JACET Journal of Advances in Computer Engineering and Technology, 2(2) 2016

An Improved COCOMO based Model to Estimate the Effort of Software Projects

Vahid Khatibi Bardsiri[1], Mahboubeh Dorosti[2]

Received (2015-12-04)
Accepted (2016-02-11)

Abstract - **One of important aspects of software projects is estimating the cost and time required to develop projects. Nowadays, this issue has become one of the key concerns of project managers. Accurate estimation of essential effort to produce and develop software is heavily effective on success or failure of software projects and it is highly regarded as a vital factor. Failure to achieve convincing accuracy and little flexibility of current models in this field have attracted the attention of researchers in the last few years. Despite improvements to estimate effort, no agreement was obtained to select estimation model as the best one. One of effort estimation methods which is highly regarded is COCOMO. It is an extremely appropriate method to estimate effort. Although COCOMO was invented many years ago, it enjoys the effort estimation capability in software projects. Researchers have always attempted to improve the effort estimation capability in COCOMO through improving its structure. However, COCOMO results are not always satisfactory. The present study introduces a hybrid model for increasing the accuracy of COCOMO estimation. Combining bee colony algorithm with COCOMO estimation method, the proposed method obtained more efficient coefficient relative to the basic mode of COCOMO. Selecting the best coefficients maximizes the efficiency of the proposed method. The simulation results revealed the superiority of the proposed model based on MMRE and PRED(0.15).**

Keywords: **COCOMO 81, effort estimation, development effort, software projects.**

1- Kerman Branch, Islamic Azad University, Kerman, Iran. (khatibi78@gmail.com)
2- Kerman Branch, Islamic Azad University, Kerman, Iran.

1-Introduction

Accurate estimation of software production cost for effective management of the project including budgeting, controlling, and programming is so important. In recent years, software has become the most expensive part of the computer projects. Some of these costs are resulted from human resource effort. Most of the cost estimation approaches focuses on individual-month based estimation. No model with effective estimation of constant software development effort has so far been introduced. Software accurate estimation is naturally a challenging process. Although various attempts have been put forth in recent decades, no consistent model for effective prediction regarding effort estimation has yet been proposed. The efficient selection of cost drivers and the measurement of COCOMO model's parameters based on the nature of the projects can be used as one solution of the problem [1].

Nowadays, there are many models to estimate software development effort. Managers select one of them depending on the type of project. One of popular and known mathematical models is COCOMO model used to estimate cost and time in software projects. This model was introduced using constant parameters in 1981 as well as applying statistical data regression analysis based on 63 various software projects [2]. Using parameters introduced many years ago, assessing a modern project is extremely challenging. Although COCOMO is a well-liked and extensively accepted model, it cannot be beneficial for today`s software projects. Generally speaking, parametric effort estimation models such as COCOMO are based on mathematical equations and they are rooted in

the study of software project data base. Most software project data bases enjoy heterogeneous nature. Ultimately, reaching a unified, logical, and acceptable parametric model (such as COCOMO) is extremely difficult for a wide range of software project sizes and properties. The objective of this paper is to review different types of COCOMO 81 model and COCOMO model improvement is then studied.

2-COCOMO 81

This method estimates the level of effort and time required to implement software project as a function of performance degree or the number of lines. These types of functions are experimentally determined. COCOMO models are extremely popular to determine cost of implementing projects. These models are used to determine total cost of project as well as the cost of project steps.

COCOMO is derived from Constructive Cost Model. This method was first introduced by Bari Boehm in 1981 [1].

Boehm presented three levels of this model: basic, medium, and detailed.

a)Basic COCOMO

It is a single -value static model which estimates software development effort (cost). In fact, it is function of program size stated by estimation of approximately a thousand delivered instructions. Projects are used by small familiar teams with working environment. The goal is usually clear for this type of projects and they do not have complex qualitative requirements. These projects are simple and they need maximum of 50000 program lines and they do not need innovation and creativity. Equation parameters are applied regardless many details of project properties. Equation 1 and 2 are for this model. COCOMO 81 model depends on two main equations:

$$\mathrm{MM} = \mathrm{a} \times \mathrm{KDSI}^{\mathrm{b}} \qquad (1)$$

$$\mathrm{TDEV} = 2.5 \times \mathrm{MM}^{\mathrm{c}} \qquad (2)$$

1. Person month: calculated from Equation 1. One month effort by an individual in COCOMO equals 152 person hours which might differ 10 to 20 percent according to the type of organization.

2. Effort and development time: obtained from Equation 2.

a, b, and c coefficients depend on the model. Table 1 lists specifications of three COCOMO models. In this table, specifications of all three types of COCOMO projects are shown including size, changes, limitations, and environment progress.

a, b, and c coefficients of basic COCOMO model are obtained from Table 2.

Table 1:Development modes

Development Mode	Project Characteristics			
	Size	Innovation	Deadline/constraints	Dev. Environment
Organic	Small	Little	Not tight	Stable
Semi-detached	Medium	Medium	Medium	Medium
Embedded	Large	Greater	Tight	Complex hardware/ customer interfaces

Table 2:Basic model coefficient

Software project	a	b	c
Organic	2.4	1.05	0.38
Semi-detached	3.0	1.12	0.35
Embedded	3.6	1.20	0.32

b)Intermediate COCOMO

In this model, software development effort is stated as function of size, program, and a collection of 15 cost stimulations including subjective evaluation of the product, hardware, personnel, and project features. This type of projects are relatively larger than simple ones and they require maximum number of 300000 lines. Team members are relatively familiar with the related system. The basic model of this method is similar to that of the previous model. In addition to effort adjustment factor, model “a” parameter differs slightly in comparison with the previous model; however, parameter “b” is equal in both models. The coefficients are shown in Table 3.

