Impact of CMMI Based Software Process Maturity on COCOMO II's Effort Estimation

Majed Al Yahya, Rodina Ahmad, and Sai Lee
Department of Software Engineering, University of Malaya, Malaysia

Abstract: *The software capability maturity model has become a popular model for enhancing software development processes with the goal of developing high-quality software within budget and schedule. The software cost estimation model, constructive cost model, in its last update (constructive cost model II) has a set of seventeen cost drivers and a set of five scale factors. Process maturity is one of the five scale factors and its ratings are based on software capability maturity model. This paper examines the effect of process maturity on software development effort by deriving a new set of constructive cost model II's PMAT rating values based on the most recent version of CMM, i.e., capability maturity model integration. The precise data for the analysis was collected from the record of 40 historical projects which spanned the range of capability maturity model integration levels, from level 1 (lower half and upper half) to level 4, where eight data points were collected from each level. We followed the ideal scale factor method in order to withhold the effect of the constructive cost model II's PMAT scale factor. All prediction accuracies were measured using PRED. The study showed that the proposed model (with the new PMAT rating values) yielded better estimates as compared to the generic , constructive cost model II model's estimates.*

Keywords: *Effort estimation, scale factor, cost driver, COCOMO, SW-CMM, CMMI.*

Received November 8, 2008; accepted December 15, 2008

1. Introduction

Software cost estimation is the process of predicting the effort required to develop a software engineering project [24]. This process becomes one of the biggest challenges and the most expensive component in software development. While software cost estimation may be simple in concept, it is difficult and complex in reality [22].

Several estimation models have been developed, and most of them have disappeared without any kind of rigorous evaluation. The reason for this might be that these models were not good and precise enough [25]. In fact, we should not forget that there is another important reason; the people who work in software development prefer to use their own estimation techniques rather than improving and applying the work of the others. According to [17], most companies have relied on experience and ''Price-to-win'' strategies for getting past competitors to win projects. Despite the emergence of concepts like SoftWare Capability Maturity Model (SW-CMM) one can never rely completely on experience based estimation in the software industry because of the rapidly changing technologies, which renders the experience-based estimates ineffective. Furthermore, price-to-win strategy is not very favorable for most of the companies [17]. Hence, the need arises to come up with a more effective cost model to account for the effort spent on developing software systems. A number of algorithmic models have been proposed as the basis for estimating the effort of a software project. They are conceptually similar but use different parameter values. The mathematical model that we discuss here is the COnstructive COst MOdel (COCOMO) [4]. COCOMO model was selected for this study for the following reasons:

- It has a long history from its original version [4] till its most recent version [6].
- It is detailed and well documented in [4] and [6].
- Its datasets are available to the public in the PROMISE repository [33].
- It provides commercial implementations such as Costar [34].

1.1. Problem Overview

Accurate software cost estimation is important for effective project management such as budgeting, project planning and control [21]. The accuracy of software cost estimation has a direct and significant impact on the quality of the firm's software investment decisions [1]. Unfortunately, despite the large body of experience with estimation models (including COCOMO), the accuracy of these models is still far from being satisfactory [23].

Different software cost estimation models have different inputs. The impact of these inputs may vary from one model to another. From the results of studies on the effect of process maturity's on effort, it seems reasonable to suggest that it is an important input to software cost estimation models.

Despite the fact that the Software Engineering Institute at Carnegie Mellon University (CMU-SEI) has released the Capability Maturity Model Integration (CMMI), which is the updated version of the original CMM, COCOMO II still relies on SW-CMM to assess its PMAT scale factor. As far as we are concerned, no new values have been derived to reflect the COCOMO II process maturity under CMMI.

1.2. Research Hypothesis

The hypothesis of the work presented here is that deriving a set of new PMAT values under the CMMI, will improve the prediction power of the COCOMO II model, and make it precisely applicable in software development organizations that are adopting CMMI. The rest of this research is organized as follows. Section 2 presents the definition of the COCOMO model and shows an overview of the CMM-Based process maturity. Section 3 presents some researches that are related to our study. Section 4 describes the data gathering and data analysis methods, while section 5 offers some conclusions of this work and presents recommended future works.

