

Empirical Study on Enhancing the Accuracy of Software Cost Estimation Model for Defense Software Development Project Applications

Taeho Lee*, Donoh Choi*, Jongmoon Baik**

* *Analysis & Evaluation Division, Agency for Defense Development(ADD)*

111 Sunam-Dong, Yuseong-Ku, Daejeon, South Korea

***Dept. of Computer Science Division, Korea Advanced Institute of Science and Technology(KAIST)*

Guseong-Dong, Yuseong-Ku, Daejeon, South Korea

lth2094@add.re.kr, donoh@add.re.kr, jbaik@kaist.ac.kr

Abstract - This paper provides the result of the empirical study on enhancing the accuracy of software cost estimation model for defense software development project applications. It is very important to build a software cost model adequate to the environment for the accurate results of the estimation. Calibration can be used for building its own software cost model by using pre-existing model such as the COCOMO II. In this paper, we practiced the calibration of the COCOMO II model using data collected from the defense software development projects by using calico calibration software. The results show that the estimation accuracy became enhanced after calibration. Also we found that the performance of calibrating both the multipliers and exponent is better than calibrating multipliers only. Finally this paper includes comments on reasons of the lower calibration performance and future works to enhance the estimation accuracy.

Keywords - COCOMO II, Cost Estimation, Calibration, Prediction Accuracy

I. INTRODUCTION

Nowadays as the trend of military weapon systems is seeking for the system of systems and requiring high inter-operability among the system, the development cost - which is comprised of hardware development cost and software development cost is steeply increasing. Especially in the area of defense and aerospace, as the size of a software project is becoming more complex and bigger, the portion of embedded software cost is continuously increasing and up to over 50% in 2006.

As the size of software project is becoming more complex and bigger, estimating software development cost is indispensable when we plan to initiate a new project. It has an important role in the process of executing a software development project by acting as a key constraint to the project. It can affect the success or failure of the software project in accordance with the quality of the estimation result. Also, software cost estimates are critical to all stakeholders to the project such as project managers, developers, customers and so

on. The outcomes of software cost estimation, such as effort, project duration and cost, are valuable information to the stakeholders for initiating and executing software project. They can be used for generating request for proposals, contract negotiations, scheduling, monitoring and control. They are used as key inputs for project bidding & proposal, budget & staff allocation, project planning & control, investment decision, tradeoff & risk analysis [1]. Accurate software cost estimation is very important for the following reasons [2]:

- It can help to classify and prioritize development projects with respect to an overall business plan.
- It can be used to determine what resources to commit to the project and how well these resources will be used.
- It can be used to assess the impact of changes and support re-planning.
- Projects can be easier to manage and control when resources are better matched to real needs.
- Customers expect actual development costs to be in line with estimated costs

There are various software cost estimation techniques. They can be classified as model-based, expert-based, learning-oriented, dynamics-oriented, regression-based and composite. The methods applying each of the techniques are as follows [3]:

- Model-based: Jensen, Bailey-Basili's, Gulezian's, Softcost, ESTIMACS, SEER-SEM, PRICE-S, COCOMO, SLIM
- Expert-based: Delphi, Rule-based systems, WBS
- Learning-oriented: Neural networks, Case-Based Reasoning
- Dynamics-oriented: System Dynamics Approach, Detailed-COCOMO, PRICE-S
- Regression-based: OLS, Robust regression
- Composite: Bayesian approach

The performance or accuracy of these models is not yet clearly determined in the defense software project area in Korea. The use of these models is determined by the characteristics or

the environments of the developing software projects. However, software cost estimation methods that are mainly used for Korea defense software development projects is limited to two or three methods. The first one is the Standard for the Software Project Pricing developed by the Ministry of Knowledge Economy(MKE). If we want to propose a software development project, we should calculate the software development project cost by mandatorily using this method. The second one is PRICE-S which is the commercial software cost estimation model provided by PRICE systems. Other methods are the Standard for the Engineering Project Pricing by the Ministry of Education and Science and Technology(MEST) and SEER-SEM provided by Galorath. SEER-SEM is recently introduced and is beginning to be used in Korea defense domain.