Table 3: intermediate model coefficient

Software project	a	b	c
Organic	3.2	1.05	0.38
Semi-detached	3	1.12	0.35
Embedded	2.8	1.2	0.32

In this model, efforts of person per month is obtained from Equation 3.

$$\mathrm{MM} = \mathrm{a} \times \mathrm{KDSI}^{\mathrm{b}} \times EAF \qquad (3)$$

In this equation, multiplication of coefficients or effort adjustment factor is obtained from

equation categorized to four groups: product features, hardware, personnel, and project.

C) Detailed COCOMO

All features of medium version, in this model, are mixed with an estimation from the effect of cost stimulations in each step (analysis, design, etc.) of software engineering process. These projects are addressed to large systems requiring between 200000 and 1000000 lines of program. Time constraints in these projects leave a lot of pressure on the project team and they need innovation. Hardware and software are closely linked to each other and with constraints. Time range is highly important for this type of software packages. This method calculates effort as a function of program size and collection of cost stimulation weights according to each step of software service life. This model applies medium model as level of elements. Then phasing-based approach is used for estimation process. Four phases of this COCOMO model are need programming and product designing, designing details, test of unit and code, and integration.

Estimation of each module is obtained by combined sub system and ultimately a general estimation of project. Life cycle of each phase is determined by taking advantage of detailed cost stimulations.

Table 4. Intermediate model effort drivers

effort Drivers	Ratings					
	Very Low	Low	Nominal	High	Very High	Extra High
Product attributes						
Required software reliability	0.75	0.88	1.00	1.15	1.40	
Size of application database		0.94	1.00	1.08	1.16	
Complexity of the product	0.70	0.85	1.00	1.15	1.30	1.65
Hardware attributes						
Run-time performance constraints			1.00	1.11	1.30	1.66
Memory constraints			1.00	1.06	1.21	1.56
Volatility of the virtual machine environment		0.87	1.00	1.15	1.30	
Required turnabout time		0.87	1.00	1.07	1.15	
Personnel attributes						
Analyst capability	1.46	1.19	1.00	0.86	0.71	
Applications experience	1.29	1.13	1.00	0.91	0.82	
Software engineer capability	1.42	1.17	1.00	0.86	0.70	
Virtual machine experience	1.21	1.10	1.00	0.90		
Programming language experience	1.14	1.07	1.00	0.95		
Project attributes						
Application of software engineering methods	1.24	1.10	1.00	0.91	0.82	
Use of software tools	1.24	1.10	1.00	0.91	0.83	
Required development schedule	1.23	1.08	1.00	1.04	1.10	

3-Review of the Literature

Software development effort estimation is critical for the success of software project management. This includes resource allocation, project auction, and project programming. The importance of accurate estimation stimulates various research attempts toward introducing software cost estimation approaches. Through a comprehensive study, these methods are classified into following sub classes including parametric models including COCOMO, lifelong management of the software, software estimation and evaluation through software estimation models, expert judgment including Delphi technique, basic user Breakdown Structure Methods, technique-based learning including machine learning, regression estimation-based comparison including regression methods including ordinary least squares regression procedures and strong regression, dynamic models based on complex methods, and finally estimation-based comparison originally as case-based reasoning approach [3].

Fi and Liu introduced F-COCOMO using fuzzy logic for software effort estimation. Since no comparison was made between fuzzy COCOMO and other effort estimation models, the estimation capability is not identified. Rodger introduced a fuzzy COCOMO recognized as adaptive model of effort drivers, though its efficiency is not mentioned [4]. Audrey et al. define a fuzzy set for linguistic values of each effort driver by a trapezoidal membership function for fuzzy COCOMO. Effort coefficients in the original model of COCOMO are obtained from fussy sets. Relative to 81 COCOMO, the fuzzy COCOMO is less sensitive toward software effort drivers [5]. Zhu et al. presented a modeling approach of fuzzy linguistic effort estimation for confronting linguistic effort drivers. They automatically generate fuzzy membership function through 81 COCOMO dataset. Relative to the main 3 models of COCOMO (basic, medium, and accurate), the proposed fuzzy identifying model presents a more accurate effort estimation [6].

Many models have been complemented for making relationship between size and effort in software cost estimation. Some of the applied methods in this regard include genetic algorithm [7], fuzzy models [8], synthetic and dynamic models [9], neural networks [10], and basic regression [11]. Two common methods of cost estimation approaches include algorithm and non-algorithm approaches [12]. Algorithm method expansively makes use of math skills. Some of them are based on simple calculation formula of statistics. Others are based on regression and differential equations. [13]. Non-algorithm approaches, on the other hand, are based on analysis, reasoning, and learning.

Recently, a research has been carried for optimizing the decision parameters in COCOMO through 81 NASA COCOMO datasets [14].

Different approaches have helped effort estimation for optimizing datasets values, machine learning, analogy, data mining, and neural networks [15]. Particle swarm optimization [16] is another approach used for optimization in this regard [17]. Da Silva et al. presented a multilayer neural network with 23 nodes in hidden layer for effort estimation in software projects. Regression approach is also run in this approach. 81 COCOMO is applied for regression approach and neural network evaluation. Normalization is carried on datasets and the results are obtained according to MMRE. The results obtained from neural network indicate higher accuracy. To estimate software project effort, Rao made use of artificial neural network of functional link whose architecture was too simple without any hidden layer. Learning process is faster in neural networks. This network is identified based on COCOMO method, 17 inputs of cost drivers, and 5 scales, andit is applied for network. The input performance is specifically carried [18].