2. Background

2.1. COCOMO II Model

COCOMO was originally published in 1981 (COCOMO 81) [4], and became one of most popular parametric cost estimation models of the 1980s. But in the 90s, COCOMO 81 faced a lot of difficulties and complications in estimating the costs of software that were developed to a new life cycle processes such as non-sequential and rapid development process models, reuse-driven approaches, and object-oriented approaches [5]. Thus, COCOMO II was published initially in the annals of software engineering in 1995 with three sub models; an application-composition model, an early design model and a post-architecture model [5]. COCOMO II has, as an input, a set of seventeen Effort Multipliers (EM) or cost drivers which are used to adjust the nominal effort (PM) to reflect the software product being developed. The seventeen COCOMO II factors (cost drivers) are shown in Table 1 [6].

2.1.1. Effort Estimation

The COCOMO II effort estimation model is formulated as in equation 1. This model is used for both early design and post-architecture models to estimate effort. The inputs are the size of software development, a constant *A*, an exponent *E*, and a number of Effort Multipliers (EM). The number of effort multipliers depends on the model being used.

$$PM = A \times SIZE^{E} \times \prod_{i=1}^{N} EM_i \qquad (1)$$

where the constant A=2.94, and the exponent *E* will be described in the following section.

2.1.2. Scale Factors

A study accomplished by [27] presents the conclusion that the most critical input to the COCOMO II model is size, so, a good size estimate is very important for any good model estimation. Size in COCOMO II is treated as a special cost driver, so it has an exponential factor, E. The exponent *E* in equation 2 is an aggregation of five scale factors. All scale factors have rating levels. These rating levels are Very Low (VL), Low (L), Nominal (N), High (H), Very High (VH) and Extra High (XH). Each rating level has a weight W, which is a quantitative value used in the COCOMO II model. The five COCOMO II scale factors are shown in Table 2 [6]:

$$E = B + 0.01 \times \sum_{j=1}^{N} SF_j \qquad (2)$$

where *B* is a constant = 0.91. *A* and *B* are constant values devised by the COCOMO team by calibrating to the actual effort values for the 161 projects currently in COCOMO II database.

Table 1. COCOMO II Cost drivers.

Cost Driver	Description
RELY	Required Software Reliability
DATA	Data base size
RUSE	Developed for Reusability
DOCU	Documentation needs
CPLX	Product Complexity
TIME	Execution Time Constraints
STOR	Main storage Constraints
PVOL	Platform Volatility
ACAP	Analyst Capability
PCAP	Programmer Capability
APEX	Application Experience
PLEX	Platform Experience
LTEX	Language and Tool Experience
PCON	Personnel Continuity
TOOL	Use of Software Tools
SITE	Multisite Development
SCED	Required Development Schedule

Table 2. COCOMO II Scale factors.

Scale Factor	Description
Precedentedness (PREC)	Reflects the previous experience of the organization.
Development Flexibility (FLEX)	Reflects the degree of flexibility in the development process.
Risk Resolution (RESL)	Reflects the extent of risk analysis carried out.
Team Cohesion (TEAM)	Reflects how well the development team knows each other and work together.
Process Maturity (PMAT)	Reflects the process maturity of the organization.

The procedure for determining PMAT which is the factor of interest in this study- is organized around the SEI-CMM, Table 3 [6].

Table 3. PMAT Scale factor with its rating levels and values.

PMAT Description	CMM Level 1 (lower)	CMM Level 1 (upper)	CMM Level 2	CMM Level 3	CMM Level 4	CMM Level 5
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
Values	7.80	6.24	4.68	3.12	1.56	0.00

According to [14], the CMM level 1 (lower half) is for organizations that rely on "heroes" to do the job. They don't focus on processes or documenting lessons learned. The CMM level 1 (upper half) is for organizations that have implemented most of the requirements that would satisfy CMM level 2. In CMM's published definition, level 1 (lower half) and (Upper half) are grouped into level 1.

2.2. CMM Based Process Maturity

SW-CMM published by SEI is used to rate an organization's process maturity [28]. SW-CMM provides a number of requirements that all organizations can use in setting up the software processes used to control software product development. The SW-CMM specifies "what" should be in the software process rather than "when" or "for how long". There are five levels of process maturity, level 1 (lowest half) to level 5 (highest). To be rated at a particular level, the organization should demonstrates capabilities in a set of Key Process Areas (KPA) associated with a specific SW-CMM level. The capabilities demonstrated in moving from lower levels to higher levels are cumulative. For example, level 3 organizations should show compliance with all KPAs in levels 2 and 3. The SW-CMM process maturity framework is presented in Table 4.

