A Variant of COCOMO II for Improved Software Effort Estimation

Ziyad T. Abdulmehdi, M. S. Saleem Basha, Mohamed Jameel, and P. Dhavachelvan

***Abstract*—Accurate effort estimation is the state of art of the Software Engineering activities and of course it is a complex process. On the other hand, it is also widely accepted that due to the inherent uncertainty in software development requirements and activities, it is unrealistic to expect very accurate effort estimates over the software development processes. Among the diversified effort estimation models, empirical estimation models are found to be possibly accurate compared to other estimation schemes. The work reported in this paper aims at improving the accuracy of one of the popular effort estimation models, COCOMO II. Since the accuracy of the COCOMO II stands over the cost derives and scale factor, this work investigates the influence of the cost drives and scale factor to improve the accuracy of the effort estimation. It is proved that, with a set of possible modifications in scale factor and cost drives, the overall accuracy of the COCOMO II can be improved. The improvement has been proved in terms of the performance validation factors such as Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE), Root Mean Square (RMS) and Relative Root Mean Square (RMS & RRMS).**

***Index Terms*—Effort Estimation, COCOMO II, graphical estimation, SLIM, SEER.**

I. Introduction

The most crucial part in the software development activity is the effort estimation. For the past few decades many researches has been carried out to predict the actual effort to develop a software project. But, still it is night mare to achieve the closer result, because it involves both measurable and non measurable factors in other words we can say that it involves functional and non functional aspects of the software development process.

To allocate the recourses in terms of man and machine the effort estimation plays a vital role and also in scheduling the task. Moreover, many researchers are dedicating their precious research time and money to work on various software effort estimation models to perk up the accuracy of those software effort estimation models. Although a great amount of research time, and money have been devoted to improving accuracy of the various estimation [1]. Though there is no proof on software cost estimation models to perform consistently accurate within 25% of the actual cost and 75% of the time [2], still the available cost estimation models extending their support for intended activities to the possible extents.

The accuracy of all the models is depends on the software data of that particular project and the way how they calibrate those factors / values and of course the accuracy is an important factor that decides the applicability of the individual models in the appropriate environments. For the corporate, it is vital to maintain the precision and reliability of the effort estimation to grab the attention of the customers and also among the competitive companies.

There are many estimation models have been proposed and can be categorized based on their basic formulation schemes; estimation by expert [3], analogy based estimation schemes [4], algorithmic methods including empirical methods [5], rule induction methods [6], artificial neural network based approaches [7]-[9], Bayesian network approaches [10], decision tree based methods [11] and fuzzy logic based estimation schemes [12], [13].

There are many diversified estimation models are there. But in particular the empirical models are believed to be an accurate estimation models when compare to such other diversified models. To name some of the popular empirical estimation models from the literature are COCOMO, SLIM, SEER-SEM and FP analysis schemes are popular in practice [14], [15].

During the past decades of empirical estimation models, the estimation factors are gathered from pragmatic values obtained from several similar projects and derived an obsolete values for the parameters to find near values of the estimation. But, now a days by the use of enormous techniques and tools namely neural method, bio inspired methods, Genetic algorithmic methods, etc., the parameter's values are finetuned and comes with different names of the estimation models. Accurate effort and cost estimation of software applications continues to be a critical issue for software project managers [16].

Due to above said scenario of using various tools and techniques, it is noticed that there are more changes, updates and versions of the same model. A common modification among most of the models is to increase the number of input parameters and to assign appropriate values to them.

Despite the fact that few estimation schemes are flooded with more and more input parameters to achieve additional features of that scheme to create the credibility among the customers and competitors, in this way unknowingly they are injecting complexities into their estimation models. But fails to achieve the accuracy of their estimation schemes. Although they are diversified, they are not generalized well for all types of environments [17].

Hence there is no silver bullet estimation scheme for different environments and the available models are environment specific. Since the research focus of this paper is to refine the COCOMO II estimation scheme and to provide an improved estimation scheme, our discussion is limited with COCOMO II estimation models only. COCOMO II

Manuscript received December 16, 2013; revised April 8, 2014.