The COCOMO model is introduced to most of the related software engineers, but it was not used at all in practice even though it is widely known to the software industry. The reasons why the COCOMO model is not frequently used are, the lack of effort to propagate the model in our special situation of politics and business, the lack of various research on applying it to our situation, the lack of customizing effort, etc. Among the reasons as mentioned above, the lack of various researches on applying it to our situation is more dominant.

It is very important to build the software development cost model adequate to the development environment for the accurate results of the estimation. Calibration can be used for building the software cost model by using a pre-existing model such as COCOMO. However, it is difficult to execute the calibration activity if we have a small number of project data with which to calibrate or if some of the project characteristics vary significantly from one project to another. Also, the calibrated model may focus on the normal or typical circumstances. This will lead us to situations in which extreme cases are difficult to predict. Thus, there are two sides to calibration: advantages or disadvantages [4]. The Calibration results in a model tailored very specifically to an organization. Thus, a calibrated model usually provides more accurate estimation results than general, all-purpose models do. And calibrated models can be modified to include input variables and considerations that may affect the estimation results. But, it has a weak point. In case of new or radically different projects, they may have no historical database or baseline project from which to draw the calibrated line. An improperly calibrated model can easily lead to an incorrect estimate.

Various researches on software cost estimation models including the COCOMO model has been carried out [5], but calibration effort has not yet been performed for practical use, especially in the defense project area. There are some calibrating the COCOMO related researches, such as Empirical practice on calibrating COCOMO model using US historical data[6], techniques for the COCOMO calibration[7][8][9], Validation method [10], etc. But, in case of Korea, the calibration effort in applying the COCOMO model has not been

performed yet, especially in the area of defense software development which is the major area of applying software cost estimation.

The primary objective of this research is to enhance the accuracy of the software development cost estimating model, COCOMO-II model, by calibrating the COCOMO-II model using historical data on defense software development projects and assess the usability of these calibration results in the situation of future defense software projects in Korea.

II. BACKGROUND

A. COCOMO- II Model

The Constructive Cost Model(COCOMO) is an algorithmic Software Cost Estimation Model developed by Barry Boehm [9]. The model uses a basic regression formula, with parameters that are derived from historical project data and current project characteristics. It drew on a study of 63. The study examined projects ranging in size from 2,000 to 100,000 lines of code, and programming languages ranging from assembly to PL/I.

COCOMO-II is the latest major extension to the original COCOMO(COCOMO 81) model. It consists of three submodels, each one offering increased fidelity the further along one is in the project planning and design process. Listed in increasing fidelity, these sub-models are called the Applications Composition, Early Design, and Post-architecture(PA) models. The COCOMO-II PA model predicts software development effort - Person Months(PM) and project duration in months. It uses source lines of code or function points for sizing, a set of 17 Effort Multipliers(EMs) – shown in Table 1 – and a set of 5 scaling cost drivers to determine the project's Scaling Factors (SFs) – shown in Table 2 –.

Table 1. Cost driver for COCOMO-II PA model

Attribute	Type	Description
RELY	Product	Required system reliability
CPLX	Product	Complexity of system modules
DOCU	Product	Extent of documentation required
DATA	Product	Size of database used
RUSE	Product	Required percentage of reusable components
TIME	Computer	Execution time constraint
PVOL	Computer	Volatility of development platform
STOR	Computer	Memory constraints
ACAP	Personnel	Capability of project analysts
PCON	Personnel	Personnel continuity
PCAP	Personnel	Programmer capability
PEXP	Personnel	Programmer experience in project domain
AEXP	Personnel	Analyst experience in project domain
LTEX	Personnel	Language and tool experience
TOOL	Project	Use of software tools
SCED	Project	Development schedule compression
SITE	Project	Extent of multisite working and quality of inter-site communications

Table 2. Scale factor for COCOMO II PA model

Scale Factor	Explanation
Precedented -ness	Reflects the previous experience of the organization with this type of project. Very low means no previous experience; Extra high means that the organization is completely familiar with this application domain.
Development	Reflects the degree of flexibility in the development process. Very low means a prescribed process is used; Extra high means that the client sets only general goals.
Team cohesion	Reflects how well the development team knows each other and work together. Very low means very difficult interactions; Extra high means an integrated and effective team with no communication problems.
Process maturity	Reflects the process maturity of the organization. The computation of this value depends on the CMM Maturity Questionnaire, but an estimate can be achieved by subtracting the CMM process maturity level from 5.