Attarzadeh et al. presented a two-layer feedforward neural network. They considered 2 COCOMO principles for creating this network. 17 cost drivers and 5 scale factors operated as the input and the estimated effort as output. Sigmoid function was used as transfer function, and 81COCOMO was selected for network and artificial datasets evaluation. The results were presented according to MMRE and PRED (0.25) and compared with 2 COCOMO. As indicated by the results of the comparison, the presented networks can create higher accurate results [19].

Soda et al. proposed two neural networks of RBF and GR for development effort estimation of software projects. 81 COCOMO dataset is used in this study, and the obtained results of neural network are compared with the results obtained from COCOMO. As indicated by the results, both RBF and GR relative to COCOMO demonstrate higher accurate results. RBF presented the best results [2].

Rady and Raju presented a feedforward neural network with 22 neurons in input layer, two hidden layers, and one node in output layer. This architecture is based on 17 COCOMO effort coefficient and 5 scale factor. COCOMO equation changed to a linear equation. Therefore, linear transfer function is selected for the network. 81 COCOMO dataset was used for network performance evaluation according to MMRE. Fifteen projects are randomly selected as order set while others perform as test set. As indicated by the obtained results compared with COCOMO results, the proposed network presented higher accurate results [20].

Neural networks are extensively used for estimation objectives in various sciences. It was also used in software development effort estimation.

Rao used functional link artificial neural network in order to estimate effort for software projects. Network architecture is extremely simple. There is no hidden layer and learning process is extremely quick in this type of neural networks. According to COCOMO, this network receives 17 cost stimulation as input and 5 determined scale factor [21].

Reddy and Raju introduced feed forward neural network with 22 neurons in input layer, two hidden layer, and one node in output layer. This architecture is considered according to COCOMO estimation coefficients (EMS 17) and scale factors (5SFS). COCOMO equation changed into a linear equation and linear transfer function, therefore, was selected for the network. COCOMO 81 data collection was used to evaluate network performance according to MMRE. Total number of 50 projects act randomly as trial set and other projects as test set. Results compared to COCOMO results show that the ability of proposed network is more accurate for estimation [22].

Idri et al. tried to find an appropriate structure of Radial Basis Function Neural Network and in particular the number of neurons in hidden layer. This study focused on the effect of Gaussian function width on the level of accuracy of prediction in software projects. Two models were proposed to determine the width using K-means clustering. COCOMO 81-based artificial data collection and Tulutuku data collection were produced with 252 and 53 projects, respectively and they were used to evaluate the network. Two network configurations with different number of neural cells in hidden layer were evaluated and the effect of width adjustment was tested [16].

Artificial neural network is the most common learning-based method used to estimate software development effort. Neural network application is relatively new to estimate effort in a research. In the last few years, several studies were carried out to use model-based methods. In the last couple of decades, neural networks were used to predict in various effort estimation applications and the

results were reported better in comparison with ordinary methods. Model-based methods do not provide appropriate estimation at the beginning of the project due to limited project knowledge; while, artificial neural network-based methods can provide appropriate estimation in case of accessible information from efforts of completed projects [24].

Mirsa, in his studies, introduced new model to estimate software effort for NASA supported projects using binary genetic algorithm. Modified version used the popular COCOMO by considering the effect of methodology to estimate effort. Developed model performance was tested on NASA software project events. Developed model was able to provide appropriate estimation capabilities [25].

Particle SWARM Optimization algorithm is inspired by social behavior of bird population. The main advantage of PSO is its quick convergence which is comparable with many global optimization algorithms such as genetic and firefly algorithms. PSO enjoys great similarities with complete calculation techniques such as genetic algorithm. This system starts with a population of solutions and random search for being optimized. Initialization is met by generation updating. Nevertheless, despite genetic algorithm, PSO does not have any evolutionary operator such as mutation and crossover. In PSO, potential solutions are called particles and flight is going on in question space by following optimum particles. Sheta et al. used soft calculation techniques to construct an appropriate model structure in order to estimate software effort better for NASA software projects. To this end, they used particle SWARM optimization to regulate COCOMO parameters. Also, they programmed a collection of linear models based on software code lines to apply fuzzy logic advantages. They evaluated proposed model performance using NASA software project data collection [26].

Salaria et al. introduced a new combined model of Bayesian network and PSO to estimate effort. They, in the past, had proved that Bayesian network with PSO provides more accurate results in comparison with other methods. To facilitate, NASA93 data collection was used to study the model and proposed model was also compared with COCOMO and Bayesian neural network. The results show that developed model provides better estimation compared to other models.

Recently, Rao et al. introduced a model to estimate software cost using multi-purpose particle swarm optimization. According to objectives of medium prediction of absolute and relative error intensity, model parameters were adjusted by multi-purpose particle swarm optimization. COCOMO data collection was used to test. After comparison, they proved that developed model with multi-purpose particle swarm optimization provides better results compared to standard COCOMO [18].

In 2013, Rao et al. proposed particle swarm optimization method, acting on data collection using k-means clustering algorithm. PSO uses COCOMO to produce parameters for each set of data values. Propagation technique is used for neural network training data. COCOMO 81 data collection is used to test. Also, the results of standard COCOMO are compared with those of neural fuzzy one. The results revealed that neural networks with efficient adjustment of parameters by PSO factor in clustering can lead to better results [19].

4- Bee Colony Algorithm

Optimization is one of the most important issues in expert and intelligent systems. The most known optimization algorithms in this regard include genetic algorithm, particle swarm algorithm, evolution differential algorithm, etc. Most of the optimization algorithms are inspired by the social behavior of some living creatures included in swarm intelligent domain. For instance, bee colony optimization algorithm, ant colony optimization algorithm, particle swarm algorithm, and some other optimization algorithms are included in this group.

