Abstract—The estimation of software effort is an essential and crucial activity for the software development life cycle. Software effort estimation is a challenge that often appears on the project of making a software. A poor estimate will produce result in a worse project management. Various software cost estimation model has been introduced to resolve this problem. Constructive Cost Model II (COCOMO II Model) create large extent most model has been introduced to resolve this problem. Constructive in a worse project management. Various software cost estimation project of making a software. A poor estimate will produce result and crucial activity for the software development life cycle. The proposed method is applied on Nasa93 dataset. The experiment result of proposed method able to reduce error down to 11.891% and 8.082% from the perspective of COCOMO II model. The method has achieved better results than those of previous researches and deals proficient with inexplicit data input and further improve reliability of the estimation method.

Keywords—COCOMO II Model; Effort Estimation; Time Development Estimation; Fuzzy; Multi-Objective PSO; Optimization.

I. INTRODUCTION

Software Development is a systematic approach of Software Engineering discipline in constructing and maintaining of software system. The software project manager is a person who responsible for control software development in all activities. The main goal of the software project manager is to make sure that the project is accomplished with the concept of “high quality of software should be produced with less cost concern within time and budget given”. Estimation of software cost is the major challenge in software project development. The accuracy of estimation is vital to guide software companies in making good management to develop a software. Moreover, good management of software development can estimate the cost and resources of software precisely. It is calculated in term of person-month and it can handle both overestimates and underestimates of software effort and cost. This accuracy derives from some variables or cost drivers. So, obtaining an accurate of software cost estimation needs accurate prediction method.

Several cost estimation methods have been proposed and improved by many researchers in the last few decades. These methods are categorized into Expert judgement, Algorithmic method and Analogy based method. Constructive Cost Model (COCOMO) is the most well-known among all the software estimation model and widely used in calculate the software cost. Currently, many issues have arisen regarding the applicability of these methods to solve the software cost estimation. Heuristic techniques are used to overcome the limitation of these methods and improve the applicability [1]. Various heuristic optimization methods are used in optimization problems. These methods can be used in the software cost estimation also. These methods are Particle Swarm Optimization [6, 11, 12, 13, 15, 17], Multi-Objective Particle Swarm Optimization [16, 18], Genetic algorithm [13, 26], Firefly Algorithm [14], and many others.

This paper presents a utilize of applying Fuzzy Logic and Multi-Objective Particle Swarm Optimization (MOPSO) as calibration and an optimization algorithm in optimizing the COCOMO II model parameters, so that a more realistic and accurate effort can be estimate. The remaining paper is organized as follows: In Section II, literature review, brief introduction of COCOMO model and basic principle of the methods discussed in this paper. In Section III, it describes related works that have been researched. Section IV explain the methodology steps of work with used in this experimentation. Section V, describe the evaluation criterial and dataset. In Section VI presents experimental and results comparison. And Section VII, concludes the study.

II. LITERATURE REVIEW

A. Software Cost Estimation

Software project management require reliable software cost estimation to make judgement the amount of effort and resources in create a software. The accuracy of cost estimation is significant in developing software. Estimating at the early stages can help to manage the planning, budgeting and monitoring the activities of a project. Because there are a limited number of resources for a project, accurate of software
estimation can provide sufficient support for the decision making process with efficiently and effectively. However, the most difficult problem to estimate the software cost is the obstacle of uncertainty data and the complicated that make bad effect to software development process. So, there are some techniques and procedures to handle this issue. Both algorithmic method and non-algorithmic method can help to estimate software cost. Algorithmic method usually use linear regression method and collection of previous data in prediction. Non-algorithmic method tries to construct rules that fit to the data. These include analogy method, artificial neural network, fuzzy, and genetic algorithm. Cost estimation is usually measured in terms of effort and time development. The effort is the amount of time one person to work for a definite period of time. And time development is the number of month a project is scheduled. Normally, more efforts and time development are used, more expensive cost will be.

