GLOBAL JOURNAL OF PURE AND APPLIED SCIENCES VOL 16, NO. 4 2010: 479-492

 ISSN 1118-0579
www.globaljournalseries.com; Email: info@globaljournalseries.com

AN IMPROVED COCOMO SOFTWARE COST ESTIMATION MODEL

DUKE, STEPHEN OROK OKOR AND OBIDINNU, JULIUS NWAFILI

(Received 19 April 2010; Revision Accepted 7, June 2010)

ABSTRACT

In this paper, we discuss the methodologies adopted previously in software cost estimation using the COnstructive COst MOdels (COCOMOs). From our analysis, COCOMOs produce very high software development efforts, which eventually produce high software development costs. Consequently, we propose its extension, called Improved Benchmark for COCOMO software Cost Estimation (IBCOCOMO). Here, we implement the extension by adopting development efforts in three perspectives – optimistic, pessimistic and most-likely. We carry out the implementation with four different software sizes – 10,000, 15,000, 20,000 and 50,000 lines of codes (LOC). When compared, the estimated values of the optimistic efforts (E_{opt}) estimated with COCOMO II using the sample sizes, IBCOCOMO model reduced it by 9.67%, 9.74%, 9.72% and 9.70% (≈ 10%) respectively. In order to ensure a realistic developmental effort favourable to both software developers and customers, a standard effort multiplication factor(e_r) is introduced, to further calibrate and remove any unforeseen residual errors that may defect the effort due to human factors unseen or unsighted.

KEYWORDS: COCOMO, Effort-Multipliers, Scale-Factors, Cost-drivers, Constants, Effort-Adjustment-Factors

1.0 INTRODUCTION

Software development cost estimation is an aspect of Software Engineering that deals with quantifying various costs expended in the development of software. The desirability of developing low priced and reliable software is one of the objectives of this paper.

Accurate software costs are crucial and critical because they can be used to generate Software requests, Project scheduling, Contract proposals, controls, and negotiations. When the cost of software is overestimated, it results in the commitment of more resources than are necessary to the project. However, when it is underestimated, it results in either poorly developed software or a job that may not be completed at all.

We review the COCOMOs in this paper, and analyze its elements (cost factors and effort multipliers) as propounded by Boehm in 1978, published in 1981, and updated in 1997. We maintain that the cost factors should be very low, low and nominal. Our consideration ignores such cost factor ratings as; high, very high and extra high. This reduces the incidence of high financial losses, and risks to human lives in software development.

2.0 LITERATURE REVIEW

Various authors have made contributions to the subject of software cost estimation using Algorithmic Models. Algorithmic models are mathematical models, which produce cost estimate as a function of number of variables, considered to be the major cost factor.

According to Dan (1997), there are varieties of these models available. The best known ones are Boehm's COCOMOs and Putnam's SLIM models. These models provide direct estimates of effort. Our focus in this paper is on the COCOMOs.

2.1 COCOMOs

COCOMO is derived from **CO**nstructive **CO**st **MO**del. It was derived by Boehm (Boehm, 1981). COCOMO is a simple on-line cost model for estimating the number of effort in persons per month required to develop software. It also estimates the development schedule in months, and is applicable to the large majority of software projects. The original COCOMO was first published in 1981**.**

Boehm and his colleagues had since defined an updated COCOMO, called **COCOMO II,** which accounts for recent changes in software engineering technology. The COCOMO has been widely accepted in practice. In the COCOMOs, the Code-size, S, is given in thousand Lines of Code (KLOC), while Effort (E) is in Persons-Month (PM).

2.2 GENERATIONS OF COCOMOs

Basically there are two generations of COCOMOs (the original COCOMO and COCOMO II). Boehm divided the applications of the first generation COCOMO into:

* Basic COCOMO that is applied early in the project, and
* Intermediate COCOMO that is applied after requirement specification.

In general, these take the mathematical form shown in **equation (1)** Fenton (1997.)

E = a*S^b * EAF … (1)

where E is the effort in persons-months, S is the size measured in thousands of lines of Codes (KLOC), and EAF is an effort adjustment factor. The factors a, and b depend on the development mode. In Fenton (1997), Boehm classified the COCOMOs into three development modes of Organic, Semi-detached and Embedded modes. These will be used in the computations later to describe the applications listed above.

