate software estimations. The proposed model to hybrid the parametric models are expected to deliver accurate & failure free estimates and that to within a specified period of time. This will in turn lead to a reduction of the number of IT-project failures and make better use of scarce financial and human resources. The proposed work offers important perspectives on the role of effort estimation in the development process, and shows how effort estimation directly improves software quality and output efficiency.

We are proposing enhancement in the SLIM model for effort estimation. The new algorithm is proposed for selecting the most relevant features of effort estimation and cost drivers which has significant influence on the SLIM model for effort estimation where cost drivers are the effort multiplier. The LOC of SLIM model is directly proportional to effort estimation which will affect the effort estimation directly, some time it will be overestimated or underestimated due to vagueness in the cost drivers. So, by hybridization of two different models will enhance the accuracy and will efficiently improve the performance of the effort estimation.

5.2 Future Scope

A successful project is one delivered ‘on time, within budget and with required quality’. Over-estimate may cause the project to take longer than it would otherwise while under-estimated project might not be completed on time or within the specified cost. An estimate is not really a prediction, it is a management goal. The future scope is to improve accuracy of the estimation model. The possibility of enhancing the accuracy of the effort estimation can be done through hybridization of other parametric models. Accurate estimate will produce or estimate better result which is helpful for future use. New model help to remove problems which occur in existing model and produce better result. By working on the shortcomings of the software parametric models the enhancement in accuracy is possible.

REFERENCES

RESEARCH PAPER

[1] Aihua Ren, Chen Yun, “Research of Software Size Estimation Method”, ©2013 IEEE.

[2] Briand, L. C. and I. Wieczorek (2002),”Resource estimation in software engineering. Encyclopedia of software engineering”, J. J. Marcinak. New York, John Wiley & Sons: 1160-1196.

[3] Chetan Nagar and Anurag Dixit (2012), “Efforts estimation by combining the use case point and COCOMO”, *IJCA journal.*

[4] Derya Toka, Oktay Turetkan, “Accuracy of Contemporary Parametric Software Estimation Models”, ©2013 IEEE.

[5] D. Manikavelan, Dr.R.Ponnusamy, “To find the accurate software cost estimation using Differential Evaluation algorithm”, ©2013 IEEE.

[6] G. Stark, “A Comparison of Parametric Software Estimation Models Using Real Project Data,” *Crosstalk*, no. Jan/Feb 2011.

[7] H. Leung and Z. Fan, “Software Cost Estimation,” pp. 1–14, 2002.

[8] Himani Rastogi, Misha Kakkar,“A Survey on Software Effort Estimation Techniques”, ©2014 IEEE.

[9] H. Azath, R.S.D Wahidabanu, “Efficient effort estimation system viz. function points and quality assurance coverage*”, IET soft. 2012, Vol.6, Iss. 4*

[10] Inderpreet kaur walia and Manpreet kaur, “To Enhance the Effort Estimation Accuracy of COCOMO Model using Function Point, IJESRT, December 2013.

[11] Jørgensen, M. Shepperd, M. "A Systematic Review of Software Development Cost Estimation Studies".

[12] Juan J.Cuardrado Gallego, Daniel Rodiguez, Software Project Effort Estimation Based on Multiple Parametric Models Generated Through Data Clustering,*Journal of computer science and technology*, may 2007.

[13] Juha Koskenkylä, "Cost estimation in global software development – review of estimation technique",Department of Information and Service Economy, Aalto university,2012

[14] Mohammad Azeeh, Software Cost Estimation Based on Use Case Points for Global Software Development", *CSIT*, ©2013 IEEE

[15] Poonam Pandey, "Analysis of the Techniques for Software Cost Estimation", ©2013 IEEE

[16] Putnam, Lawrence H. (1978). "A General Empirical Solution to the Macro Software Sizing and Estimating Problem". IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. SE-4, NO. 4, pp 345-361.

[17] R. N. Charette, "Why Software Fails?" IEEE *Spectrum*, *Sept.2005*, 2005.

[18]Wei Lin Du, Danny Ho,Luiz Fernando Capretz "Improving Software Effort Estimation UsingNeuro-Fuzzy Model with SEER-SEM", *Global Journal of Computer Science and TechnologyVol. 10 Issue 12 (Ver. 1.0) October 2010*

BOOK

[19] Software Project Management by Bob Hughes published by Tata McGraw Hill.

[20] Metrics and Models in Software Quality Engineering by Stephen H. Kan published by PEARSON.

[21] Cost estimation model – online source

APPENDIX

COCOMO	Constructive Cost Model
SLIM	Software Lifecycle Management
SEER-SEM	SEER for software, which means one has ability to foresee the future
FPA	Function point analysis
SLOC	Source lines of code
GUI	Graphical user interface
TCP/IP	Transport layer protocol
API	Application programming interface
UFP	Unadjusted function point
VAF	Value adjusted function point
EIF	External interface file type
LIF	Logical internal file type
WBS	Work breakdown structure
MMRE	Mean magnitude of error
MRE	Magnitude of error