Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 10, Issue. 3, March 2021, pg.46 – 54

A New Cost-Quality Estimation Model Based on Case-Based Reasoning Technique

Abdulelah G.F.Saif
Computer Science Department & King Khalid University, KSA
absaif@kku.edu.sa

DOI: 10.47760/ijcsmc.2021.v10i03.006

Abstract— Software cost, time and quality estimation is very important for developing new software projects. Accurate estimates of software cost, time and quality are required for developing software systems efficiently. A lot of estimation methods have been developed. Among those, COCOMO II, the model widely used for estimating effort and time, and COQUAMO, the model used to estimate the quality of the software project. These two models need to be calibrated when the dataset grows up to get new values for their parameters. Currently, neural network, and fuzzy logic modelling etc. are used for finding the accurate estimates. Since fuzzy logic approach is difficult to use and there are no clear guidelines for designing neural networks. In this paper, a new cost-quality model based on case based reasoning CBR technique is introduced for estimating software effort, time, and quality due to its dependence on historical information from past similar completed software project which means accurate estimates can be obtained. To estimate new software projects, the experiments have been conducted on NASA 93 dataset. As result, good estimates are obtained.

Keywords— software cost and quality estimation, case based reasoning, estimation models, machine learning

I. Introduction

A successful software project must be completed on time, within budget, and deliver a quality product that satisfies users and meets requirements. Unfortunately, many software projects fail. A report by the Standish Group noted that only a third of all software development projects were successful, in terms of they met budget, schedule, and quality targets [1]. Software projects usually don't fail during the implementation and most project fails are related to the planning and estimation steps. During the last decade several studies have been done in term of finding the reason of the software projects failure. Galorath et al. performed an intensive search between 2100 internet sites and found 5000 reasons for the software project failures. Among the found reasons, insufficient requirements engineering, poor planning the project, suddenly decisions at the early stages of the project and inaccurate estimations were the most important reasons [2]. So accurate software cost, time and quality estimation is necessary and is critical to both developers and customers. Software cost estimation is related to how long and how many people are required to complete a software project. Software cost estimation starts at the proposal state and continues throughout the life time of a project [3].The major part of cost of software development is the human-effort and most cost estimation methods focus on this aspect and give estimates in terms of person-month [4]. In spite of accurate planning, well documentation and proper process control during software development, occurrences of certain defects are inevitable. These software defects may lead to degradation of the quality which might be the underlying cause of failure [5]. Therefore, in order to manage budget, schedule and quality of software projects, various software estimation methods have been developed. Among those methods, COCOMO II is the most widely used model due to its simplicity for estimating the effort in person-month and the time in months for a software project at different stages, and COQUAMO is the model used to estimate the quality of the software project in terms of defects/KSLOC(Thousand Source Line Of Code) (or some other unit of size). Today's models are based on neural network, genetic algorithm, the fuzzy logic modeling etc .Since fuzzy logic approach is difficult to use and there are no clear guidelines for designing neural networks [4]. In this paper, a new cost-quality model based on case-based reasoning technique is developed for estimating effort, time and quality for the software project under development. To estimate new software projects, the experiments have been conducted on NASA 93 dataset. As result, good estimates are obtained.

The rest of the paper is organized as follows: in section II overview of machine learning, section III case-based reasoning, section IV architecture of the proposed model, section V experiments and evaluation, and section VI discusses and concludes the paper.

II. RELATED WORK

A. *SOFTWARE COST ESTIMATION*

Lately many researchers have been focused on cost estimation field for the software projects using the AI techniques. But it is not possible to say that AI is 100% percent to estimate the costs accurately, however the studies showed that the AI techniques have been more efficient in comparison to the algorithmic techniques. COCOMO is a model for cost and time estimation of the software projects among the algorithmic methods [6]. Authors in [7] present study aimed at investigating the estimation accuracy of four widely used parametric software estimation models, namely COCOMO II, SEER-SEM, SLIM, and TruePlanning by comparing their performances on 51 software development projects residing in the ISBSG project repository. The results with regard to effort estimation were the accuracy levels of TruePlanning, SEERSEM and SLIM are alike, whereas COCOMO II scored the lowest in terms of effort estimation accuracy. The COCOMO II follows a pessimistic approach, while the approach followed by the other three is optimistic. The results with regard to duration estimation, SEER-SEM had the lowest *Mean Magnitude of Relative Error* (MMRE) value, whereas all four methods are pessimistic in estimating duration. COCOMO II performed better in estimating the project duration than the effort. SEER-SEM was (relatively) successful in both effort and duration estimation. TruePlanning performed better in estimating effort than duration. The major limitation for their study is the partial project information in the ISBSG project repository. This study suggests that these models need improvements regarding prediction accuracy.