Duke, Stephen Orok Okor, Department of Computer Science, Cross River University of Technology, Calabar, Nigeria
Obidinnu, Julius Nwafili, Department of Computer Science, Cross River University of Technology, Calabar, Nigeria

2.2.1 Basic COCOMO: According to Fenton and Pfleeger (1997), the Basic COCOMO Model computes effort as a function of program size. They maintain further that," Basic COCOMO is good for rough order of magnitude estimate of software costs, but its accuracy is necessarily limited because of the following: its lack of factors to account for differences in hardware constraints, personnel quality and experience, use of modern tools and techniques, and other project attributes known to have significant influence on costs". Basic COCOMO uses three sets of {a, b,} depending on the complexity of the software only. The factors a, and b for the Basic COCOMO are connected, as shown in **equation (2)** Boehm (1981.)

E = a (KLOC) ^b or a (KDSI) ^b … (2)

where KLOC = Lines of code in thousands.

KDSI = Delivery source instructions in thousands.

The factors a, and b and their respective values are shown in Table 1**.**

Table 1: Effort parameters (factors) for the three modes of Basic COCOMO (Boehm 1981)

Mode	(a)	(b)
Organic	2.4	1.05
Semi- detached	3.0	1.12
Embedded	3.6	1.20

Substituting these parameters or factors in **(equation 2**) computes the effort required for the software development for the three modes:

1. Organic mode:

(a). To compute the development effort, we use **equation (3).**

E = 2.4 (KLOC)^ 1.05 (PM) …(3)

(b) To calculate the total development time (TDEV) we use **equation 4**

TDEV = 2.5 (Effort) ^ 0.38 … (4)

(c) To calculate the number of people required to complete the project in the time-scale we use **equation (5.)**

N = Effort (PM)/TDEV (people) … (5)

2. Semi –detached Mode:

(a) To compute the development effort, we use **equation (6)**

Effort = 3.0 (KLOC) ^ 1.12 … (6)

(b) The total development time (TDEV) is obtained using **equation (7)**

TDEV = 2.5 (Effort) ^ 0. 35 … (7)

(c) The number of people required for computing the project in the time-schedule or time-scale is calculated using **equation (5)**

3. Embedded Mode:

(a) To compute the development effort, we use **equation (8.)**

Effort = 3.6 (KLOC) ^1.20 … (8)

(b) The total development time (TDEV) is obtained **equation (9.)**

TDEV = 2.5 (Effort) ^0. 32 … (9)

(c) The number of people required to complete the project in the time- schedule is computed using **equation (5.)**

Intermediate COCOMO: The intermediate COCOMO computes effort as a function of program size and the set of cost drivers. The Intermediate COCOMO equation for the effort is given in **equation (10)**

Effort = a (KLOC) **b* EAF … (10)

The effort factors or parameters, a, and b for the three intermediate COCOMO Model development modes are shown in Table 2**.**

Table 2: Effort Parameters for three Mode of Intermediate COCOMO Boehm (1981)

Mode	a	B
Organic	3.2	1.05
Semi- detached	3.0	1.12
Embedded	2.8	1.20

There are 15 effort adjustment factors (EAF) to predict effort and schedule. These cost drivers are shown in Table 3 (see Appendix A.) The cost drivers are grouped into four categories: Product, computer, personnel and project. Each cost driver is rated on a six–point ordinal scale ranging from low to high importance. The product of all effort multipliers is the EAF. The ratings are described as follows:

Very Low: Slight inconvenience
Low: Easily recoverable losses
Nominal: Moderate, easily recoverable losses
High: High financial losses
Very High: Risk to human life
Extra High: High risk to human life

The development modes for efforts, schedule, TDEV and the required numbers of person (N) for the three modes are given in equations (**11), (12) and (13), (14), (15)** and (**16), (17), (18)** and **(19)** respectively.