Carboga (2005) introduced bees colony algorithm as one of the new algorithms inspired from bees group behavior [20]. This method modeling the seeking collective behavior of bees for maximizing the available honey in hive tries to solve optimization problems through real numerical parameters.

In the proposed method, each particle is a sign of food resource or bee. Different bees of each colony are classified into 3 groups according to their role including worker bees moving toward food resources and seeking locally, observer bees selecting food resources according to the dance of worker bees, and administrator or watch bees picking food resources randomly and according to some internal motivations or external signs.

The colony is incorporated by worker and observer bees.

In each repetition, the algorithm stages are as follows

1-Setting theinitial population: generating Sn real n-dimensionalrandomvectorasworker beesorfoodresources, $\{X_{i,1}, X_{i,2}, \ldots, Xi, n\}$- Xindicates the i_{th}food resources generated as follows:

$$X_{i,j}=X_{min,j}+rand(0,1)(X_{max,j}- X_{min,j}) \qquad (4)$$

In which $X_{min,j}$ and $X_{max,j}$ are the low and high limit of the j_{th} parameter or dimension. The efficiency of each vector should also be estimated. N is the size of the population.

2- Worker seeking bees in food resource space: in this stage, each X_i worker bee generates a new food resource (V_i) in its current neighborhood according to the following seeking equation:

$$V_{ij}=X_{i,j}+\phi_{i,j}(X_{i,j}-X_{k,j}) \qquad (5)$$

In which k and j are random indices and ϕ is random number in [0,1] interval. Each of the V_i and X_i is replaced with the one with more efficiency in population.

3- Observer bees: after worker bees finish their seeking activities, bees share their information about honey (efficiency) and their status with observer bees. Observer bees evaluate all received information about honey selecting a food resource according to the probability of the amount of honey. This probabilistic selection depends on the solution efficiency in population operated through roulette wheel mechanism according to which the available honey in each food resource depends on the proportion of the efficiency level to the total level of all efficiency.

Roulette wheel mechanism operates according to a Formula 6:

$$P_i = f_i / \sum_{j=1}^{N} f_j \qquad (6)$$

In which f_i is the efficiency of the i_{th} resource so that the higher the p_i, the higher the probability of i_{th} food resource selection.

In order to select a food resource, a random number is selected in [0,1] interval. For each resource whose corresponding probability is a higher number, the observer bees changes the position of that resource according to the (5) formula, and like worker bees, it is replaced with the one with more efficiency.

4- Administrative bee stage: if one food resource is not improved after some limited predetermined repetition, the observer bee should abandon the position and seek the food in other areas so that the corresponding observer bee is removed and replaced with one administrative bee according to (4) formula.

In bee colony algorithm and in each repetition, only one position is abandoned and only one worker bee is replaced with one administrative bee. The number of the swarm individuals does not change in this system [29].

5-The proposed method

The information related to the proposed method is presented in this section. The present synthetic approach tries to obtain the best estimation as much as possible through suitable selection of attributes and COCOMO coefficient optimization. The procedure of synthesizing different sections of the proposed model is carried in a way that the estimation error decreases. The proposed method consists of two sections including model training section and model testing section elaborated in the following section.

A- Model training section

The proposed method is generated and configured in the training section. The parameters of the proposed method, a and b parameters in COCOMO are indeed set in this section. It means that the goal is to produce the optimized coefficient. All projects, at first, are randomly classified into two groups of training and testing. Training data should be classified into three groups including developed, organizational, and semi-detached according to the COCOMO formula. Because of the heterogeneity of COCOMO projects, in the proposed model projects are classified. Indeed, the proposed model tries to increase the estimation accuracy locally and in a detailed manner. Therefore, all activities of projects are repeated. Since the quality of data may not be high in these three groups, a suitable normalization method is applied. Through normalization, training improves and data quality increases. (7) formula is used for data normalization.

$$P_i = \mathrm{f_i} / \sum_{i=1}^{N} \mathrm{f_i} \qquad (7)$$

In this formula, the minimum value of each attribute is subtracted from the attribute and then divided to the result of the subtraction of the minimum value from the maximum one. Therefore, each attribute is put in the range of 0 to 1. In the next stage, one group is selected from different developed, organizational, and semi-detached projects. After normalization in the first stage, the available projects effort in the selected group is estimated through bee algorithm and the proposed coefficients. It should be note that in addition to a and b coefficients, the bee algorithm generates 15 other coefficients in the range of 0 and 1 each of which are related to one attribute. These 15 attributes are indeed the same cost drivers in COCOMO equation. The coefficient of suitable cost drivers are selected through theses 15 attributes so that the coefficient more than 0.5 indicates selection while the coefficient less than 0.5 shows non-selection of cost drivers. Available projects effort should be estimated through COCOMO's (8) formula after identifying a and b coefficients and 15 other coefficients by bee algorithm.

$$\mathrm{MM} = \mathrm{a} \times (\mathrm{Size})^{\mathrm{b}} \times \mathrm{EAF} \qquad (8)$$

In fact, the effort of all available projects in the study group is estimated through the proposed coefficients. Performance parameters of one group are evaluated after the project estimation of that group finishes. Performance parameters here are MMRE and PRED. For MMRE and PRED's estimation, first MRE should be estimated through (9) formula [16]. After MRE's estimation for all projects, MMRE and PRED values are estimated through (10) and (11) formula. The value of MMRE value equals to the obtained value means for MRE in the study group. The PRED value equals to 100% of projects whose MRE value is equal or less than X.

$$\mathrm{MRE} = \frac{|Estimated - Actual|}{\mathrm{Actual}} \qquad (9)$$

$$\mathrm{MMRE} = \frac{\sum_{i=1}^{N} \mathrm{MRE}}{\mathrm{N}} \qquad (10)$$

$$\mathrm{PRED(x)} = \frac{\mathrm{A}}{\mathrm{N}} \qquad (11)$$

In (11) formula, A is the number of projects with MRE less or equal to X, and N is the number of the estimated projects. The acceptable level of X in methods of software effort estimation is 0.25 according to which the proposed methods are compared. MMRE as the total value of the error should be minimized whereas PRED (0.25) should be maximized [30].