In [8], authors discuss popular software cost estimation techniques including expert-judgment, analogy-based, function points, COCOMO II, neural networks and fussy logic and also in [9] authors discuss, in addition to previous methods in [8], Particle Swarm Optimization (PSO) and Genetic Programming (GP) in terms of their capabilities, strengths and weaknesses, in order to project manager choose the best estimation method according to the information and data available about project to avoid project failures.

In [2], authors present the strength and weakness of various software cost estimation methods which include algorithmic methods such as function points, COCOMO 81, COCOMO II, SLIM ,SEL, Doty, Walston-Felix, BaileyBasil and Halstead models and non-algorithmic methods such as expert-judgment, analogy-based, Top-Down, Bottom-up, Parkinson’s Law and Price-to-win and perform a comparative analysis using COCOMO dataset among algorithmic models and the performance is analysed and compared in terms of MMRE and PRED (Prediction). They also focus on some of the relevant reasons that cause inaccurate estimation. At the end, authors conclude that all estimation methods are specific for some specific type of projects. It is very difficult to decide which method is better than all other methods because every method or model has an own significance or importance. To understand their advantages and disadvantages is very important when we want to estimate our projects.

One of the problems with using COCOMO today is that it does not match the development environment of recent times, so authors, in [9] and [10], presented a detailed study on the use of binary genetic algorithm as an optimization algorithm which provided solution to adjust the uncertain and vague properties of software effort drivers by tuning COCOMO parameters in order to get better effort estimate. The performance of the developed models, in [9] and [10], was tested on NASA software project dataset and the developed model in [9] compared to the pre-existed model. The developed model in [9] was able to provide better estimation capabilities.

In [6], authors have proposed a hybrid model based on Genetic Algorithm (GA) and Ant Colony Optimization (ACO) for optimization of the effective factors’weight in NASA dataset software projects. The results of the experiments show that the proposed model is more efficient than COCOMO model in software projects cost estimation and holds less Magnitude of Relative Error (MRE) in comparison to COCOMO model.

In [11], authors have investigate the role of fuzzy logic technique in improving the effort estimation accuracy using COCOMO II by characterizing inputs parameters using Gaussian, Trapezoidal and Triangular membership functions and comparing their results. NASA (93) dataset is used in the evaluation of the proposed Fuzzy Logic COCOMO II. After analyzing the results, it had been found that effort estimation using Gaussian member function yields better results for maximum criterions when compared with the other methods.

B. SOFTWARE QUALITY ESTIMATION

There are many prediction models that can be used to predict software defects such as machine learning based models (artificial neural networks (ANN), Bayesian belief networks (BBN), reinforcement learning (RL), genetic algorithms (GA), genetic programming (GP) and decision trees) [5] and fuzzy logic models [2] etc. However, each one has its own advantages and disadvantages and each one can be used for specific projects at different stages [5].

III. OVERVIEW OF MACHINE LEARNING

Machine learning deals with the issue of how to build programs that improve their performance at some task through experience. Machine learning methods are used to predict or estimate software quality, software size, and software development cost or software effort. Machine learning has been utilized in various problem domains. Some typical applications of machine learning are: Optical character recognition, Face detection, Spam filtering, Fraud detection, Medical diagnosis and Weather estimation. Major categories of machine learning techniques are: Case-based reasoning (CBR), Rule induction(RI), Neural networks(NN), Genetic algorithms(GA) and Inductive logic programming(ILP)[12].

IV. CASE-BASED REASONING

Case-Based reasoning is one of the most popular machine learning techniques. Case-based reasoning is a problem solving paradigm that is fundamentally different from other major AI approaches, in that instead of relying solely on general knowledge of a problem domain, it uses specific cases. Case-based estimation is one of the more attractive techniques in the software quality estimation field. Central tasks that all case-based reasoning methods have to deal with are to identify the current problem situation, find a past case similar to the new one, and use that case to suggest a solution to the current problem, evaluate the proposed solution, and update the system by learning from this experience [12].

A. ANALYSIS

We can summarize the four primary steps of a CBR estimation system Fig.1 as: 1. ***RETRIEVE***: The most similar case or cases 2. ***REUSE***: The information and knowledge in that case to solve the problem 3. ***REVISE***: The proposed solution 4. ***RETAIN***: The parts of this experience likely to be useful for future problem solving.

A new problem is solved by retrieving one previously experienced case, reusing the case in one way or another, revising the solution based on reusing a previous case, and retaining the new experience by incorporating it into the existing knowledge-base (case-base) [12].