Ziyad T. Abdulmehdi, M. S. Saleem Basha, Mohamed Jameel, and P. Dhavachelvan are with the Department of Computer Science and Information Technology, Mazoon University College, Muscat, Oman (e-mail: Ziyad@mazooncollege.edu.om, saleem@mazooncollege.edu.om, mohammed.jameel@mazooncollege.edu.om).

used 31 parameters to predict effort and time [18], [19] and this larger number of parameters resulted in having strong co-linearity and highly variable prediction accuracy. Besides these meritorious claims, COCOMO estimation schemes are having some disadvantages. The underlying concepts and ideas are not publicly defined and the model has been provided as a black box to the users [20]. This model uses LOC (Lines of Code) as one of the estimation variables, whereas Fenton *et al*. [21] explored the shortfalls of the LOC measure as an estimation variable. The COCOMO also uses FP (Function Point) as one of the estimation variables, which is highly dependent on development the uncertainty at the input level of the COCOMO yields uncertainty at the output, which leads to gross estimation error in the effort estimation [22].

Irrespective of these drawbacks, COCOMO models are still influencing in the effort estimation activities due to their better accuracy compared to other estimation schemes. By considering the popularity and applicability of the COCOMO estimation schemes, the research work reported in this paper offer a new direction of approach for improving the accuracy of the COCOMO II effort estimation model. This paper is organized as follows: Section II reviews the COCOMO estimation models and analyses the performance of the COCOMO II model in particular. Section III describes the proposed model and its formulation schemes. Section IV validates the proposed model against the COCOMO II model with the standard data sets and Section V concludes the proposed research work with its potential merits.

II. Review of COCOMO Models

COCOMO 81 (Constructive Cost Model) is an empirical estimation scheme proposed in 1981 [23], [24] as a model for estimating effort, cost, and schedule for software projects. It was derived from the large data sets from 63 software projects ranging in size from 2,000 to 100,000 lines of code, and programming languages ranging from assembly to PL/I. These data were analysed to discover a set of formulae that were the best fit to the observations. These formulae link the size of the system and Effort Multipliers (EM) to find the effort to develop a software system. In COCOMO 81, effort is expressed as Person Months (PM) and it can be calculated as

$$PM = a \times Size^{b} \times \prod_{i=1}^{15} EM_i \qquad (1)$$

where "*a*" and "*b*" are the domain constants in the model. It contains 15 effort multipliers. This estimation scheme accounts the experience and data of the past projects, which is extremely complex to understand and apply the same.

In 1997, an enhanced scheme for estimating the effort for software development activities, which is called as COCOMO II. In COCOMO II, the effort requirement can be calculated as

$$PM = a \times Size^{E} \times \prod_{i=1}^{17} EM_i \qquad (2)$$

where, $E = B + 0.01 * \sum_{j=1}^{5} SF_j$

COCOMO II is associated with 31 factors; LOC measure as the estimation variable, 17 cost drives, 5 scale factors, 3 adaptation percentage of modification, 3 adaptation cost drives and requirements & volatility. Cost drives are used to capture characteristics of the software development that affect the effort to complete the project.

Cost drives have a rating level that expresses the impact of the driver on development effort, PM. These rating can range from Extra Low to Extra High. For the purpose of quantitative analysis, each rating level of each cost driver has a weight associated with it. The weight is called Effort Multiplier. The average EM assigned to a cost driver is 1.0 and the rating level associated with that weight is called Nominal.

III. Proposed Estimation Scheme

A. Two–Stage Estimation Schemes of COCOMO

From the above discussions, it is observed that the COCOMO models are always using the two stage estimation schemes for calculating effort requirement and this observation can be explained through the following illustrations:

Case 1: In the Fig. 1(a), let the shaded area be the effort required for a given LOC. Let LOC = 100, then by assigning the nominal values to all effort multipliers, the required effort in the semi detached mode can be calculated as (a $\times$ size b). i.e $3 \times 1001.12 = 521.3$ Person Months.