After obtaining MMRE and PRED, the goal function should be estimated through (3-9) formula in bee optimization algorithm. It should also be minimized since the goal is to minimize the MMRE and to maximize the PRED. Obviously, the goal is fulfilled through minimizing the merit function according to (12) formula since the value of MMRE is minimized and the value of PRED is maximized.

$$\text{Fitness Function} = \mathrm{MMRE}\text{-}\mathrm{PRED} \qquad (12)$$

Finally, the bee algorithm proposes a and b coefficients several times through goal function. This continues until the output condition of bee algorithm is obtained. When algorithm reaches the output condition, it means that the optimized coefficients are indeed estimated, and since the optimized coefficient are obtained, coefficients are stored to be applied in the testing section of COCOMO. These coefficients are selected depending on the groups, and this process is repeated for all groups. All these processes are illustrated in (1) figures. As indicated by the figure, the synthesis of optimization algorithm and COCOMO is independent from the kind of the algorithm, and the proposed model running the estimation process is adaptable with different kinds of optimization algorithms. The following section elaborates on the application of the training stage results in the testing stage.

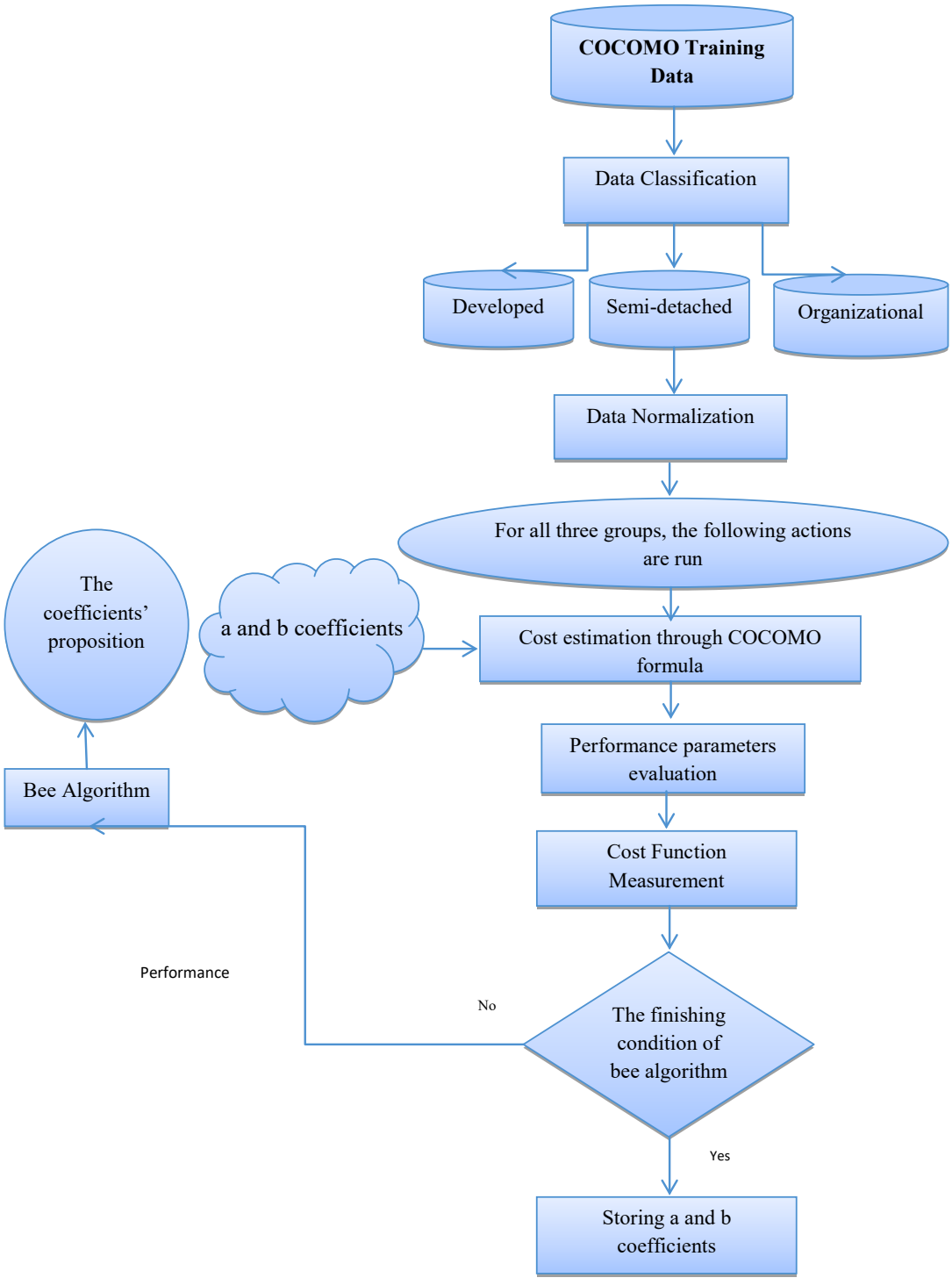

Figure 1: Training stage in the proposed model

B- Testing part in the proposed model

In this stage, we made use of the results obtained from training stage to evaluate the proposed model. Data used in this section are testing projects, and optimization projects are used for estimation. First, one of the testing projects is selected. Then, it is identified that this project belongs to which class of COCOMO projects (developed, organizational, and semi-detached). After class identification of the project, the related coefficients of the study class are extracted. It should be mentioned that

these coefficients are stored in training stage recoverable now. For example, organizational coefficients are allocated to the organizational projects and the effort of the selected project is obtained through (8) COCOMO meaning that effort estimation process is run through applying a and b and other 15 coefficients stored in the previous stage. In this stage, the obtained effort for the study project is stored, the new project is selected from testing stage, and the process is repeated for effort estimation of the new project.