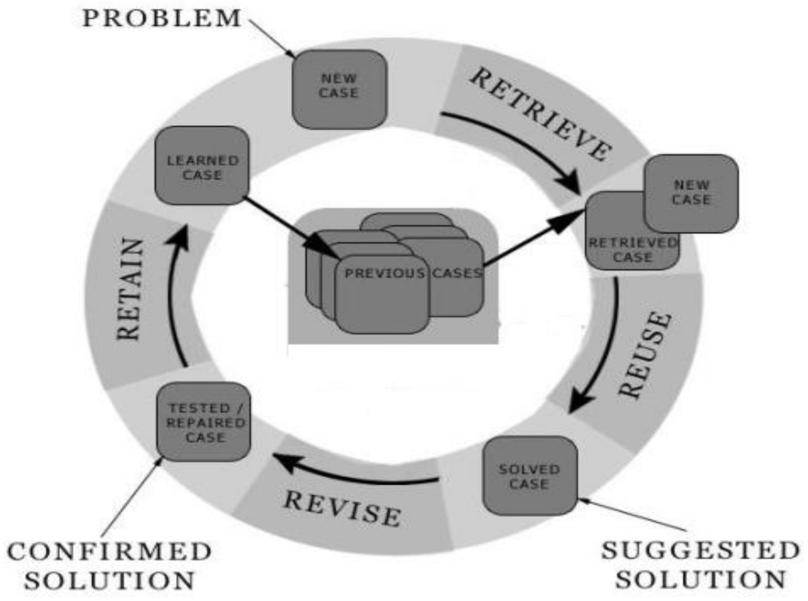


Fig. 1 The CBR Cycle

B. SIMILARITY MEASURES

Suppose a record set P_1 of n fields has following values $f_1, f_2, \dots, f_n$ for the n fields respectively. Similarly a record set P_2 of same type as P_1 with field values $g_1, g_2, \dots, g_n$. Then The Euclidian distance (ED) of P_1 from P_2 is:

$$ID = \sqrt{(f_1 - g_1)^2 + (f_2 - g_2)^2 + \dots + (f_n - g_n)^2} \quad \dots..(1)$$

The Manhattan distance (MD) of P1 from P2 is:

$$MD = |abs(f_1 - g_1) + abs(f_2 - g_2) + \dots + abs(f_n - g_n)| \quad \dots..(2)$$

V. ARCHITECTURE OF THE PROPOSED MODEL

The proposed model Fig 2 works as follow: firstly, the features of the new project need to develop, software size, project factors rating, product factors rating, platform factors rating, and personal factors rating, are entered into the model. Then, CBR compares the new project with all projects in the dataset using similarity measures(ID/MD). After that, the project in the dataset with smallest similarity measure (ID/MD) is selected as an estimate for the new project. Finally, the new project with its features are added to the dataset.