Case 2: Considering a projected value for ACAP (Application capability) effort multiplier and the values of all other Effort Multipliers are nominal, then for the same LOC, the required effort can be calculated as $3 \times 1001.12 \times 1.46 = 761.15$ Person months. This is illustrated in the Fig. 1(b).

Case 3: In this case, a projected value is considered for CPLX (Complexity) only and all other Effort Multipliers are kept at nominal level, then for the same LOC, the required effort can be calculated as $3 \times 1001.12 \times 0.7 = 364.93$ Person Months, which is described in the Fig. 1(c).

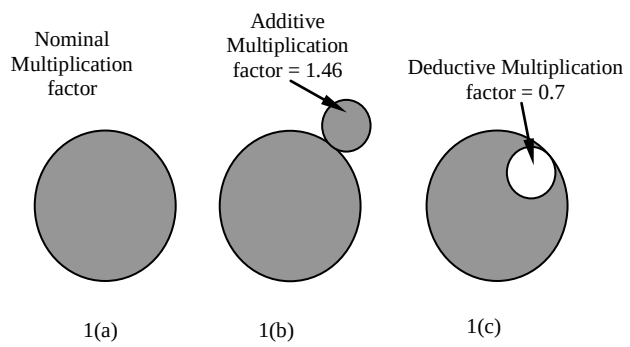


Fig. 1. Two—stage estimation of COCOMO.

Hence, in principle, the effort is calculated by multiplying the estimation variable with the constant 'a' in the first stage, and some effort can be added with or deducted from the calculated effort at the second stage. In addition to that, each user should calibrate the model and the attribute values in accordance according to their own historical projects data, which will reflect local circumstances that greatly influences accuracy of the model. From these perspectives, whenever using the algorithmic effort estimation models, it is preferred

that the impacts of cost drives have to be quantified and assessed in a proper way.

But on the other hand, the cost drives of a software project being developed are characteristically vague and not able quantifiable accurately at the early stage of its life cycle; hence, it is difficult to generate an accurate effort estimate. Thus the vagueness of the cost drives significantly affects the accuracy of the output of effort estimation models. These biases usually rectified when the classification and measurement of the software cost drivers are based on human judgments. Though this cognitive approach has its own uncertainty, but its influence will be comparatively increase the accuracy of the estimation schemes. Since the significance of the vagueness and uncertainty features that are inhabited in the effort drives due to the cognitive judgments is less, this approach can be preferred and applied to change the estimation scheme of the COCOMO II. In this perspective, the work presented in this paper proposed a new approach for handling the cost drivers and scale factors and so as to improve the performance of the COCOMO II effort estimation scheme.

B. Model Formulation Using Graphical Representations

It is observed that the accuracy of the COCOMO II can be viewed as three attributes estimation scheme; the value of estimation variable, the overall value of the effort multipliers and their impact in the estimation scheme. Here the estimation variable is the primary attribute and standard methods available for estimating the estimation variables (e.g. LOC, FP count). It is simple to estimate the overall value of the effort multipliers after assigning the proper values as per the requirements. But the complicated issue is to estimate impact of the effort multipliers, which plays a major role in the estimation scheme and causes for overestimation or underestimation of the software development effort. In this view, this work is aimed at refining the cost drivers and scale factors handling mechanisms in COCOMO II estimation scheme.

There are 17 effort multipliers as cost drivers, whose values are qualitatively defined as *very low, low, nominal, high, very high, extra high*. Based on their impact over the overall effort, these effort multipliers can be devised into two groups; Optimistic Group and Pessimistic Group.