This process is repeated for all available projects in testing set. Then, the MRE related to testing projects is estimated i.e. there would be MRE per number of estimated testing projects. MMRE is obtained from the mean of the MREs. Then, in evaluation part, MMRE and PRED are estimated. All these processes are shown in (2) figure.

As indicated by 2 figure, project classification of training stage is applied in testing stage so that each project enjoys having specific coefficients based on its nature.

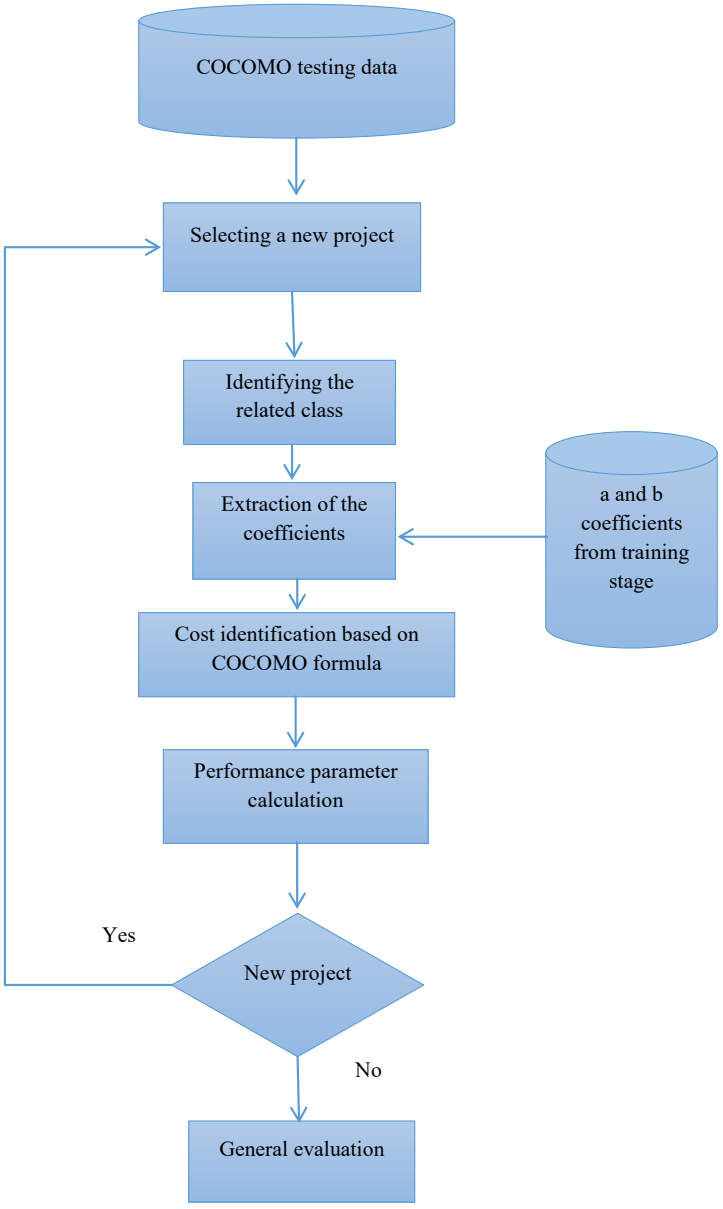

Fig 2: testing stage in the proposed model

6- Results evaluation

In this section, real performance of the proposed method is evaluated applying the data related to the real projects.

A- Evaluation process

10-fold method is one of the most common methods used for model results evaluation. This method divides initial data randomly into 10 parts. In each part, one part is applied as testing set and others as training sets i.e. error parameters are estimated in each step known in the present study as MMRE and PRED. The process of dividing sets is repeated 10 times meaning that 10 MMRE and 10 PRED are finally estimated whose mean is considered as the ultimate result.

For instance, if there are 1000 projects, 100 projects are used as testing and the remaining 900 as training. 100 estimation is obtained from this mixed with other projects and another 100-projest set is selected. This process is repeated 10 times totally resulting in 10 MMRE and PRED. the mean of MMRE and PRED is then estimated.

B- results presentation and analysis

Table 5 shows the results related to different kinds of developed projects. In this table, a and b coefficients, MMRE and PRED performance criteria, and the subtraction results of MMRE and PRED performance criteria for 10-fold are presented. As indicated by the results, a's coefficient range is from 3 to 3.1096 and b's coefficient range is from 1.1578 to 1.1865.

MMRE's range for these projects is from 0.10046 to 0.77801. The best MMRE for 10-fold is 0.10046. The difference of the minimum and the maximum MMRE is 0.67755, and their mean is 0.31697. PRED's range is from 0 to 1 with their difference of 1 and mean of 0.6. The best PRED for fold is 1. The best result for fold was obtained -0.89945.