In COCOMO-II PA model, estimating effort and schedule can be calculated from equation (1) and equation (2).

$$\begin{aligned}
 & \bullet PM_{NS} = A * Size^E * \prod_{i=1}^{17} EM_i \quad (1) \\
 & \quad \text{Where } E = B + 0.01 * \sum_{j=1}^5 SF_j, \quad A = 2.94, B = 0.91 \\
 & \bullet TDEV = C * (PM)^F \quad (2) \\
 & \quad \text{Where } F = D + 0.2 * 0.01 * \sum_{j=1}^5 SF_j = D + 0.2 * (E - B) \\
 & \quad C = 3.67, D = 0.28
 \end{aligned}$$

B. Performance Measures

There are various measures for assessing the accuracy of the models: Mean Magnitude of Relative Error(MMRE), Prediction Level(PRED), Root Mean Square(RMS), and so on. MMRE are calculated based on a number of actual data observed and estimates generated by the model. It is derived from the basic magnitude of the relative error MRE, which is defined as:

- *Relative Error (RE)*

$$= (Estimated\ Effort - Actual\ Effort) / Actual\ Effort$$
- *Magnitude of Relative Error (MRE)*

$$= |Estimated\ Effort - Actual\ Effort| / Actual\ Effort$$
- *Mean Magnitude of Relative Error (MMRE)*

$$= (\sum MRE) / n$$

In calculating MMRE, all of the MRE estimates are included. Therefore extreme values of MRE can affect MMRE significantly. To overcome this problem, another important criterion used for model evaluation is PRED. PRED(L) is defined as the percentage of estimates where MRE is not greater

than L, where k is the number of estimates with MRE falling within $MRE \leq L$ and n is the total number of estimates. We can see that unlike MMRE, PRED(L) is insensitive to errors greater than L [1].

Nowadays MMRE and PRED are the most widely used metrics for evaluation of the accuracy of cost estimation models[12]. Therefore, in this research, we report the results primarily using PRED(0.3) and additionally report MMRE also. This PRED(0.3) value is considered as a standard in reporting COCOMO calibration and model improvement in previous studies [1].

III. MODEL CALIBRATION

We practiced the calibration of COCOMO-II PA model according to the following steps: Collect historical data, make assumptions, determine the calibration method, model and tools, execute the calibration, analyze the calibration result, assess and evaluate the result. Followings are description of the calibration process and contents.

A. Data Collection

The data for calibration is collected by the Agency for Defense Development(ADD) which is the organization develop military weapon systems and research related core technologies for Korea government. The research and development sector of the weapon systems and technical area by ADD is diverse: the grounding system, naval system, missile system, electronics, telecommunication, aero or fluid dynamics and so on.

We investigated 400 software which were developed by ADD researchers or outsourced to make the Cost Estimating Relationships(CER) of the development cost, person month and duration . Data was recorded on a data collection form by e-mail that asked 8 questions: software application area, operation environment, language, source lines of code, development cost, actual person months, duration, and category of the program.

The data was collected from the completed projects since 1998. The total replied observations were 400, but the final data used in the calibration was 86. Rest of the data was not used due to the smaller SLOC size below one thousand and smaller person month below one or consolidated to one due to the similarity of data set.

Figure 1 shows the application area of the software. 48(56%) of 86 software are for the use of Science and Technology, such as analysis, simulation, and so on.

Figure 2 shows the programming language of the software. 60(70%) software are programmed by using C/C++ or visual C++ programming language.