Definition 1: Pessimistic Group: This group can be defined as a set of effort multipliers of whose range values are directly proportional to the overall effort to be predicted and it can be described as follows:

PG= {EM_{10}, EM_{11}, ….., EM_{17}} = {RELY, DATA, CPLX, RUSE, DOCU, TIME, STOR, PVOL}

Definition 2: Optimistic Group (*OG*): This group can be defined as a set of effort multipliers of whose range values are inversely proportional to the overall effort to be predicted and it can be described as follows:

OG = {EM_1, EM_2,….., EM_9} = {ACAP, PCAP, PCON, AEXP, PEXP, LTEX, TOOL, SITE, SCED}

This classification makes a sense in the estimation scheme and plays a vital role in improving the accuracy of the COCOMO II estimation model. The three attributes model can be visualized as a three edged object in a graphical form. In this scheme the overall effort can be estimated in terms of the area(s) of three edged object(s). Let us consider the Fig. 2(a).

In the triangle A, the slope 'ca_1' represents the overall value of the effort multipliers of the Pessimistic Group (*PG*). The length of the line 'ce_1' will be determined accordingly by the value of the estimation variable, here it is LOC measure. By having the lengths of 'ca_1' and 'ce_1', the length of 'δ_1' can be determined automatically, which represents the impact of the effort multipliers over the overall effort to be estimated. The angle between the lines 'ca_1' and 'ce_1' will be determined by the slope of 'ca_1', which will be determined by overall value of the effort multipliers of the Pessimistic Group (*PG*). For a constant LOC value, depending on the values of the effort multipliers of the Pessimistic Group (*PG*), the area under the triangle 'ca_1e_1' will be varied accordingly.

Similarly, in the triangle B, the slope 'cb_1' represents the overall value of the effort multiplier of the Optimistic group (*OG*). The length of the line 'cd_1' will be determined accordingly by the value of the estimation variable, here it is LOC measure. By having the length of 'cb_1' and 'cd_1', the length of 'δ_1' can be determined automatically, which represents the impact of the effort multipliers over the overall effort to be estimated. The angle between the lines 'cb_1' and 'cd_1' will be determined by the slope of 'cb_1', which will be determined by overall value of the effort multipliers of the Optimistic Group (*OG*).

For a constant LOC value, depending on the values of the effort multipliers of the Optimistic Group (*OG*), the area under the triangle 'cb_1d_1' will be varied accordingly.

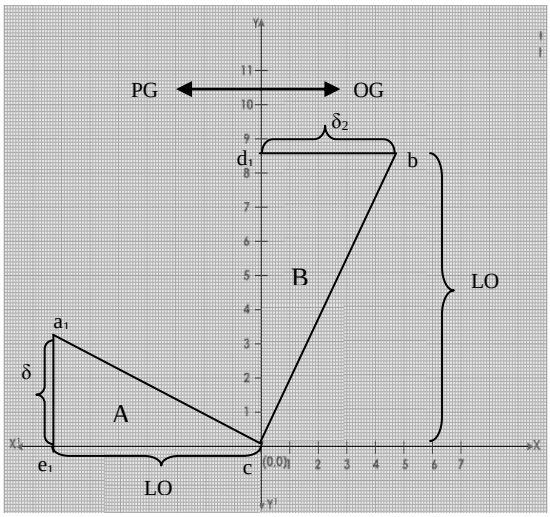
Let LOC be the height of the triangle, δ_1 be the impact of the effort multipliers in the *PG* set and δ_2 be the impact of the effort multipliers in *OG* set. The effort can be calculated as the areas of the triangles for the *PG* and *OG* sets.

The corresponding areas can be calculates respectively as follows:

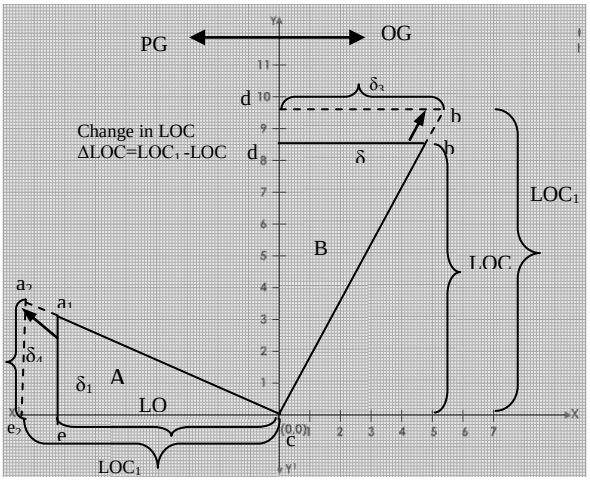
$$^{PG}E_{2a} = 0.5 * \delta_1 * height \quad (3)$$