B. COCOMO II Model

Various software cost estimation methods have been proposed for helping project manager to do estimating and making correct decision in building the software system with high quality accurate in estimation [8]. Constructive Cost Model so-called COCOMO has become one of the most valuable and broadly used cost estimation models. COCOMO was published by Barry Boehm in 1981 [9]. Effort and schedule estimation models are two main models deliver in COCOMO for software management. The model was developed from the dataset that consisted of 63 projects. Each project was divided into 16 variables. COCOMO divided cost driver into 3 aspects such as Effort Multiplier (EM), Line of Code (LOC) and Scale Factors (SF). All the cost drivers will be calculated with an equation to produce the number of effort in person-months (PM) and time development (TDEV). In 2000, Barry Boehm [10] introduced COCOMO II model which has been provided more accurate with some aspect of improvement in several cost drivers. The Post model gets more attention by researchers as it involves the actual development and maintenance of the software product. The Post model model is used to estimate the effort and development time.

1) Effort Estimation Model. COCOMO II model [7] the Equation that used for calculating the software development effort is given in Equation (1) and Scales Factors Computation is defined by Equation (2):

\[ Effort (PM) = A \cdot Size^B \times \prod_{i=1}^{17} EM_i + PM_{Auto} \] (1)

\[ E = B + 0.01 \times \sum_{i=1}^{5} SF_j \] (2)

where A and B are the multiplicative and exponential constant, have value 2.94 and 0.91. Size is estimated size of a project in Kilo Source Lines of Code (KLOC), E define scaling exponent for effort, it is an exponential factor which has a record of accounts for the associate with economies or diseconomies of scale extendable as the software project size increases, \( EM_i \) is the Effort Multipliers where \( i = 1 \) to 17 and \( SF_j \) is Scale Factors where \( j = 1 \) to 5. There are two constants for schedule calculation. Multiplicative constant C is schedule coefficient, has value 3.67. And exponential constant D is scaling base-exponent for schedule that has value 0.28.

2) Schedule Estimation Model. The Equation that used for calculating the development time (TDEV) is given in Equation (3) and its effort multipliers is defined in (4):

\[ TDEV = \left[C \times (PM_{NS})^F\right] \times \frac{SCED\%}{100} \] (3)

\[ F = D + 0.2 \times [E-B] \] (4)

where C and D are the multiplicative and exponential constant of development time, have value 3.67 and 0.28. F defines the scaling exponent for Schedule.

A, B, C and D are called COCOMO II Model coefficients or parameter. The propose of this paper is to optimize four variations of COCOMO II model parameters effort calculation and schedule calculation for better improvement of the model using Fuzzy Logic and MOPSO for NASA dataset.

C. Fuzzy Logic

Fuzzy logic (FL) is the term given to a system of mathematics developed to model the human brain’s curious way of processing words. It was originally proposed by Zadeh in year 1965 [22]. The main objective behind FL is the existence of imprecision in the measurement process. Zadeh explains that “As complexity rises, precise statements lose meaning and meaningful statements lose precision” [22]. Fuzzy logic provides capabilities that allow handling both quantitative and qualitative data within one model. It is a form of multivalued logic derived from fuzzy set theory to deal with reasoning that is approximate rather than precise. Fuzzy sets are sets whose elements have degrees of membership [23]. Several membership functions in Fuzzy logic, they are triangular, trapezoidal, Gaussian and many others. In this study, we investigate and application of Gaussian Membership Functions.

Fuzzy Logic System (FLS) is the method given to the system that consist of relationship with fuzzy and fuzzy logic principles. Most famous FLS can be classified into three types [24]: Original FLS, Takagi and Sugeno’s fuzzy system, and FLS with fuzzifier and defuzzifier. Most of the applications engineering create input using crisp data produce crisp data as output. FLS with fuzzifier and defuzzifier widely used one where the fuzzifier maps crisp inputs into fuzzy sets and the defuzzifier maps fuzzy sets into crisp outputs. Fig 1 illustrated of logic system proposed by Mamdani [25].
D. Multi-Objective Particle Swarm Optimization (MOPSO)

1) Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is a swarm intelligence algorithm based on nature-behavior inspire. The PSO was presented in year 1995, by Kennedy and Eberhart [3]. Because of the simplicity, durability, and flexibility of the PSO algorithm, it has become one of main well-known and broadly used swarm intelligence based algorithms. It uses randomness with real number, and local and global communication among the particles of swarm [2]. The PSO algorithm use to search the space of an objective function by adjusting the movement of individual object call “particles”. Each particle is tries to move toward the position to the global best $f^{(g)}$ and its personal best $x^{(p)}$ according to best experienced. When a particle of swarm finds a position that is better than any previously found position, it will update the position as the new current best for particle i. After several number of iteration or the objective is no longer move and improve, the purpose of finding the global best value, $f^{(g)}$ can also include any equalities, because an equality $\Phi(x) = 0$ can be converted into two inequalities $\Phi(x) \leq 0$ and $\Phi(x) \geq 0$. However, for clarity, here we list the equalities and inequalities separately [2] [5].

In the real-world problems always involve the optimization of two or more objectives. A multi-objective optimization, does not necessarily have an optimal solution that minimizes all the multi-objective functions simultaneously and the optimal parameters of some objectives usually do not lead to the optimality of other objectives. [4]. Therefore, among these always conflicting objectives, we must choose some tradeoff or achieve a certain balance of objectives. We must compare different objectives and make a compromise. This usually requires a reformulation, and find a scalar-valued function that represents a weighted combination or preference order of all objectives [2].

To convert the single-objective PSO, every objective has its own weight, we should combine the objectives into single weighted formula:

$$F(x) = w_1 f_1(x) + w_2 f_2(x) + \cdots + w_M f_M(x)$$

and normalize the weights sum method using

$$\sum_{i=1}^{M} w_i = 1, w_i \in (0,1)$$

III. RELATED WORK

There are various on previous works in optimizing and trying to improve the accuracy and calibrated the parameters value of COCOMO.

Riyanto and Johannes [19] [20] investigated the role of Effort Multiplier (EM) and Line of Code (LOC) to utilized the effort estimation. Gaussian Membership Function (GMF) [9] has been applied to the COCOMO II to represent the EM. GMF could makes a smoother transition which means a more accurate Effort Multipliers. And they also applied Neural Network (NN) approach [20]. The proposed model shows a major improvement rather than pure Fuzzy model or basic COCOMO model. Baiquni and Riyanto [21] proposed model based on Fuzzy Logic, Local Calibration, and Tabu Search. They tried to improve accuracy by fuzzifying cost drivers in Fuzzy Logic with Gaussian Membership Functions (GMF) to redesigned the Effort Multiplier. And Local Calibration as
In this study, the rules is formulated as following:

unchanged

need. FIS Editor allows us to create rules from input to output.

any range and any number of membership function that we

implemented using fuzzy logic tool box in MATLAB software.

This FIS Editor GUI helps us to create input and output with

transition from one level to another level. Fuzzy Logic was

multiplier has ran ge from Very Low to Very High. The

Because it is derived from LTEX effort multiplier. For example, low level has
description a very low multiplier. The LTEX description of every levels is translated
into GMF. For example, LTEX has description a very low rating i s given for experience of less than 2 months and very high rating is given for experience of 6 or more years so we draw the low interval to less than 2 and so on.

The proposed approach of MOPSO is used to accommodate the Fuzzy COCOMO effort and development time estimation. This approach is required 17 Effort Multipliers, 5 Scale Factors, Actual Effort, and month as development time. MOPSO act as a global optimization technique. It is applying to investigate and resolve unpredictable input and optimize the parameters coefficient relating to the effort and produce the result in less execution time significantly.