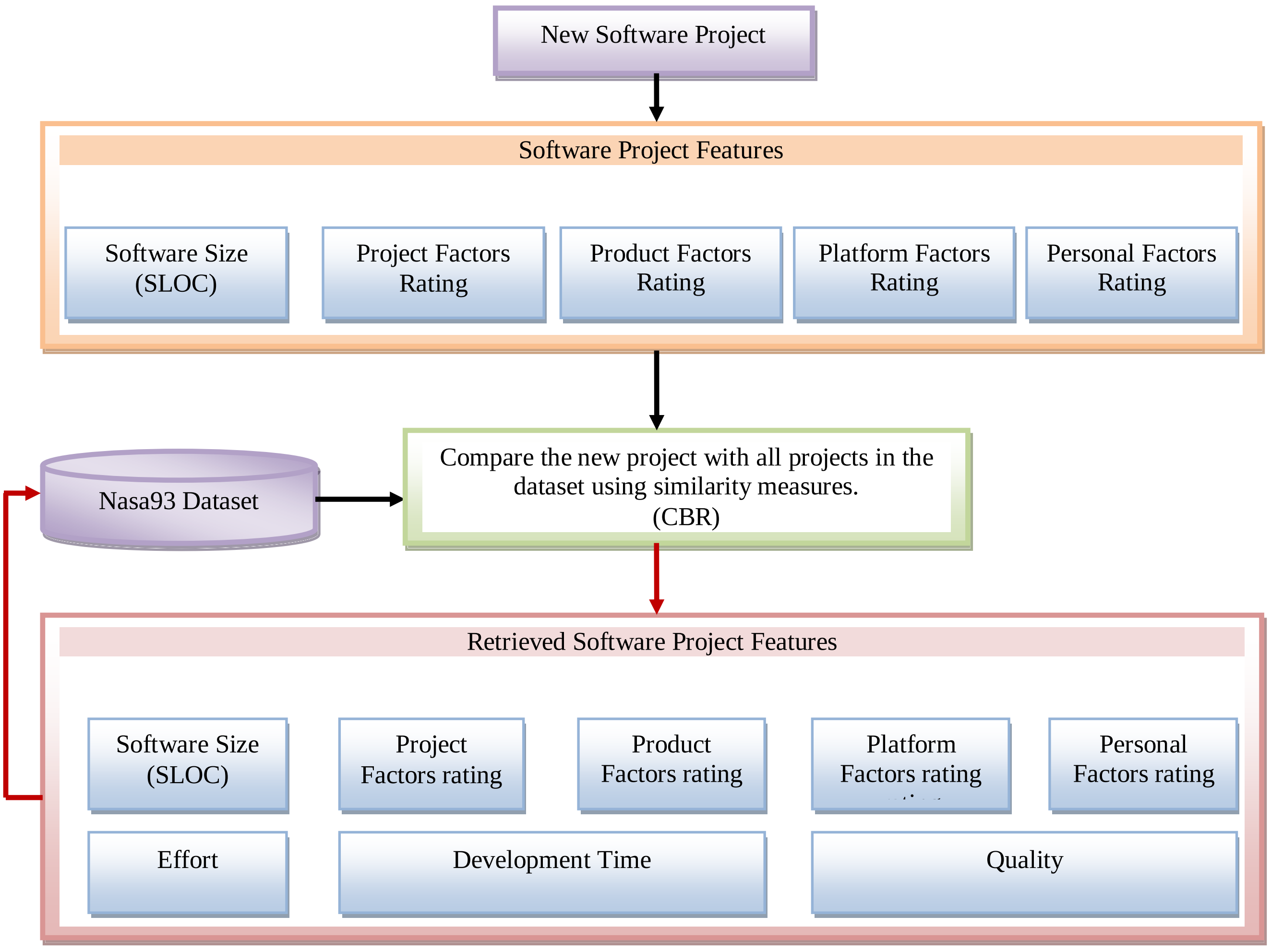


Fig.2 Architecture of the Proposed model.

VI. EXPERIMENTS AND EVALUATION

A. DATASET DESCRIPTION

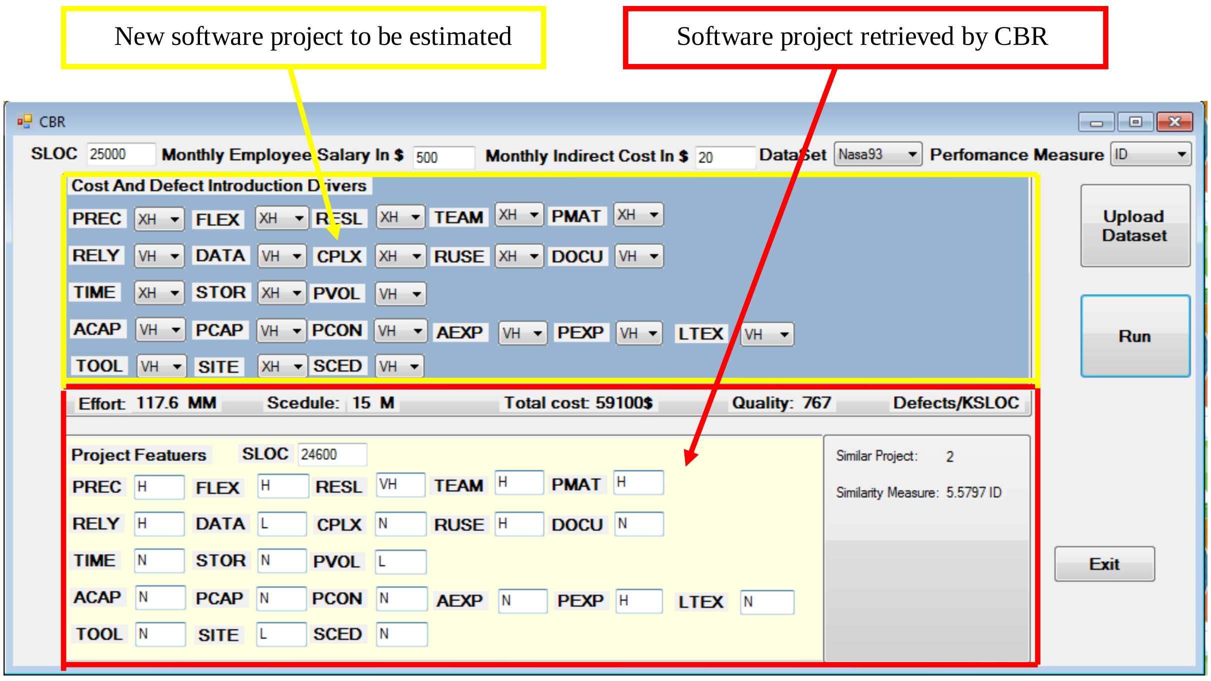
Experiments have been conducted on NASA 93 data set found in [13]. The dataset consist of 93 completed projects with its size in kilo line of code (KLOC), actual effort in person-month, development time in months and quality in Defects/KLOC. Factors rating from Very Low to Extra High are also given in this dataset.