$$^{OG}E_{2a} = 0.5 * \delta_2 * height \quad (4)$$

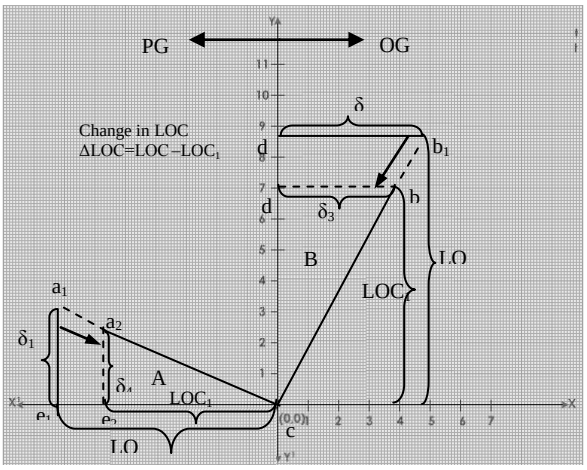
where, $^{PG}E_{2a}$ is the effort due to the Pessimistic Group in the Fig. 2(a) and $^{OG}E_{2a}$ is the effort due to the Optimistic Group in the Fig. 2(a).



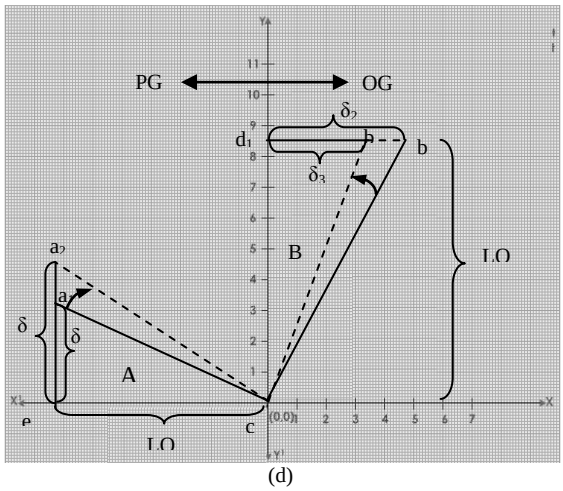
(a)



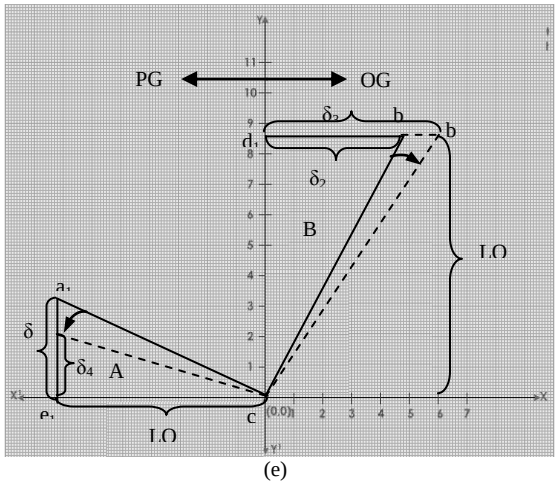
(b)



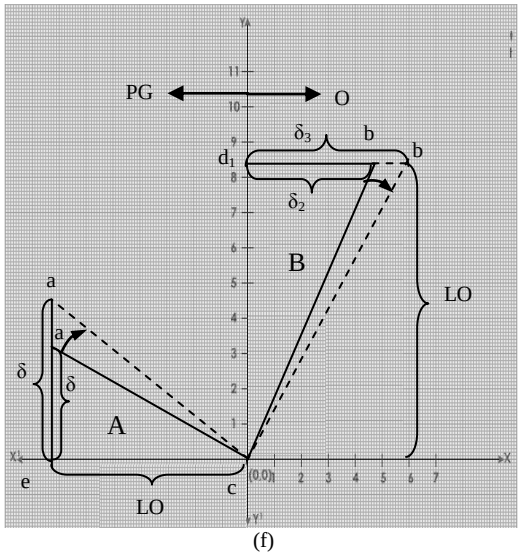
(c)



(d)



(e)



(f)

Fig. 2. Model Formulation using Graphical RepresentationsNow based on the insights of the deductive and additive theories of impact of effort multipliers over the overall effort as discussed in the section 3.1, the values of δ_1 and δ_2 can be assumed accordingly.