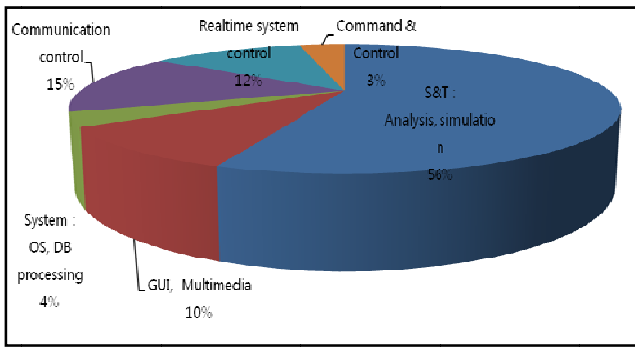


Figure 1. Application area of the software

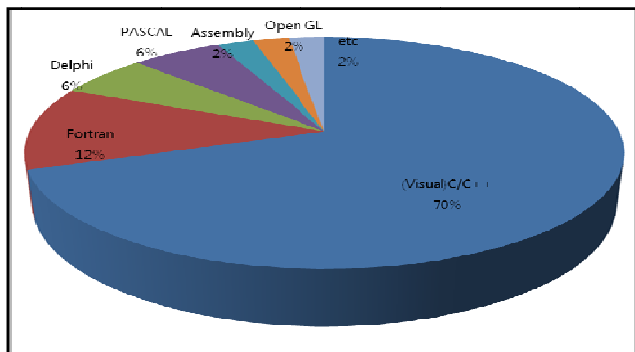


Figure 2. Programming Language of the software

The data collected included the actual effort and schedule spent on the projects. Effort is in units of Person Months. A person month is 160 hours a month and includes development and management hours. Person month is 152 hours a month in America, but person month is 160 hours a month in Korea. Therefore we should adjust this point when we used effort data. Schedule is calendar months and adjusted KSLOC is the thousands of lines of source code count.

The following two histograms show the frequency of responses for this data. The 86 historical data set consists of in house development projects, excluding outsourcing project. Histogram shows that most of the software have relatively small SLOC size: minimum 5,000 ~ maximum 300,000. Also Person Month is relatively small: minimum 1.2 ~ maximum 72.

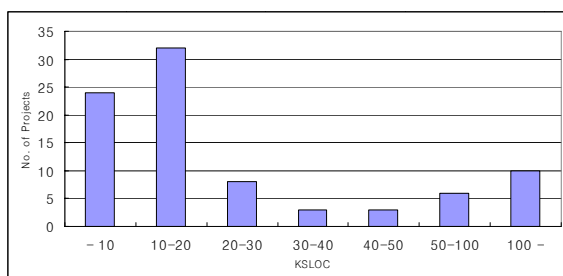


Figure 3. Data distribution per size

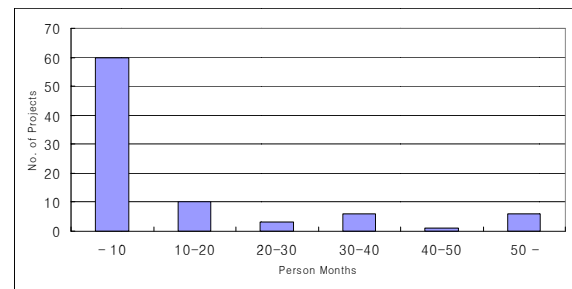


Figure 4. Data distribution per person month

B. Assumptions

As we mentioned above, data collection is intended to make CER, not calibrate COCOMO model. Therefore we cannot determine the values of cost drivers and scale factors. Inevitably we should assume the values of cost drivers and scale factors.

Fortunately, we can induce the value of RELY cost driver by analyzing the usage and application area of each software. And the homogeneous feature of defense R&D projects and the same organization, we can assume that cost drivers such as product, platform, personnel and project environment have similar value. Also is cased for scale factors. For these reasons, we finally assumed that all cost drivers except RELY and all scale factors have normal values.

C. Calibration Method

Once we collect right data, we can go to the next step: executing calibration on the person months and duration. To calibrate we should linearize equation (1) and equation (2) by using a log-transformed model. The generalized form of log-transformed model for calibrating Person Months is shown equation (3).

$$\begin{aligned} \ln(PM) = & a_0 + a_1 * 1.01 * \ln(size) + a_2 * SF_1 * 1.01 * \ln(size) + \dots \\ & + a_6 * SF_5 * \ln(size) + a_7 * \ln(EM_1) + \dots \\ & + a_{23} * \ln(EM_{17}) \end{aligned} \quad (3)$$

Calibration can be executed by using regression method, Ordinary Least Square(OLS). Also COCOMO II model support calibration in itself.