Table 4. CMM Framework.

CMM Level	**Key Process Area**
Level 1	None
Level 2 Repeatable	Requirements Management
	Software Project Planning
	Software Project Tracking and Oversight
	Software Subcontract Management
	Software Quality Assurance
	Software Configuration Management
Level 3 Defined	Organization Process Focus
	Organization Process Definition
	Training Program
	Integrated Software Management
	Software Product Engineering
	Intergroup Coordination
	Peer Reviews
Level 4 Managed	Quantitative Process Management
	Software Quality Management
Level 5 Optimizing	Defect Prevention
	Technology Change Management
	Process Change Management

All organizations are supposed to start at level 1. This is called the Initial level. At this level, few processes are defined, and the success depends on individual effort. This makes the software process unpredictable because it changes as work progresses. Project Schedules, budgets, functionality, and product quality are also unpredictable.

Each KPA has a set of goals, capabilities, key practices, measurements and verification practices. The goals and key practices are the most interesting of these because they could be used to assess the impact of a KPA on a project's development effort. The goals state the scope, boundaries, and intent of a KPA. A key practice describes "what" should happen in that KPA. There are a total of 52 goals and 150 key practices. All of the KPAs are described in [28].

3. Related Work

In this research, we look at the literature from two perspectives. One concentrates on the calibration and improvement of the COCOMO II model, while the other concentrates on the benefits of increasing maturity levels as well as the benefits of CMMI-based software process improvement. Our work is a kind of combination between the previous two perspectives, i.e., we improved the prediction power of the COCOMO II model by investigating the benefits of CMMI based software process maturity.

COCOMO II is being revised, updated, and calibrated to be more suitable for future estimation. There are several calibrations conducted on COCOMO II [9, 13, 15, 31, 32]. Also, numerous studies have been done to enhance the predictive power of the COCOMO model by adding or reducing some influencing factors or cost drivers [2, 11, 20, 21, 25, 29, 30]. On the other hand, much has been discussed on the benefits of increasing maturity levels as well as the benefits of CMMI-based software process improvements [3, 7, 12, 14, 18, 19].

Chulani *et al.* [9] reported a study with a regression tuning algorithm using the COCOMO project database producing estimates that are within 30% of the actual values, 69% of the time, while Clark [15] reported a study with a Bayesian 38 tuning are within 30% of the actual values, 76% of the time after stratification by organization. Hale *et al.* [20] proposed Task Assignment amendments that can aid in the tuning of existing estimation models. They claimed that the prediction ability of COCOMO II would be increased by augmenting it with Task Assignment factor. Miyazaki Y. and Mori in [25] tailored the COCOMO 81 to their own environment. They concluded that the original COCOMO 81 overestimates the effort required to develop software in their Japanese environment, but its tailoring methodology is applicable. Jensen [29] claims that poor management can increase software cost by an immense factor. He reported that by neglecting management approaches, high effort estimates are generated in modern organizations. Yahya *et al.* [30] improved the COCOMO II's predictive power by adding a set of 16 factors to the model and considered it as the most influential factors in their local environment. They claimed that their enhanced model has improved the COCOMO II's predictive power by 9% as compared to the generic COCOMO II. Chen *et al.* [11] concluded that the COCOMO II model can be improved via WRAPPER feature subset selection method developed by the data mining community. Using data sets from the PROMISE repository, they showed WRAPPER significantly and dramatically improves COCOMO II's predictive power. Huang *et al.* [21] have proposed a novel neuro-fuzzy COCOMO for software cost estimation. They claimed that the validation using industry project data shows that the model greatly improves estimation accuracy in comparison with the generic COCOMO model. Based on CMM and by using a 161-project sample, Clark [14] isolated the effects on effort of process maturity versus other effects, concluding that an increase of one process maturity level can reduce development effort by 4% to 11%. While Balk argued in [36] that disaggregation of TOOL variable in COCOMIO II improves the prediction accuracy from 67% to 87%.