By following the deductive principle, the value of δ_1 can be taken as LOC/EM and by following the additive principle, the value of δ_2 can be taken as LOC×EM. This cognitive assumption improves the accuracy of the COCOMO estimation scheme to a larger extent.

IV. Conclusion

Accurate effort estimation is the state of art of the Software Engineering activities and of course it is a complex process. It is well understood that the accuracy of the individual effort estimation models can be defined based on understanding the calibration of the software data and this has been explained by Two – Stage Estimation Schemes in the COCOMO II estimation model in this paper. In this perspective, this paper has described a variant approach for COCOMO II effort estimation model, by redefining the effort multipliers and the scale factors and thereby the overall accuracy of the COCOMO II has been improved. An unique form of graphical representation schemes have been used for enhancing the estimation schemes of COCOMO II. The enhancement has been proved and clearly compared with the traditional COCOMO II in terms of the performance validation factors such as Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE), Root Mean Square (RMS) and Relative Root Mean Square (RMS & RRMS). The observations and analyzes over the obtained results may encourage the researchers to enhance other effort estimation schemes to further levels.

References

[1] R. C. Satyananda, "An improved fuzzy approach for COCOMO's effort estimation using Gaussian membership function," *Journal of Software*, vol. 4, issue 5, pp. 452-459, 2009.

[2] P. Musflek, W. Pedrycz, G. Succi, and M. Reformat, "Software cost estimation with fuzzy models," *Applied Computing Review*, vol. 8, pp. 24-29, 2000.

[3] M. Jorgen and D. I. K. Sjoberg, "The impact of customer expectation on software development effort estimates," *International Journal of Project Management*, vol. 22, issue 4, pp. 317-325, 2004.

[4] N. H. Chiu and S. J. Huang, "The adjusted analogy-based software effort estimation based on similarity distances," *Journal of Systems and Software*, vol. 80, issue 4, pp. 628-640, 2007.

[5] J. Kaczmarek and M. Kucharski, "Size and effort estimation for applications written in Java," *Journal of Information and Software Technology*, vol. 46, issue 9, pp. 589-60, 2004.

[6] R. Jeffery, M. Ruhe, I. Wieczorek, "Using public domain metrics to estimate software development effort," in *Proc. the 7th International Symposium on Software Metrics*, Washington, DC, pp. 16–27, 2001.

[7] A. Heiat, "Comparison of artificial neural network and regression models for estimating software development effort," *Journal of Information and Software Technology*, vol. 44, issue 15, pp. 911-922, 2002.

[8] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort," *IEEE Transactions on Software Engineering*, vol. 21, pp. 126-137, 1995.

[9] A. R. Venkatachalam, "Software cost estimation using artificial neural networks," presented at 1993 International Joint Conference on Neural Networks, Nagoya, Japan, 1993.

[10] G. H. Subramanian, P. C. Pendharkar, and M. Wallace, "An empirical study of the effect of complexity, platform, and program type on software development effort of business applications," *Empirical Software Engineering*, vol. 11, pp. 541-553, 2006.

[11] R. W. Selby and A. A. Porter, "Learning from examples: generation and evaluation of decision trees for software resource analysis," *IEEE Transactions on Software Engineering*, vol. 14, pp. 1743-1757, 1988.

[12] S. J. Huang, C. Y. Lin, and N. H. Chiu, "Fuzzy decision tree approach for embedding risk assessment information into software cost estimation model," *Journal of Information Science and Engineering*, vol. 22, no. 2, pp. 297–313, 2006.