Another way of calibration is using the Calico software tool which is freely provided by Softstar Systems [13]. Calico support three types of calibration methods: COCOMO 81 style, COCOMO II style and Multiplier & Exponent calibration. COCOMO 81 and COCOMO II style calibrate multiplier only. COCOMO II style calibration multiplier is calculated by using following set of equations (4). Calico model gives the coefficient of multiplier and exponent. Also it gives the

performance measure values: MMRE and PRED.

$$x = \sum_{i=1}^n \ln(\text{Actual} - \text{Effort}_i)$$

$$y = \sum_{i=1}^n \ln(EAF_i), \quad EAF_i = \prod_{j=1}^{16} EM_{i,j}$$

$$z = \sum_{i=1}^n \left[\text{Exponent} - \text{Base} + \frac{SF_i}{100} \right] * \ln(KSLOC_i) \quad (4)$$

$$A = e^{(x-y-z)/n} \Leftarrow \text{calibrated multiplier}$$

where, n = # projects, d = # cost drivers

In this research, we used the Calico tool for the calibration. First, we used COCOMO II style calibration model. After that we also additionally execute other Methods: Multiplier Calibration with COCOMO 81 style model. Multiplier & Exponent calibration model.

After the calibrations, we also calculated the performance measure of original COCOMO II PA model by using actual data in order to compare the accuracy enhancements by the calibration.

D. Calibration Modeling: COCOMO II style. Multiplier

Table 3 shows the performance measure value by using the original coefficient of COCOMO II PA mode and the result of calibration using COCOMO II.

Table 3. Performance measure of original COCOMO II and Calibration result

		Multiplier	MMRE	PRED (0.20)	PRED (0.25)	PRED (0.30)
Original COCOMO II	Effort	2.94	1,099.8	0	0	0
	Schedule	3.67	171.0	22	29	30
Calibration Result	Effort	0.3082	52.58	22	33	45
	Schedule	4.2183	79.85	16	23	36

In case of effort, the accuracy of estimation performance, PRED(0.30), is enhanced from 0% to 45% by calibration. This means that we can improve the accuracy of the model estimation by calibration and we can use it practically in our case if we take more effort to it.

If we compare the coefficient of multiplier with original value of multiplier – effort(2.94), schedule(3.6), the calibrated value – effort(0.31), schedule(4.2) – especially effort is significantly different from the original value. This means that characteristics or environment of our case is different from America's case.

Performance measure, MMRE and PRED, shows that calibrated model is not good fit to our case. Literature review [7] on the performance of effort estimation provided the information that the level of PRED(0.25) is between 64 and 83. Thus our calibration result is not satisfactory to apply in our case directly.

The result of calibration using COCOMO 81 style is shown table 4. MMRE value is too high and PRED(0.30) value is too low compared to above mentioned. Thus calibration model by COCOMO 81 style is not good fit for practical use.

Table 4. Calibration result using COCOMO 81 style

	Multiplier	MMRE	PRED (0.20)	PRED (0.25)	PRED (0.30)
Effort	0.1651	50.22	12	15	20
Schedule	6.2556	125.07	22	29	29

E. Calibration Modeling: Multiplier & Exponent

The result of calibration both multiplier and exponent is shown table 5. If we compare the coefficient of multiplier with original value of multiplier, the calibrated value - effort(0.74), schedule(3.2) – especially effort is significantly different from the original value as with the same case of COCOMO II style.

If we think about the effort multiplier, the value of MMRE is 41.60 and the value of PRED(0.30) is 59. Thus our calibration result is not satisfactory to apply in our case directly. But, the performance measure of effort, PRED(0.30), is 14% enhanced compared to the COCOMO II style multiplier only. It gives us the research problem that calibrating both multiplier and exponent is more effective than calibrating multiplier only. We will let this problem to the further research work.

Table 5. Calibration result using multiplier and exponent

	Multiplier	Exponent	MMRE	PRED (0.20)	PRED (0.25)	PRED (0.30)
Effort	0.7370	0.6103	41.60	44	53	59
Schedule	3.1773	0.4180	78.48	19	33	36

F. Evaluation of result

The results of these calibrations are not satisfactory to use directly in the area of defense software development project. We can think about the reasons as follows:

Most of the defense R&D projects except developing information or command/control systems have the activities on integrating hardware and software. Data used in this research are mainly composed of hardware/software integration projects, not developing software only projects. Therefore it is not easy to extract the portion of software development effort exactly.

Also project schedule is mainly dependent on the final hardware/software integration result, not the only completion of software development. The consistency of the data measurement problem can be the one reason.