4. Research Methodology

The primary data collection tool was a questionnaire that has been used in order to collect a historical data from individual projects, i.e., each questionnaire should be applied only on one project. The questionnaire is based on "COCOMO II cost estimation questionnaire" which was prepared in the center of software engineering at university of southern California, for COCOMO II's annual updating [8].

4.1. Data Collection Procedure

Out of the 75 questionnaires distributed to over 20 software development organizations, 56 questionnaires were returned. Some questionnaires could not be verified with project managers or senior project staff; therefore, 16 questionnaires were rejected and eliminated from this study. Therefore, 40 questionnaires were analyzed. The returned datasets were from various fields such as banking, insurance, communication, simulation, web development, *etc.* The questionnaires were distributed to software organizations that have already achieved one of the CMMI levels, and spanned the range of its levels, from level 1 (lower half) to level 4, i.e., 8 data points were collected from each level.

For each project, there was a meeting with the project manager or team leader who would be filling out the forms, in order to clarify each question to ensure that it was understood well and each manager would answer consistently.

4.2. Data Analysis

Once the questionnaires were returned, they were checked for consistency and went through a data validation process, based on some constraints determined in [6]. In fact, for each questionnaire, there are four aspects that would be extracted and computed:

1. A set of seventeen COCOMO II's cost drivers. To deal with these seventeen cost drivers, we computed their multiplication. A sample of the cost drivers is shown in Table 5.
2. A set of five exponential scale factors. To deal with these five scale factors, we computed their summation. A sample of these scale factors is shown in Table 6 (excluding the last row).
3. Actual effort in Person Months (PM), which extracted for the person hours, as shown in Table 7.
4. The size of the project. We collected the project size as a thousand lines of code (KLOC), which is the baseline size in COCOMO II.

To predict the effort in PM, we applied equation 1 which is the basic COCOMO II's formula [6]. At the end of this analysis, we got the estimated effort for the generic COCOMO II as well as the actual effort for this project. To derive the new PMAT values, we computed ISF as shown in the next section.

4.3. Ideal Scale Factor Analysis on PMAT

Boehm [4] has described a method to normalize out contaminating effects of individual cost driver attributes in order to have a clearer picture of that cost driver's contribution. Since we have relatively similar situation, i.e., we need to normalize out contaminating effects of a scale factor (in our case, PMAT) rather than a cost driver. Therefore, in our context, we defined that:

For the given project P, compute the estimated development effort using the COCOMO II estimation procedure, with one exception: do not include the value for the Scale Factor Attribute (SFA) being analyzed. Call this estimate PM (P, SFA). Then the ideal scale factor, ISF (P, SFA), for this project/scale-factor combination is defined as the value which, if used in COCOMO II, would make the estimated development effort for the project equal to its actual development effort PM (P, Actual). i.e.,

$$ISF\ (P,\ PMAT) = PM\ (P,\ Actual)\ /\ PM\ (P,\ PMAT) \qquad (3)$$

where

- ISF (P, PMAT): the ideal scale factor on PMAT for project P.
- PM (P, Actual): the actual development effort for the project P.

- PM (P, PMAT): COCOMO II estimate excluding the PMAT scale factor.
- PM: person-months.

4.3.1. Steps for ISF-PMAT Analysis

We performed the following steps to complete the ISF-PMAT analysis on our datasets:

1. Compute the *PM* (P, SFA), using the following formulas:

$$PM= A \times SIZE^{E} \times \prod_{i=1}^{17} EM_i \tag{4}$$

   where A is a model constant, EM is a set of seventeen effort multipliers as shown in Table 1, and

$$E= B + 0.01\times \sum_{j=1}^{4} SF\_But\_PMAT_j \tag{5}$$

   where B is a model constant, and *SF_But_PMAT* refers to scale factors except PMAT, including PREC, FLEX, RESL, and TEAM.
2. Compute the ISF (P, SFA) using equation 6.
3. Group ISF (P, SFA) by the current CMM PMAT rating (i.e., VL, L, N, H, VH).
4. Compute the mean value for each group as ISF-*PMAT* value for that rating.

This step involves the computation of the mean value of ISF-PMAT for each CMM rating level.