[13] S. Kumar, B. A. Krishna, and P. S. Satsangi, "Fuzzy systems and neural networks in software engineering project management," *Journal of Applied Intelligence*, vol. 4, pp. 31-52, 1994.

[14] M. V. Genuchten and H. Koolen, "On the use of software cost models," *Information and Management*, vol. 21, pp. 37-44, 1991.

[15] T. K. Abdel-Hamid, "Adapting, correcting, and perfecting software estimates: A maintenance metaphor," *IEEE Computer*, vol. 26, pp. 20-29, 1993.

[16] K. Maxwell, L. V. Wassenhove, and S. Dutta, "Performance evaluation of general and company specific models in software development effort estimation," *Management Science*, vol. 45, pp. 787-803, 1999.

[17] V. Nguyen, B. Steece, and B. Boehm, "A constrained regression technique for COCOMO calibration," in *Proc. ESEM'08*, 2008, pp. 213-222.

[18] B. W. Boehm, *Software Engineering Economics*, Prentice Hall, 1981.

[19] B. W. Boehm, E. Horowitz, R. Madachy, D. Reifer, B. K. Clark, B. Steece, A. W. Brown, S. Chulani, and C. Abts, *Software Cost Estimation with COCOMO II*, Prentice Hall, 2000.

[20] F. J. Heemstra, "Software cost estimation," *Information and Software Technology*, vol. 34, pp. 627-639, 1992.

[21] N. Fenton, "Software measurement: A necessary scientific basis," *IEEE Transactions on Software Engineering*, vol. 20, pp. 199-206, 1994.

[22] C. S. Reddy, "Improving the accuracy of effort estimation through fuzzy set representation of size," *Journal of Computer Science*, vol. 5, no. 6, pp. 451-455, 2009.

[23] B. Boehm, *Software Engineering Economics*, Englewood Cliffs, NJ: Prentice-Hall, 1981.

[24] S. D. Conte, H. E. Dunsmore, and V. Y. Shen, *Software Engineering Metrics and Models*, Benjamin-Cummings Publishing Co., Inc., 1986.

Ziyad T. Abdulmehdi is the head of Computer Science & Information Technology Department and also the chair for M.Tech (master computer science) at Mazoon University College, Sultanate of Oman. He holds postdoctoral in database security in 2008, Ph.D. awards in computer science (mobile database systems, in 2007) and master of science degree in computer science (distributed system, in 2003) from University Putra Malaysia (UPM), Selangor, Malaysia. He is currently working in the area of software project management specific effort estimation models. He has published more than 45 research papers in national and international journals and conferences.

M. S. Saleem Basha is working as an assistant professor in the Department of Computer Science, Mazoon University College, Muscat, Sultanate of Oman. He has obtained B.E degree in the field of electrical and electronics engineering, in Bangalore University, Bangalore, India and M.E degree in the field of computer science and engineering, in Anna University, Chennai, India and Ph.D. degree in the field of computer science and engineering in Pondicherry University, India. He is currently working in the area of SDLC specific effort estimation models and web service modeling systems. He has published more than 60 research papers in national and international journals and conferences.

Mohamed Jameel Hashmi is working as the deputy HOD in the Department of Computer Science, Mazoon University College, Muscat, Sultanate of Oman. He has obtained his MCA from Osmania University, Hyderabad, India and he is pursuing Ph.D. in the field of computer science in India. He is currently working in the area of network security specific intrusion detection systems and software engineering. He has published more than 5 research papers in national and international journals and conferences.

P. Dhavachelvan is working as a professor in the Department of Computer Science, Pondicherry University, India. He has obtained his M.E. and Ph.D. degrees in the field of computer science and engineering in Anna University, Chennai, India. He has more than a decade's experience of being an academician and his research areas include software engineering and standards, web service computing and technologies. He has published around 125 research papers in national and international journals and conferences. He is collaborating and coordinating with the research groups working towards to develop the standards for Attributes Specific SDLC Models & Web Services computing and technologies.