Table 6 shows the obtained results for organizational projects. As indicated by the results, a's coefficient range is from 4.3305 to 7.2181 and b's coefficient range is from 0.74079 to 0.89176. MMRE's range for these projects is from 0.016733 to 0.60677. The best MMRE for 10-fold is 0.016733. The difference of the minimum and the maximum MMRE is 0.590037, and their mean is 0.22953. PRED's range is from 0 to 1 with their difference of 1 and mean of 0.6. The best PRED for fold is 5, 6, 7, and 10. The best result for 10-fold case was obtained -0.98327.

Table 5- Results related to different kinds of developed projects

	a	b	MMRE	PRED	MMRE- PRED
Fold1	3	1.1865	0.16391	1	-0.83609
Fold2	3	1.1864	0.22761	0.66667	-0.43905
Fold3	3.1096	1.1578	0.37779	0	0.37779
Fold4	3	1.1866	0.77801	0.66667	0.11134
Fold5	3	1.1865	0.28323	0.66667	-0.38344
Fold6	3.0429	1.1856	0.23447	0.66667	-0.43219
Fold7	3	1.1864	0.31516	0.33333	-0.018173
Fold8	3	1.1864	0.51995	0	0.51995
Fold9	3	1.1864	0.16906	1	-0.83094
Fold10	3	1.1865	0.10046	1	-0.89954
AVG	------	------	0.31697	0.6	------

Table 6- Obtained results for organizational projects

	A	B	MMRE	PRED	MMRE- PRED
Fold1	5.9115	0.79848	0.60677	0	0.60677
Fold2	5.9986	0.79198	0.46869	0	0.46869
Fold3	4.3305	0.89176	0.29865	0.33333	-0.034685
Fold4	7.2181	0.74079	0.16455	0.66667	-0.50212
Fold5	5.94	0.79591	0.028316	1	-0.97168
Fold6	5.9597	0.79392	0.066924	1	-0.93308
Fold7	5.9625	0.79455	0.12468	1	-0.87532
Fold8	6.0428	0.78984	0.2404	0.5	-0.2596
Fold9	5.9388	0.79551	0.27955	0.5	-0.22045
Fold10	6.0935	0.7873	0.016733	1	-0.98327
AVG	------	------	0.22953	0.6	------

Table 7 shows the results related to different kinds of semi-detached projects. As indicated by the results, a's coefficient range is from 206748 to 2.7938, and b's coefficient range is from 1.1318 to 1.1653. MMRE's range for these projects is from 0.008202 to 1.1545. The best MMRE for 10-fold is 0.008202. The difference of the minimum and the maximum MMRE is 1.146298, and their mean is 0.25645. PRED's range is from 0 to 1 with their difference of 1 and mean of 0.7. The best PRED for fold is 2,3,4,5,8,9 and 10. The best result for 9-fold case was obtained -0.9918.

Table 7- Results related to different kinds of semi-detached projects

	a	b	MMRE	PRED	MMRE- PRED
Fold1	2.6753	1.1653	1.1545	0	1.1545
Fold2	2.7578	1.1586	0.12522	1	-0.87478
Fold3	2.7648	1.1581	0.08659	1	-0.91341
Fold4	2.6748	1.1653	0.027362	1	-0.97264
Fold5	2.7453	1.1595	0.2507	1	-0.97493
Fold6	2.7938	1.1318	0.32619	0	0.32619
Fold7	2.7665	1.1578	0.57282	0	0.57282
Fold8	2.7893	1.156	0.035492	1	-0.96451
Fold9	2.7621	1.1581	0.0082022	1	-0.9918
Fold10	2.7665	1.1578	0.20314	1	-0.79686
AVG	------	------	0.25645	0.7	------

C-Results evaluation

The results related to the application of the proposed method are presented in Tables 4, 5, and 6. As indicated by the results, the best PRED value was obtained for semi-detached projects indicating the influence of coefficients optimization on increasing the percentage of estimation in these projects. The difference between the obtained values for PRED in these projects relative to the other two kinds is clear. The best value of MMRE is obtained for organizational projects probably caused by large number of these projects. The important point is the notable difference of the proposed coefficients for all three kinds of projects caused by projects variation.

Table 8 presents the obtained results through bee colony algorithm without driver selection presented for testing projects. These results are related to 3 kinds of software projects including developed, organizational, and semi-detached. As indicated by the results, the best MMRE is obtained 0.22953 for organizational projects, the most MMRE is 0.31697 for developed projects, and finally the MMRE mean is 0.26765. The best PRED is obtained 0.7 for semi-detached projects, the least PRED is obtained 06 for developed and organizational projects. The PRED mean is 0.63 with the difference of 0.1. Finally, the best result was obtained -0.44355 for semi-detached projects. These results are presented in Figure 1. As indicated by the results, the best results of PRED and MMRE occur in semi-detached and organizational projects, respectively.

Table 8-The obtained results through bee colony algorithm presented for testing projects

Project type	MMRE	PRED	MMRE-PRED
em	0.31697	0.6	-0.28303
org	0.22953	0.6	-0.37047
sem	0.25645	0.7	-0.44355
AVG	0.26765	0.63	-0.36235

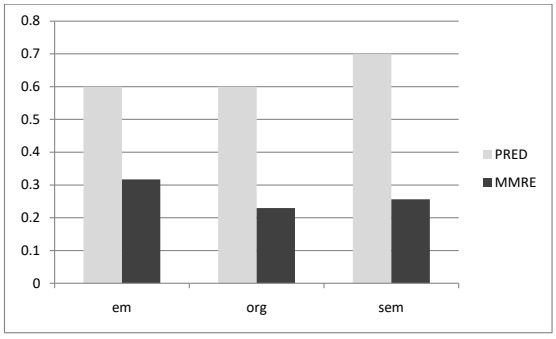

Fig 3: Results in three modes of projects

7- Conclusion

Accurate estimation of software development plays a critical role in software project management since it considerably influences project planning and management. Most of the software development costs are caused by human effort, and most of the cost estimation methods focus on person-month based estimation.