There are steps of MOPSO in optimizing parameters as follow: Step 1: Initialize m particles by randomly position and velocity vectors [p_1, p_2… p_m] and [v_1, v_2…v_m] accordingly for parameters use to optimized, Step 2: Initialize every particle as P_best particles, Step 3: Rate the fitness functions f_1(x), f_2(x) using Equations (1), (3), (9), and (10) for every particle. The goal of f_1(x) is to minimize and goal of f_2(x) is to maximize, Step 4: Convert from Multi-Objective form into Single-Objective form using weighted sum method. For each two

Calico and Tabu Search used to search the value of parameters and gives the new value for the parameters calculation of COCOMO II model. The new value able to improve the accuracy and decreasing error significantly.

Prasad Reddy et al. [16] and Ruchi Puri [17] proposed Multi Objective Particle Swarm Method (MOPSO) model for software estimation. The proposed model gives better results when compared with the standard COCOMO model and it is also observed, when provided with good classification among training data may give more better results. Satapathy et al. [18] presented the use of MOPSO for software cost estimation with COCOMO model. They were show that the results observed by using MOPSO gives better results. They show the testing of performance of the proposed model in terms of the MARE error and the results were found to be useful and provide more accurate.

IV. METHODOLOGY

There are various of uncertainties in effort and development time estimation using basic COCOMO II model parameters. Our focus is on optimizing the multiplicative and exponential constants parameter A, B, C, and D of COCOMO II model. In this paper, the methodology in optimizing parameter of the COCOMO II model is optimized using Gaussian Membership Function (GMF) of Fuzzy Logic and MOPSO.

A. Fuzzy Logic

In this section, the steps of the proposed Fuzzy Logic approach is presented. The method of Fuzzy Logic is based on the research of [19] and [21]. This study tries to learn effort multipliers in COCOMO II Model. Each effort multiplier (EM) uses linguistic values to represent the character of each EM. The cost drivers are in linguistic values and ranged from Very Low to Extra High. This research divides EM into two categories: qualitative and quantitative EM. The quantitative effort multipliers are DATA, CPLX, RUSE, DOCU, TIME, STOR, PVOL, ACAP, PCAP, PCON, APEX, PLEX, LTEX, TOOL, SITE, and SCED and another are quantitative EM. Fuzzy Model is used to redesign the quantitative EM because quantitative EM description can be translated into Fuzzy Logic. For example, Language and Tool Experience (LTEX) effort multiplier has range from Very Low to Very High. The difference for every level is percentage use of available execution time. This study uses Gaussian Membership Function (GMF) for Fuzzy Logic. GMF creates a smoother transition from one level to another level. Fuzzy Logic was implemented using fuzzy logic tool box in MATLAB software. The tool box is named as Fuzzy Inference System (FIS) Editor. This FIS Editor GUI helps us to create input and output with any range and any number of membership function that we need. FIS Editor allows us to create rules from input to output. In this study, the rules is formulated as following:

R1 : IF Input LTEX is low THEN Output data is increased

R2 : IF Input LTEX is nominal THEN Output data is unchanged

and so on....
objectives give ranks for every particle. Insert the ranks of objectives and assigned to each particle. The final fitness is minimized values, Step 5: If the fitness of particle (p) better than the fitness Personal Best \((P_{\text{p}}}\) then Personal Best \((P_{\text{p}}}\) = Particle \((p)\), so set the best of Personal Best \((G_{\text{best}})\) as a Global Best \((G_{\text{best}})\), Step 6: Update the particles velocity and particle position using Equations (5) and (6), Step 7: Repeat steps 4 to 8 until particles is no move and change in the objectives, Step 8: And give the Global Best\((G_{\text{best}})\) value parameters as optimal solution optimization. The result in these steps give an optimal value in optimization method. The parameter value then used to compute new better result for effort and development time of COCOMO II Model.