B. RESULT ANALYSIS

In this experiment, we want to estimate the software projects with size (SLOC) 3000, 25000, 50000, and100000 and their features are shown in Table I below. The software project with size 50000 is estimated using ID without adding the retrieved project to the dataset. So ED is better than MD, Fig 3and F ig 4.

TABLE I
SOFTWARE PROJECT FEATURES

Factor	**Description**	**Type**	**Rating**					
			Very low (VL)	**Low (L)**	**Nominal (N)**	**High (H)**	**Very high (VH)**	**Extra high (XH)**
RELY	Required software reliability	Product					✓	
DATA	Data base Size	Product					✓	
RUSE	Developed for Reusability	Product					✓	
DOCU	Documentation needs	Product					✓	
CPLX	Product complexity	Product					✓	
TIME	Execution Time Constraints	Computer						✓
STOR	Main Storage Constraints	Computer						✓
PVOL	Platform Volatility	Computer					✓	
ACAP	Analyst Capability	Personnel					✓	
PCAP	Programmer Capability	Personnel					✓	
APEX	Application Experience	Personnel					✓	
PLEX	Platform Experience	Personnel					✓	
LTEX	Language and tool experience	Personnel					✓	
PCON	Personnel Continuity	Project					✓	
TOOL	Use of Software Tools	Project					✓	
SITE	Multi site Development	Project						✓
SCED	Required development schedule	Project					✓	
PREC	Precedentedness	Project						✓
FLEX	Development Flexibility	Project						✓
RESL	Risk Resolution	Project						✓
TEAM	Team Cohesion	Project						✓
PMAT	Process maturity	Project						✓


New software project to be estimated
Software project retrieved by CBR
CBR
Upload Dataset
Run
Exit

SLOC 25000 Monthly Employee Salary In $ 500 Monthly Indirect Cost In $ 20 DataSet Nasa93 Perfomance Measure ID

Cost And Defect Introduction Drivers

PREC	XH	FLEX	XH	RESL	XH	TEAM	XH	PMAT	XH
RELY	VH	DATA	VH	CPLX	XH	RUSE	XH	DOCU	VH
TIME	XH	STOR	XH	PVOL	VH				
ACAP	VH	PCAP	VH	PCON	VH	AEXP	VH	PEXP	VH
TOOL	VH	SITE	XH	SCED	VH				

LTEX VH

Effort: 117.6 MM Scedule: 15 M Total cost: 59100$ Quality: 767 Defects/KSLOC

Project Featuers SLOC 24600

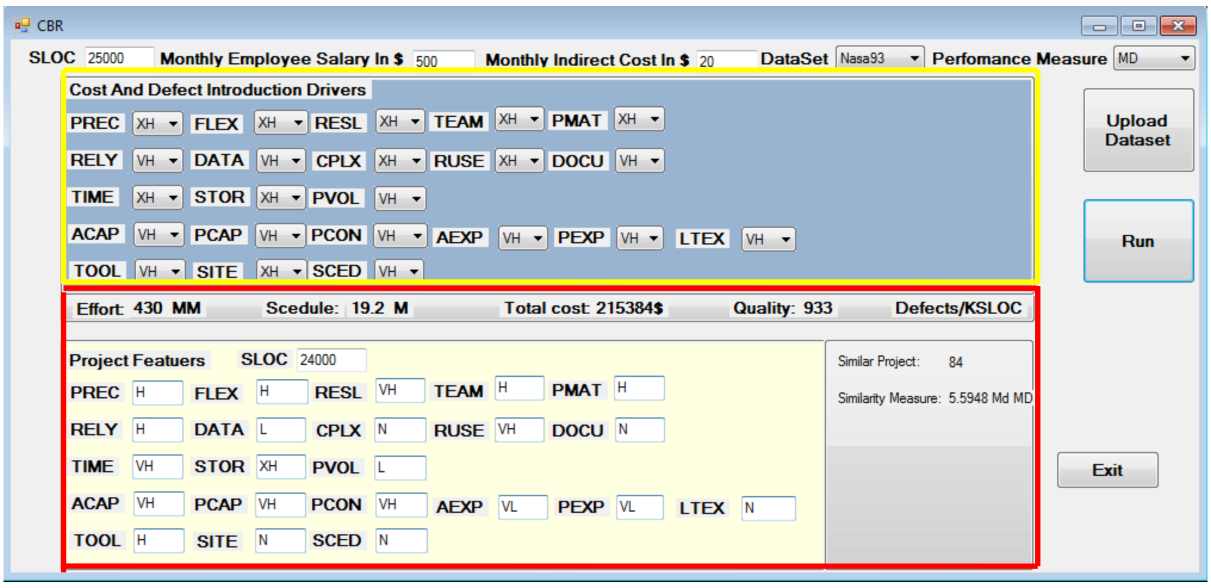
PREC	H	FLEX	H	RESL	VH	TEAM	H	PMAT	H
RELY	H	DATA	L	CPLX	N	RUSE	H	DOCU	N
TIME	N	STOR	N	PVOL	L				
ACAP	N	PCAP	N	PCON	N	AEXP	N	PEXP	H
TOOL	N	SITE	L	SCED	N				