(1.) Organic mode:

(a) Effort = 3.2 (KLOC)^ 1.05 * EAF …(11)
(b) TDEV 2.5 (Effort) ^ 0.38 …(12)
(c) Dev_Cost = Effort (PM)* burdened labour rate per month … (13)

(2) Semi-detached:

(a) Effort = 3.0* (KLOC)^ 1.12 * EAF … (14)
(b) TDEV = 2.5*(Effort) ^ 0.35 … (15)
(c) Dev_Cost = Effort (PM) * burdened labour rate per month … (16)

(3) Embedded Mode:

(a) Effort = 2.5* (KLOC)^ 1.20 * EAF …(17)
(b) TDEV = 2.5* (Effort) ^ 0.32 …(18)
(c) Dev_Cost = Effort (PM) * burdened labour rate per month … (19)

2.3 COCOMO II

COCOMO II is the update of Intermediate COCOMO. According to Fenton and Pfleeger (1997) COCOMO II accounts for recent changes in software engineering technology. The overall COCOMO framework remained about the same but significant changes were found to be necessary to keep pace with the changing nature of software development and evolution. Boehm *et al* (2000) demonstrate that, these changes have included a move away from the Waterfall process model towards evolutionary, incremental, and spiral models; product line management approaches to software reuse; application capabilities; and graphics user interface builder tools that made traditional size metrics such as source line of code (SLOC) inappropriate.

The COCOMO development modes (organic, semidetached and embedded) have been replaced by a set of scale factors- (precedentedness, development-flexibility, architecture and risk resolution, team-cohesiveness, and process maturity). These factors enable project managers to control their project's economies and diseconomies of scale. Some multiplicative cost drivers (development for reuse, degree of documentation, multisite development) were added to COCOMO II. Turnaround Time cost driver was dropped; the modern programming practices cost driver was merged into the process maturity scale factor; and the requirements volatility cost driver was changed into a size factor.

The main size parameter of COCOMO was changed from Delivered Source Instructions to a user-determined mix of SLOC and function points in COCOMO II; it was also changed to more detailed non-linear model of software reuse effect; and provided a family of models (Applications Composition, Early Design, and Post-Architecture) turned to the information available at different stages of the development process.

COCOMO II was extended to address the emerging trends such as Rapid Application Development (RAD) and Commercial Off-The-Shelf (COTS) integration. Boehm *et al* (1997) demonstrate that, COCOMO II provides up–to -date support for business software, object oriented software, software created via spiral or evolutionary development models, and software developed using commercial–off-the-shelf application composition utilities. The included Application Composition model is for early prototyping efforts while the Early Design and post-Architecture models are for subsequent portions of the lifecycle. The Early Design model equation is as shown in **equation (20)**

Effort = a* (KLOC)* EAF

… (20)

where a, is a constant, provisionally set to 2.45.

The Effort Adjustment Factor (EAF) is calculated as in the original COCOMO model using the 17 cost drivers. The scale factors for COCOMO II is presented in Table 4 (see Appendix B), while the EAF is calculated using the 17 cost drives shown in Table 5 (see Appendix C)

2.3 CALIBRATION TECHNIQUES OF COCOMO II

Multiple Regressions method which is one of the statistical approaches has been used to calibrate successive versions of COCOMO II. In this section, we discuss it briefly.

Multiple Regressions expresses the response (e.g. Person-Months) as a linear function of K predictors (e.g. Source Lines of Code, etc.) This linear function is estimated from the data using the ordinary least squares approach. A typical multiple regressions model is presented in **equation (21)**.

$$y_t = \beta_0 + \beta_1 X_{t1} + \ldots + \beta_k X_{tk} + \varepsilon_t \quad \ldots (21)$$

where $X_{t1} \ldots X_{tk}$ are the values of the predictor (or repressor) variables for the t_{th} observation, $\beta_0 \ldots \beta_k$ are the coefficients to be estimated, ε_t is the usual error team, and y_t is the response variable for the t_{th} observation. The COCOMO II Post Architecture model has the form in **equation (22)** for estimating effort.

$$\text{Effort (E)} = (A) * (size)^{B+\sum w_i} * \prod EM_j \; 1 \le i \le 5, 1 \le j \le 17 \ldots . (22)$$

Park (1992) explains that A = Multiplicative constant; Size = size of the software project measured in terms of KSLOC (Thousand Source Lines of Code), Function points or object point.