If the feature of project is different from other projects, the actual data on effort and duration can be different from each other even though software size is similar and we measured these in a right way. Our assumption is that all of the cost drivers except RELY have normal value. But it may not be true. There exist many software with similar sloc size but different Person Months and duration in our data because of the different project features.

NASA effort data [7] give us interesting figure. The average of actual person months per ksloc is 4.91 PMs/ksloc with minimum value of 1.85 PMs/ksloc and maximum value of 9.68 PMs/ksloc. In our case, the average of actual person months per ksloc is 0.46 PMs/ksloc with minimum value of 0.10 PMs/ksloc and maximum value of 1.56 PMs/ksloc. This can be one reason for the difference between the original COCOMO II model and the actual effort data.

IV. CONCLUSION

This paper has described the calibration process of the COCOMO II Post-Architecture model by using 86 historical data on Korea defense domain. The result has indicated that calibration can enhance the accuracy of the original estimation model. By using PRED(0.30) performance measure, we found that estimation performance is enhanced from 0% to 45% compared to the original the COCOMO model. But calibration results shown in this paper are far from being satisfactory for cost estimation purpose not yet.

Also, we found that calibrating both multiplier and exponent estimate is better fit than calibrating multiplier estimate only, even if it is not verified statistically. We need more accurate data for meaningful result.

As we already mentioned above, there are some limitations in this research. To overcome these limitations, our suggestion for future works to do is as follows:

First, enlarge historical data set with various SLOC size and Person Month. Additionally reinvestigate the project data by using the COCOMO survey format. After collecting more populated data, execute calibration with various categorization of the project data in terms of project characteristics.

Second, calibrate the seventeen cost drivers per each. This can make the calibration model more accurate for software cost estimation.

Third, compare the performance of calibration result on the both multiplier and exponent estimate with calibration result on the multiplier estimate only. Define the criteria for comparison and determine which method is more useful.

Finally, we can consider another cost driver which reflecting

defense software development project characteristics, such as, inter-operability for the Network Centric Warfare(NCW), degree of testing difficulty for real-time embedded software, degree of security and so on.

In conclusion, we can use the COCOMO II model by doing these further works successfully, in the defense software development cost estimation in Korea.

REFERENCES

- [1] V. Nguyen, B. Steece, B. Boehm, "A constrained Regression Technique for COCOMO calibration", sunset.usc.edu/csse/TECHRPTS/2008/usc-csse-2008-806
- [2] H. Leung, Z. Fan, "Software cost estimation", cs.pitt.edu/chang/handbook
- [3] B. Clark, S. Chulani, B. Boehm, "Calibrating the COCOMO II Post-Architecture Model", *Proc. 20th Int'l Software Engineering Conference*, 1998
- [4] Z. Chen, T. Menzies, D. Port, and B. Boehm, "Finding the right data for software cost modeling," *IEEE Software*, Nov 2005.
- [5] S. Chulani, B. Boehm, and B. Steece, "Bayesian analysis of empirical software engineering cost models," *IEEE Transactions on Software Engineering*, vol. 25, July/August, 1999.
- [6] T. Menzies, Z. Chen, J. Hihn, K. Lum, "Selecting Best Practices for Effort Estimation," *IEEE Transactions on Software Engineering*, 2006
- [7] T. Menzies, D. Port, Z. Chen, J. Hihn, S. Stukes, "Validation Methods for Calibrating Software Models", *Proc. 27th Int'l Software Engineering Conference*, 2005
- [8] M. Jorgensen, M. Shepperd, "A Systematic Review of Software Development Cost Estimation Studies," *IEEE Transactions on Software Engineering*, v.33 n.1, p.33-53, January 2007
- [9] B.W. Boehm, "Software Engineering Economics," Prentice Hall, 1981.
- [10] S. Pfleeger, F. Wu, R. Lewis, "Software cost estimation and size methods : issues and guidelines", RAND, 2004
- [11] B. Boehm, S. Chulani, "Software Development Cost Estimation Approaches – A Survey", USC, 2000
- [12] M. Korte, D. Port "Confidence in software cost estimation results based on MMRE and PRED", *PROMISE'08*, May 12-13, 2008, Leipzig, Germany
- [13] D. Ligett, "Techniques for Calibrating COCOMO Estimating Models", <http://www.softstarsystems.com>