LTEX N

Similar Project: 2

Similarity Measure: 5.5797 ID

Fig.3 Estimating software project with size 25000 SLOC using ID.



SLOC 25000 Monthly Employee Salary In $ 500 Monthly Indirect Cost In $ 20 DataSet Nasa93 Perfomance Measure MD

Cost And Defect Introduction Drivers

PREC	XH	FLEX	XH	RESL	XH	TEAM	XH	PMAT	XH
RELY	VH	DATA	VH	CPLX	XH	RUSE	XH	DOCU	VH
TIME	XH	STOR	XH	PVOL	VH				
ACAP	VH	PCAP	VH	PCON	VH	AEXP	VH	PEXP	VH
TOOL	VH	SITE	XH	SCED	VH				

LTEX VH

Effort: 430 MM Scedule: 19.2 M Total cost: 215384$ Quality: 933 Defects/KSLOC

Project Featuers SLOC 24000

PREC	H	FLEX	H	RESL	VH	TEAM	H	PMAT	H
RELY	H	DATA	L	CPLX	N	RUSE	VH	DOCU	N
TIME	VH	STOR	XH	PVOL	L				
ACAP	VH	PCAP	VH	PCON	VH	AEXP	VL	PEXP	VL
TOOL	H	SITE	N	SCED	N				

LTEX N

Similar Project: 84

Similarity Measure: 5.5948 Md MD

Fig.4 Estimating software project with size 25000 SLOC using MD.

Table II shows the results of comparing software projects in terms of software sizes, effort, development time, and quality using ID and MD measures. Fig.5 through Fig.9 shows the results graphically.

TABLE II
ESTIMATING SOFTWARE PROJECTS WITH DIFFERENT SIZES

Actual Software size (SLOC)	Software size (SLOC) retrieved by CBR using ED measure	Software size (SLOC) retrieved by CBR using MD measure	ED	MD	Effort(MM) using ID	Effort(MM) using MD	Development time (ED)	Development time using MD	Quality(ED)	Quality(MD)
3000	3500	3500	5.5877	5.5877	10.8	10.8	7.8	7.8	109	109
25000	24600	24000	5.5797	5.5948	117.6	430	15	19.2	767	933
50000	50000	50000	3.4691	3.4691	370	370	25.4	25.4	2685	2685
100000	100000	100000	3.0977	3.0977	360	360	28	28	4342	4342

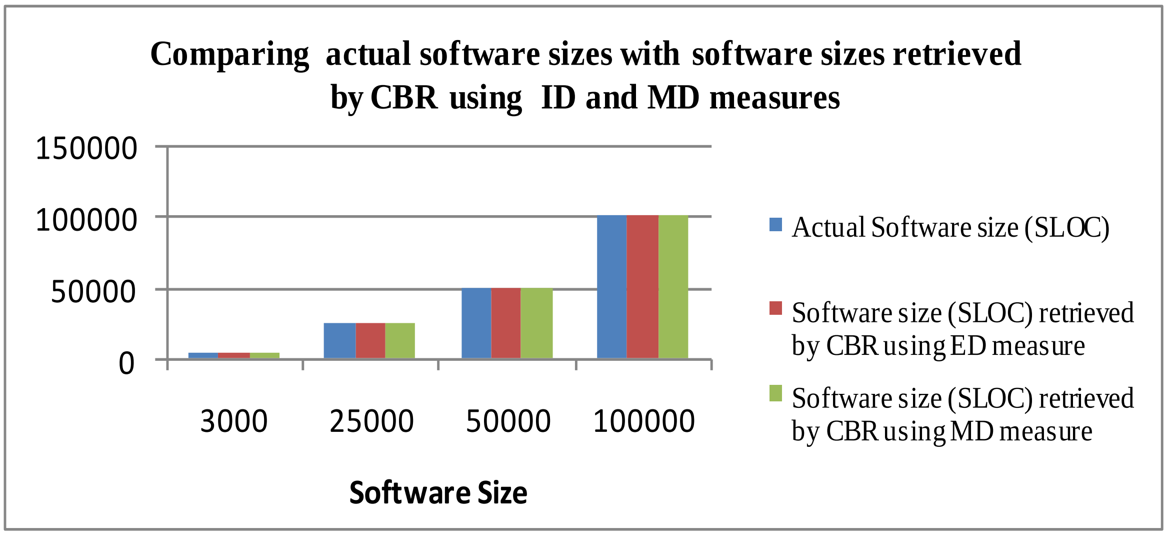


Fig.5 Comparing actual software sizes with software sizes retrieved by CBR using ID and MD measures.