V. EVALUATION CRITERIA AND DATA SET

We propose to use Mean Magnitude of Relative Error (MMRE) as the fitness functions for the proposed method

The main objective of estimation method is to verify whether the predictions are precise; the gap between the predicted of effort, Estimated Effort, and the realistic actual effort, Actual Effort, should be measure as close as possible. Large values different between Actual Effort and Estimated Effort will reduce accurate of prediction and create bad effect on the effort in the software system development. In this paper, Magnitude of Relative Error (MRE) [3] apply as common criteria on software cost estimation to evaluate accuracy of estimated effort. The MRE consider to calculate for each project point as defined in Equation (9):

\[
\text{MRE}_i = \frac{|\text{Actual Effort}_i - \text{Estimated Effort}_i|}{\text{Actual Effort}_i} \times 100
\]  

(9)

Mean MRE (MMRE) [3] use to average the resulting of individual accuracy prediction value that measures in MRE criteria, giving in Equation (10):

\[
\text{MMRE} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\text{Actual Effort}_i - \text{Estimated Effort}_i|}{\text{Actual Effort}_i}
\]  

(10)

The parameters setting of program are set as in Table I. The experiments apply MOPSO in optimizing the COCOMO II model parameters based on the NASA93-dem data set. The dataset consists of data from 93 projects. Each project consists of 27 attributes which include of Project ID, 5 Scale Factor, 17 Effort Multiplier in the value interval range from VeryLow to ExtraHigh, effort as actual effort in person months, Project Size represented in lines of program source code (LOC) and months as actual development time. All project data points will be used in calibration. Result from calibration can be used for the next project from similar category.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterations</td>
<td>200</td>
</tr>
<tr>
<td>Population and Repository Size</td>
<td>200, 100</td>
</tr>
<tr>
<td>Weight Acceleration coefficient</td>
<td>[1.0, 2.0]</td>
</tr>
</tbody>
</table>

VI. EXPERIMENTATION AND RESULT

This section presents the experiment and accomplish result of applying the proposed method to the dataset. The main objective of optimization is empirically to reduce the uncertainties of COCOMO II model coefficients, parameters A, B, C, and D using Fuzzy and MOPSO technique and do comparison the obtain results with the basic coefficients. The method is implemented in MATLAB, the computed parameters can significantly simplify the estimation of the software effort for all projects. Implementation conducted in in several iterations. After several iterations, we able to obtain the new optimized parameter result A=4.3852, B=0.2830, C=2.7802 and D= 0.3615 instead of the basic COCOMO II values are A=2.94 B=0.91, C= 3.67 and D=0.28

The result of implementation is targeted to reduce MMRE errors, the smaller value of MRE or MMRE is better to closer actual effort and actual development time. For example, Project ID 9 have 58.331% and 24.326% error for effort and development time by using COCOMO II standard parameters, 53.053% and 13.305% error by using proposed method. The implementation result show that the proposed method able to reduce 5.257% and 11.021% from default COCOMO II model setting.

The MMRE of each method represent MEAN of accuracy measurement. The MMRE value of COCOMO II model, Fuzzy MOPSO as 50.584%, and 38.693% for effort estimation and 19.982% and 11.900% for development time estimation. It is mean the proposed method able to reduce error down to 11.891% and 8.082% from the perspective of COCOMO II model. The result of MMRE show that effort and TDEV estimation by the proposed method is delivered much better measurement. The MMRE value of COCOMO II model, Fuzzy MOPSO as 50.584%, and 38.693% for effort estimation and 19.982% and 11.900% for development time estimation.
VII. CONCLUSION

The challenge in achieving a reliable trust and accurate software cost estimation has been studied and improved both in software industry and academic field. The more accurate software cost estimation can handle the more software development resources efficiently. Several software cost estimation models that applicable to applied for forecast software cost. In this paper, we investigated the efficiency of applying the Gaussian Membership Function a type of Fuzzy Logic and multi-objective swarm intelligence, Multi-Objective Particle Swarm Optimization (MOPSO) as a calibration and optimization algorithm approach to improve the accurate degree of COCOMO II model by optimize its parameters. The proposed method has implemented with the NASA dataset. The method has assessed according to evaluation criteria. The proposed method gives significant in reduced MMRE and evaluation results has shown that the calibration and optimization with proposed method gives a improved estimation compared to the basic COCOMO II model.

REFERENCES