4.4. Evaluation of the Prediction Accuracy

The focus of this paper is on the degree to which the model's estimated effort measured in Person-Month (PMes) matches the actual effort (PMact). If the model is perfect (this is rare) then for any project, PMes = PMact. A common criterion for the evaluation of cost estimation models is the Relative Error (RE) or the Magnitude of Relative Error (MRE), which are defined as:

$$RE= (PMes - PMact)/PMact \tag{6}$$

$$MRE= \left|(PMes - PMact)\right| / PMact \tag{7}$$

The RE and MRE values are calculated for each project whose effort is predicted. Another criterion that is commonly used is the percentage of predictions that fall within *P* % of the actual, denoted as *PRED (P)* [16],

$$PRED\ (P) = K / N \tag{8}$$

K is the number of projects where MRE is less than or equal to *P*, and *N* is the number of projects. According to [10], a standard method for assessing COCOMO performance is PRED (30). Therefore we used this criterion to assess the COCOMO II performance as compared to the proposed model (with new PMAT values). Table 5 through Table 10 show samples of the calculated data, which represent one project from our forty datasets.

Table 5. COCOMO II Cost drivers with their effort multipliers.

Cost Driver	Value
RELY	1.1
DATA	1
RUSE	1
DOCU	1.23
TIME	1.29
STOR	1.05
PVOL	0.87
ACAP	0.71
PCAP	0.88
PCON	0.9
APEX	0.81
PLEX	0.85
LTEX	0.84
TOOL	0.78
SITE	0.86
SCED	1
CPLX	1.34

Table 6. COCOMO II Scale factors and their values.

Scale Factor	Value
PREC	3.72
FLEX	1.01
RESL	2.83
TEAM	2.19
PMAT	1.59
New PMAT	1.03

Table 7. The actual time, effort, size, estimated time, and the cost drivers multiplication.

Description	Value
Actual Time	١٦,٥
Actual Effort	143.62
Size (KSLOC) =	١١٠
Estimated Time, T =	١٧,٣
Π Cost Drivers, EM =	0.466

Table 8. Estimated effort in generic COCOMO II.

Description	Value
∑ Scale Factors, SF =	11.310
Estimated Effort, PM =	168.02
Magnitude Relative Error=	0.17

Table 9. ISF and Estimated effort without PMAT value.

Description	Value
∑ Scale Factors-BUT-PMAT	9.75
Estimated Effort, BUT-PMAT	156.14
Ideal Scale Factor, ISF	0.92

Table 10. Estimated effort with new PMAT values.

Description	Value
∑ scale factors with ISF-PMAT	10.78
Estimated Effort with ISF-PMAT	163.87
Magnitude Relative Error=	0.14

After applying our methodology to the forty datasets, a new set of PMAT rating values under CMMI has been derived as in Table 11.

Table 11. The new PMAT rating values.

PMAT Description	CMMI Level 1 (lower)	CMMI Level 1 (upper)	CMMI Level 2	CMMI Level 3	CMMI Level 4	CMMI Level 5
Rating Levels	Very Low	Low	Nominal	High	Very High	Extra High
New PMAT Values	٧,٥٥	٥,٧١	٣,٨١	٢,٠٨	١,٠٣	0.00

In Figure 1, *X* axis represents the 8 projects used in CMMI level 4 organization in our study. *Y* axis represents the effort. Each project (in *X* axis) has three columns: the left one (black column) represents the actual effort, the middle one (white gray column) represents the generic COCOMO II effort estimation, and the right one (dark gray column) represents the effort estimation for the proposed COCOMO II model with new ISF-PMAT values. The figure demonstrates how the proposed model (with ISF-PMAT) has succeeded to give an estimated effort which is closer to the actual effort than generic COCOMO II estimations. This case is not absolute, i.e., in some little cases like in CMMI level 1 (lower and upper) and level 2 datasets, the estimated efforts by the generic COCOMO II were relatively closer to the actual effort than the proposed model's estimation. The reason being due to some data anomalies, especially for low levels companies that do not have good and precise documentations for their historical projects.

As shown in Figure 1, there are slight consistent overestimations for most of the projects. According to [26], an effort estimation model can still be consistent if it uniformly misestimates (i.e., underestimates or overestimates) effort for a set of projects. Since the proposed model presented here is uniformly overestimated the effort for most of the 8 projects, so it could still be a consistent model.