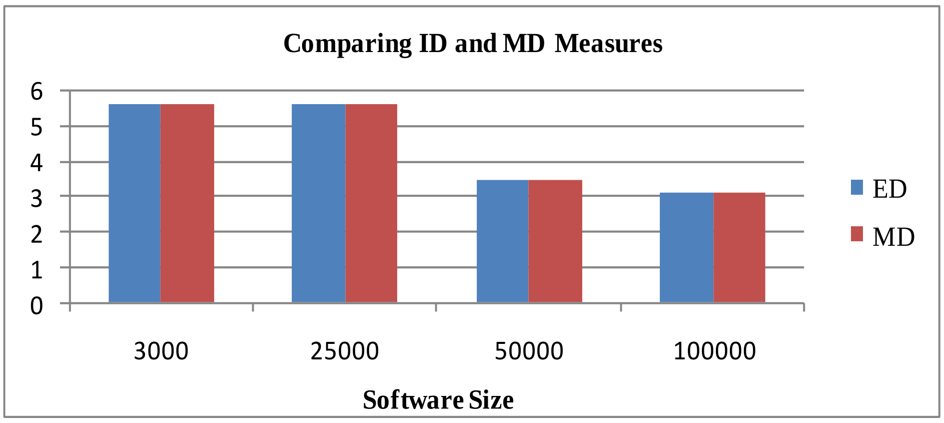


Fig.6 Comparing ID and MD measures.

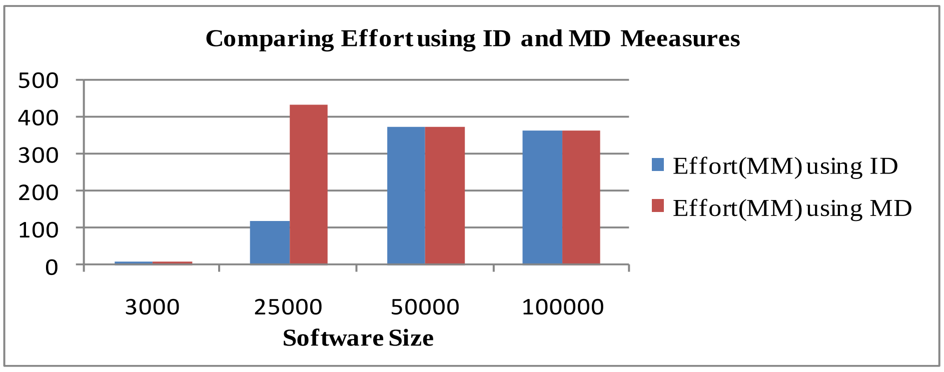


Fig.7 comparing effort using ID and MD measures.

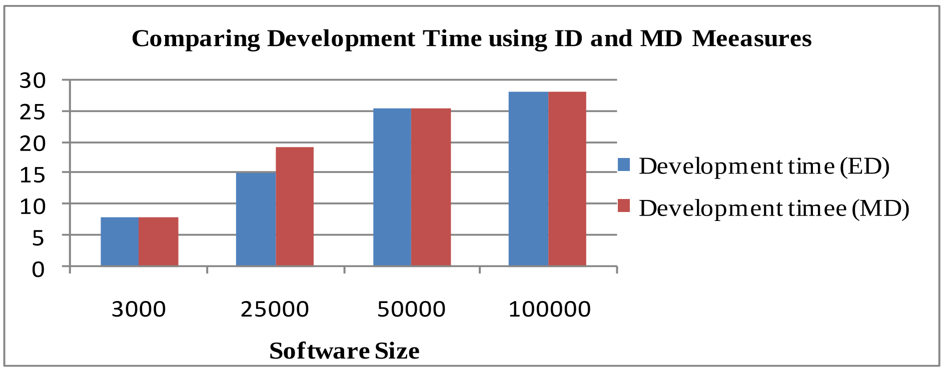


Fig.8 comparing development time using ID and MD measures.