Nowadays, various models for are proposed software development cost estimation. Managers select these models based on the kind of the project. COCOMO is one of the most common algorithmic models applied for time and cost estimation in software projects. This model was introduced through constant parameters and applying regression analysis of statistical data to 63 kinds of software projects.

Although being a favorable and widely accepted estimation model, COCOMO cannot be effective for most of the software projects. Parametric models of software cost estimation like COCOMO are based on mathematic formula. Most of software projects data sets are heterogeneous which makes achievement a unified parametric model inapplicable for an expanded range of software project. The effective selection of parameters of COCOMO according to projects' nature can be considered as one solution for this problem.

The main objective of the present paper was to present a new version of COCOMO 81 model according to coefficient optimization on 63 software projects with 15 attributes. In order to optimize coefficients, this study made use of bee colony algorithm applied for effort estimation of software development. In this study, NASA COCOMO dataset was used for the performance evaluation of the proposed method. Finally, it was clear that the COCOMO coefficients obtained through bee colony algorithm were improved considerably compared to the basic COCOMO coefficients.

REFERENCE

[1] Pressman, R. S. Software Engineering: A Practioner's Approach (6th ed.): McGraw-Hill New York, USA. (2005)

[2] Dhiman A, Diwaker C. Optimization of COCOMO II Effort Estimation using Genetic Algorithm. American International Journal of Research in Science, Technology,Engineering & Mathematics. 2013:208-212.

[3] .Li Y-F, Xie M, Goh TN. A study of project selection and feature weighting for analogy based software cost estimation. Journal of Systems and Software, 2009;82(2):241-252.

[4] Fei Z, Liu X, f-COCOMO: fuzzy constructive cost model in software engineering.IEEE International Conference on Fuzzy Systems; 1992;331-337.

[5] Idri A, Abran A, Khoshgoftaar TM, Estimating software project effort by analogy based on linguistic values. Eighth IEEE Sympoium on Software Metrics; 2002:21-30.

[6] Xu Z, Khoshgoftaar TM. Identification of fuzzy models of software cost estimation. Fuzzy Sets and Systems. 2004;145(1):141-163.

[7] Sheta A, Aljahdali S. Software effort estimation inspired by COCOMO and FP models: A fuzzy logic approach. International Journal of Advanced Computer Science and Applications. 2013;4(11):192-197.

[8] Patil LV, Shivale NM, Joshi S, Khanna V. Improving the accuracy of CBSD effort estimation using fuzzy logic. International Advance Computing Conference (IACC), 2014;1385-1391.

[9] Moløkken K, Jørgensen M. Expert estimation of web-development projects: Are software professionals in technical roles more optimistic than those in non-technical roles? .Empirical Software Engineering. 2005;10(1):7-30.

[10] Boehm BW, Valerdi R. Achievements and challenges in cocomo-based software resource estimation. Software, IEEE. 2008;25(5):74-83.

[11] Khatibi V, Jawawi DN. Software Cost Estimation Methods: A Review, Journal of Emerging Trends in Computing and Information Sciences, 2010;1(2):21-29.

[12] Keung JW, Kitchenham BA, Jeffery DR. Analogy-X: providing statistical inference to analogy-based software cost estimation. IEEE Transactions on Software Engineering, 2008;34(4):471-484.

[13] Yu W-d, Lai C-c, Lee W-l. A WICE approach to real-time construction cost estimation. Automation in Construction. 2006;15(1):12-19.

[14] Kim G-H, An S-H, Kang K-I. Comparison of construction cost estimating models based on regression analysis, neural networks, and case-based reasoning. Building and environment. 2004;39(10):1235-1242.

[15] Ismaeel HR, Jamil AS. Software Engineering Cost Estimation Using COCOMO II Model.2007

[16] Khatibi E, Investigating the effect of software project type on accuracy of software development effort estimation in COCOMO model. Fourth International Conference on Machine Vision (ICMV); 2011;8(3):60-70.

[17] Idri A, Zakrani A, Zahi A. Design of radial basis function neural networks for software effort estimation. International Journal of Computer Science Issues. 2010;7(4):1-10.

[18] Rao GS, Krishna CVP, Rao KR, Multi Objective Particle Swarm Optimization for Software Cost Estimation. ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of CSI, 2014;1(5);125-140.

[19] SinghSandhu G, Singh Salaria D. A Bayesian Network Model of the Particle Swarm Optimization for Software Effort Estimation. International Journal of Computer Applications. 2014;96(4):52-58.

[20] Karaboga D. An Idea Based on Honey Bee Swarm For Numerical Optimization. Technical Report-TR06. 2005.

[21] Rao BT, Sameet B, Swathi GK, Gupta KV, RaviTeja C, Sumana S. A novel neural network approach for software cost estimation using Functional Link Artificial Neural Network (FLANN). International Journal of Computer Science and Network Security. 2009;9(6):126-131.

[22] Reddy CS, Raju K. A concise neural network model for estimating software effort. International Journal of Recent Trends in Engineering. 2009;1(1):188-93.

[23] Berry MJ, Linoff G. Data mining techniques: for marketing, sales, and customer support, 1997.

[24] Singh BK, Misra A. Software Effort Estimation by Genetic Algorithm Tuned Parameters of Modified Constructive Cost Model for NASA Software Projects. International Journal of Computer Applications. 2012;59(9):22-36.

[25] Bardsiri VK, Jawawi DNA, Hashim SZM, Khatibi E. A PSO-based model to increase the accuracy of software development effort estimation. Software Quality Journal. 2013;21(3):501-526.