The black dotted line in Figure 2 shows the current PMAT scale factor values used in COCOMO II. It shows that an increase in process maturity level corresponds with a reduction in project effort. The gray line shows the new PMAT values derived from the ISF-PMAT analysis using our forty datasets. The VERY LOW ratings for PMAT decreased slightly from 7.80 to 7.55, while the LOW ratings decreased from 6.24 to 5.71.

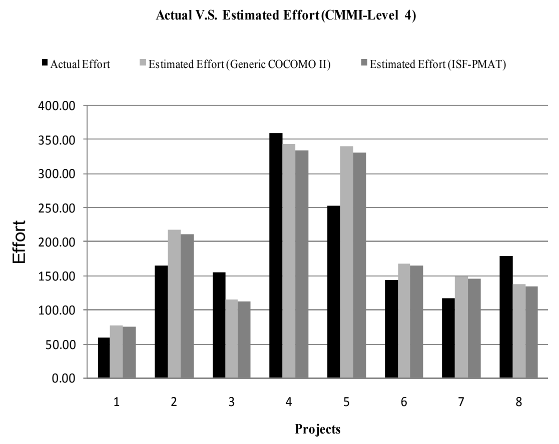

Figure 1. Actual and estimated effort in both generic COCOMO II and COCOMO II with ISF-PMAT.

Since VERY LOW and low rating levels in COCOMO II's PMAT are categorized under CMMI level 1, i.e., few number of Process Areas (PA) are assigned to this level, and success still depends on individual effort. Therefore, level 1 companies still need much effort to accomplish their projects, particularly for CMMI level 1 (lower half) companies that rely on "heroes" to do the jobs and do not show any compliance that would satisfy subsequent levels.

Another observation is that NOMINAL and HIGH rating levels (CMMI levels 2 and 3) demonstrated a relatively obvious reduction in PMAT values, which appears as a deviation in the gray line in Figure 2. Our underlying explanation behind this reduction might be due to the major additions and refinements that have occurred at CMMI maturity levels 2 and 3. As an example, going from seven key process areas in SW-CMM level 3, to 14 process areas in CMMI level 3 (included additional goals and practices), and just two PAs were dropped. These additions and refinements in maturity levels 2 and 3 reflect their significance and definitely will reduce the effort required to develop the software systems in CMMI maturity levels 2 and 3 organizations.

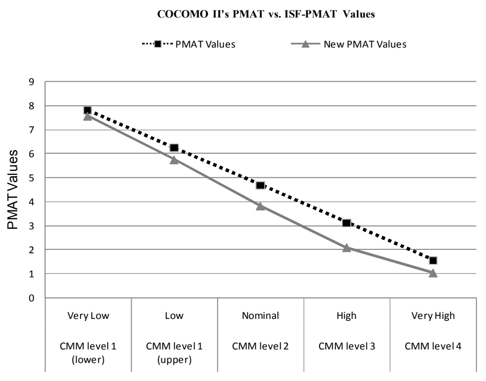

Figure 2. COCOMO II's PMAT values vs. the ISF-PMAT values.

4.5. Model Accuracy with ISF Results

After applying the derived ISF-PMAT values back to our forty datasets, improvement in the model's accuracy has been realized. This improvement is shown in Table 12.

Table 12. Accuracy analysis results of our study.

CMMI Level	PRED (30)		Improvement
	Generic COCOMO II	COCOMO II with New PMAT Values	
Level 1 (Lower)	63%	75%	12%
Level 1 (Upper)	٥٠%	٦٣%	13%
Level 2	38%	75%	37%
Level 3	38%	88%	50%
Level 4	50%	٨٨%	25%

Table 12 shows that by applying the ISF-PMAT values into our forty datasets that had been collected from CMMI organizations, the accuracy level - PRED (30) - in all maturity levels increased by 12%, 13%, 37%, 50%, and 25% respectively. As we mentioned and justified earlier, Table 12 shows that level 3 has the highest percentage of improvement, and the lowest percentage of improvement assigned to level 1 with its extensions, lower and upper halves.