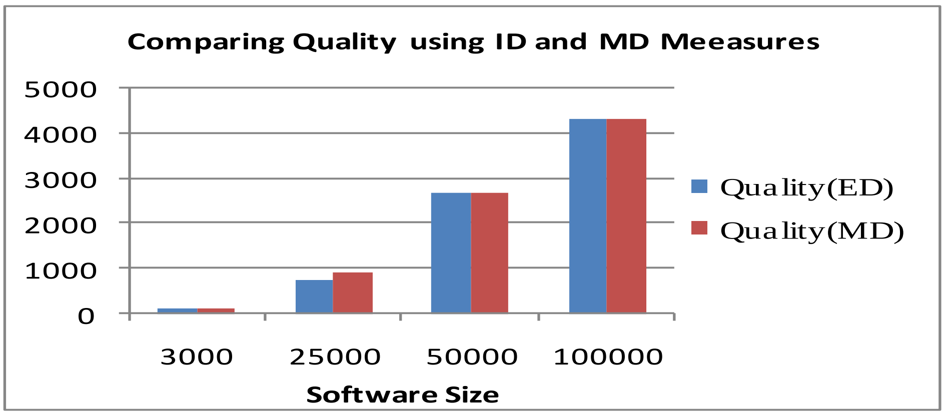


Fig.9 Comparing quality using ID and MD measures.

From table II and Fig.5 through Fig.9, we see that the results obtained by using ID and MD measure are same except for software project with size 25000 SLOC. The reason is, for software project with size 25000 SLOC, we don't include this software project to the dataset, when we used ID measure. So ID is better than MD, if we don't include the new project to the dataset.

VII. DISCUSSION AND CONCLUSION

Software cost, time and quality estimation is very important for developing new software projects. Accurate estimates of software cost, time and quality are required for developing software systems efficiently. A lot of estimation methods have been developed. In this paper, a new cost-quality model based on case based reasoning technique is introduced for estimating software cost, time, and quality due to its dependence on historical information from past similar completed software project which means accurate estimates can be obtained. To estimate new software projects, the experiments have been conducted on NASA 93 dataset. As result, good estimates are obtained.

REFERENCES

[1] Gary B. Shelly, Harry J. Rosenblatt, "*Systems Analysis and Design*", Ninth Edition", Shelly Cashman Series®, 2012.

[2] Vahid Khatibi, Dayang N. A. Jawawi, "Software Cost Estimation Methods: A Review", Volume 2 No. 1, *Journal of Emerging Trends in Computing and Information Sciences*, 2011.

[3] Sweta Kumari , Shashank Pushkar, "Performance Analysis of the Software Cost Estimation Methods: A Review ", *International Journal of Advanced Research in Computer Science and Software Engineering*, Volume 3, Issue 7, July 2013.

[4] Astha Dhiman, Chander Diwaker, "Optimization of COCOMO II Effort Estimation using Genetic Algorithm", *American International Journal of Research in Science*, Technology, Engineering & Mathematics, 2013.

[5] Mrinal Singh Rawat, Sanjay Kumar Dubey, "Software Defect Prediction Models for Quality Improvement: A Literature Study", *IJCSI International Journal of Computer Science*, Issues, Vol. 9, Issue 5, No 2, September 2012.

[6] Isa Maleki, Ali Ghaffari, and Mohammad Masdari, "A New Approach for Software Cost Estimation with Hybrid Genetic Algorithm and Ant Colony Optimization", *International Journal of Innovation and Applied Studies* ISSN 2028-9324 Vol. 5 No. 1 Jan. 2014, pp. 72-81.

[7] Derya Toka, Oktay Turetken, "Accuracy of Contemporary Parametric Software Estimation Models: A Comparative Analysis", *2013 39th Euromicro Conference Series on Software Engineering and Advanced Applications*.

[8] Vahid Khatibi, Dayang N. A. Jawawi, "Software Cost Estimation Methods: A Review ", Volume 2 No. 1, *Journal of Emerging Trends in Computing and Information Sciences*, 2011

[9] Brajesh Kumar Singh, A. K. Misra, " Software Effort Estimation by Genetic Algorithm Tuned Parameters of Modified Constructive Cost Model for NASA Software Projects", *International Journal of Computer Applications* (0975 – 8887), Volume 59– No.9, December 2012.

[10] Alaa F. Sheta, "Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects", *Journal of Computer Science* 2 (2): 118-123, 2006.

[11] Ashita Malik, Varun Pandey, Anupama Kaushik, "An Analysis of Fuzzy Approaches for COCOMO II", *I.J. Intelligent Systems and Applications*, 2013, 05, 68-75 Published Online April 2013 in MECS (http://www.mecs-press.org/).

[12] Ekbal Rashid, Srikanta Patnaik, and Vandana Bhattacherjee, "Software Quality Estimation using Machine Learning:Case-based Reasoning Technique", *International Journal of Computer Applications*, Volume 58– No.14, November 2012.

[13] Nasa 93 dataset contains effort and defect information available at http://promisedata.googlecode.com/svn/trunk/effort/nasa93-dem/nasa93-dem.arff.