5. Conclusions and Future Work

Accurate software development cost estimation is very important in the budgeting, project planning and effective control of project management. Different software cost estimation models have different inputs. One of the most important inputs to software cost estimation models (including COCOMO) is process maturity, PMAT. According to our study, it shows that the current values for the COCOMO II PMAT scale factor do not adequately reflect the impact of CMMI-based process maturity on development effort. Therefore, by using the ISF method and with the aid of our forty datasets, we have derived new PMAT values that better reflect the impact of CMMI based process maturity on software development effort. The new values resulted an improvement in COCOMO II model accuracies in terms of PRED (30) by 12% for CMMI level one (lower half), 13% for CMMI level one (upper half), 37% for CMMI level two, 50% for CMMI level three, and 25% for CMMI level four organizations.

A number of opportunities exist for future work in the area of CMMI-based process maturity using COCOMO II. Firstly, the amount of datasets allocated to each CMMI maturity level could be expanded to get a clearer picture of the impact of CMMI-based process maturity on software development effort. Secondly, locally calibrating the proposed model parameters to a particular organization, this requires collecting data from more than 10 projects belonging to the same organization. Finally, unlike SW-CMM, CMMI has two different representations; staged and continuous. Most IT organizations are adopting the Staged representation which is structurally different from the Continuous one. This study focused on the organizations that are adopting staged representation. Therefore, we recommend collecting data from CMMI organizations that are adopting CMMI's continuous representation in order to derive new PMAT rating values from continuous representation perspective.

References

[1] Al-Sakran H., "Software Cost Estimation Model Based on Integration of Multi Agent and Case Based Reasoning," *Journal of Computer Science*, vol. 2, no. 3, pp. 276-282, 2006.

[2] Baik J., "Disaggregating and Calibrating the Case Tool Variable in COCOMO II," *Computer Journal of IEEE Transactions on Software Engineering*, vol. 28, no. 6, pp. 1009-1022, 2002.

[3] Brodman J. and Johnson D., *Return on Investment from Software Process Improvement as Measured by US Industry*, John Wiley and Sons Ltd, 1995.

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

[5] Boehm B., Clark B., Horowitz E., Westland C., Madachy R., and Selby R., "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0," *in Proceedings of Special Volume on Software Process and Product Measurement*, Amsterdam, pp. 45-60, 1995.

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

[7] Butler K., "The Economic Benefits of Software Process Improvement," *in Proceedings of Crosstalk Hill AFB*, Ogden, pp. 14-17, 1995.

[8] Center for Software Engineering, *COCOMO II Cost Estimation Questionnaire*, Prentice Hall, California, 2000.

[9] Chulani S., Boehm B., and Steece B., "Bayesian Analysis of Empirical Software Engineering Cost Models," *Computer Journal of IEEE Transactions on Software Engineering*, vol. 25, no. 4, pp. 573-583, 1999.

[10] Chen Z., Menzies T., Port D., and Boehm B., "Finding the Right Data for Software Cost Modeling," *Computer Journal of IEEE Software Special Issue*, vol. 22, no. 6, pp. 38-46, 2005.

[11] Chen Z., Menzies T., and Port D., "Feature Subset Selection Can Improve Software Cost Estimation Accuracy," *in Proceedings of Workshop Predictor Models in Software Engineering*, California, pp. 245-248, 2005.

[12] Chrissis M., Konrad M., and Shrum S., *CMMI: Guidelines for Process Integration and Product Improvement*, Addison Wesley, 2003.

[13] Chulani S., Boehm B., and Clark B., "Calibrating the COCOMO II Post Architecture Model," *in Proceeding of ICSE98 IEEE*, Malaysia, pp. 477-480, 1998.

[14] Clark B., "Quantifying the Effects of Process Improvement on Effort," *Computer Journal of IEEE Software*, vol. 17, no. 6, pp. 65-70, 2000.

[15] Clark B., "Calibration of COCOMO II.2003," http://sunset.usc.edu/events/2002/cocomo17/Calibration%20fo% 20COCOMO%20I I.2003 %20 Presentation %20-%20Clark.pdf, Last Visited 2010.

[16] Conte S., Dunsmore H., and Shen V., *Software Engineering Metrics and Models*, Menlo Park, CA, Benjamin/Cummings, 1986.

[17] Dillibabu R. and Krishnaiah K., "Cost Estimation of a Software Product Using COCOMO II.2000 Model a Case Study," *International Journal of Project Management*, vol. 23, no. 2, pp. 297-307, 2005.

[18] Gibson D., Goldenson D., and Kost K., "Performance Results of CMMI Based Process Improvement," *CMU/SEI-94-TR-13 Technical Report,* 2006.

[19] Goldenson D. and Gibson D., "Demonstrating the Impact and Benefits of CMMI: An Update and Preliminary Results," *Technical Document (CMU/SEI-2003-SR-009)*, Carnegie Mellon University, 2003.

[20] Hale J., Parrish A., Dixon B., and Smith R., "Enhancing the COCOMO Estimation Models," *Computer Journal of IEEE Software*, vol. 17, no. 2, pp. 45-50, 2000.

[21] Huang X., Ho D., Ren J., and Capretz L., "Improving the COCOMO Model with a Neuro Fuzzy Approach," *Computer Journal of Applied Soft Computing Journal*, vol. 7, no. 3, pp. 29-40, 2007.

[22] Jones C., "Software Cost Estimation in 2002," *Computer Journal of Defense Software Engineering*, vol. 15, no. 6, pp. 4-8, 2002.

[23] Leung H. and Zhang F., *In Handbook of Software Engineering and Knowledge Engineering,* World Scientific, 2002.

[24] Lindstrom B. "A Software Measurement Case Study Using GQM," *Master Thesis*, Lund University, 2004.

[25] Miyazaki Y. and Mori K., "COCOMO Evaluation and Tailoring," *in Proceedings of ICSE 8, IEEE-ACM-BCS*, pp. 292-299, 1985.

[26] Mukhopadhyay M. and Kekre S., "Software Effort Models for Early Estimation of Process Control Applications," *Computer Journal of IEEE Transactions on Software Engineering*, vol. 18, no. 10, pp. 226-228, 1992.

[27] Musilek P., Pedrycz W., Sun N., and Succi G., "On the Sensitivity of COCOMO II Software Cost Estimation Model," *in Proceedings of the 8th IEEE Symposium on Software Metrics, METRICS IEEE Computer Society*, Washington, pp. 13, 2002.

[28] Paulk M., Weber C., Curtis B., and Chrissis M., *The Capability Maturity Model: Guidelines for Improving the Software Process,* Addison-Wesley, 1995.

[29] Randall J., "Extreme Software Cost Estimating," *Computer Journal of Defense Software Engineering*, vol. 17, no. 1, pp. 27-30, 2004.

[30] Yahya M., Masoud F., and Hudaib A., "The Effect of Software Development Environment on Software Cost Estimation," *in Proceedings of the 10th World Multi Conference on Systematic, Cybernetics and Informatics*, pp. 239-243, USA, 2006.

[31] Yang Y. and Clark B., "COCOMO II.2003 Calibration Status," http://sunset. usc.edu /events /2003/ March_ 2003 /COCOMO_ II_2003_ Recalibration. Pdf, Last Visited 2010.

[32] Yang Y. and Clark B., "Reducing Local Calibration Bias in COCOMO II Calibration," *in Proceedings of 19th International Forum on COCOMO and Software Cost Modeling*, pp. 26-29, 2004.

[33] Zelkowitz M., http://promise .site. uottawa. ca/ SER epository /datasetss-page.html, 2005.

[34] Zedvidović N., Boehm B., and Settles S., http://sunset.usc.edu /COCOMOII/ suite.html, Last Visited 2010.

Majed Al Yahya received his Bachelor degree in computer science from Zarqa Private University, Jordan in 2002, and Master degree in computer science from University of Jordan in 2004. His current research interests include software process improvement, capability maturity model integration, and software cost estimation.

Rodina Ahmad is a senior lecturer in software engineering and information systems at the Faculty of Computer Science and Information Technology, University of Malaya. She teaches information systems and software engineering modules at both the undergraduate and master levels. She holds a degree in computer science and mathematics from Hartford, Conn. Her Master degree was from Rensselaer Polytechnic Institute, USA. Her PhD degree in information systems is from National University of Malaysia.

Sai Lee received her Master of computer science from University of Malaya in 1990, her Diplôme d'Études Approfondies in computer science from University of Pierre et Marie Curie (Paris VI) in 1991 and her PhD degree in computer science from University of Panthéon-Sorbonne (Paris I) in 1994. Her current research interests include software reuse, application and persistence frameworks, requirements and design engineering, object oriented techniques and CASE tools. She has published more than 80 research papers in local and international journals and conferences.