W_i = Scale factor

EM = Effort Multiplier

This COCOMO II model equation can be linearized by taking logarithms on both sides of **equation (22)** as shown in **equation (23)**

In (PM) = β_0 + β_1 . 1.01. . . In (size) + β_2.SF1 In (size) + Λ + βc. SF_5 In (size) + β_7. In (EM1) + β_8 . In (EM_2) + Λ+ β_{22}. In (EM_{16}) + β_{23}. In (EM_{17}) … (23)

According to Clark, Chulani, and Boehm (1998), the 1997 calibration used a dataset consisting of 83 completed projects. The regression estimates the β coefficients associated with the 5 scale factors and 17 effort multipliers with some of the estimates being counter initiative.

2.4 SOFTWARE COST ESTIMATION METRICS

Conventionally, in software cost estimation, the metrics represent the estimates of interest. These are objective measure criteria used for estimating the cost, schedule, development cost and the number of software personnel. The metrics are listed below:

1. **Effort (pm)**: This is the amount of human effort committed to the development of the software.
2. **Schedule (months):** This is the time taken to complete the software development project.
3. **Development Cost:** This is the amount of money to be involved in developing the software.
4. **Software Personnel:** This is the number of people required in developing the software.

The effort depends solely on the estimated size; while the schedule depends solely on the effort. This implies that effort is a function of size and number of cost drivers. Size is the primary cost factor in models, while cost drivers or Effort Adjustment Factors (EAF) are the secondary cost factors. Cost drivers are the characteristics of the project, process, products or resources that influence the effort. They are used to adjust the preliminary effort estimate provided by the primary cost factor. The size could be Lines of Code in thousand (KLOC) or Function Point (FP). Models depend on the size in order to generate the required effort, which then can be converted into the monetary and schedule figures.

3.1 METHODOLOGY

We derive our model from COCOMO II and Intermediate COCOMO. We use COCOMO II Cost drivers to produce the effort multiplier (EM) (see appendix B). The proposed model is composed of the following:

Effort Multipliers (EM): This is the product of all the COCOMO II Cost drivers on ordinal scale rating. For the Optimistic Effort Multiplier (EM_{opt}) (see appendix D for the four software sizes used) the rating chosen are kept within the bound of nominal, high and very high (see **Table 8**, Appendix D.) For the Pessimistic Effort Multiplier (EM_{pest}), the bound is within very low, low, and nominal (see **Table 10**, Appendix D). The Most Likely Effort Multiplier (EM_{moly}) is obtained by finding the mean of optimistic and pessimistic effort multiplier.

Scale Factors (W_i): This replaces the organic development mode of the Intermediate COCOMO model. The scale factors for both optimistic and pessimistic approaches are maintained at extra high respectively.

Constants: The constants used include

- Effort multiplicative constant (A = 2.94)
- Exponential effort calibration constant (B = 1.01)

The effort multiplier (EM) and scale factor (W_i) (appendix D) are used to adjust the preliminary effort provided by the primary cost factor (size). The effort is the most paramount software cost estimation metric. The schedule, development cost software personnel estimation depends on the effort.

3.1.1 Model Formulation

The new software cost estimation model is derived from COCOMO II Post Architecture model **equation (22)** for estimating effort. A new variable, e_r has been introduced, as seen in **equation (24)**

$$\mathrm{E}_{\mathrm{opt}} = (A - (A * e_r)) * (size)^{B+\sum w_i} * \left(\prod EM_{jopt} - \left(\prod EM_{jopt} * e_r\right)\right) \ldots (24)$$

e_r is called further effort multiplier reduction. It is introduced to reduce the effort multiplicative constant (A) and the effort multiplier (EM) by 5 percents thereby leading to the reduction of the main effort to a reasonable estimated effort, favourable to both software developers and their clients/customers.

3.1.2 Computation of Efforts

i. SIZE: 10, 000 lines of codes

NEW MODEL
For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = (A - (A * e_r)) * (size)^{B+\sum w_i} * \left(\prod EM_{jopt} - \left(\prod EM_{jopt} * e_r\right)\right) = 11.58(pm)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = (A - (A * e_r)) * (size)^{B+\sum w_i} * \left(\prod EM_{jpesst} - \left(\prod EM_{jpesst} * e_r\right)\right) = 11.43(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = \left(\mathrm{E}_{\mathrm{opt}} + E_{pesst}\right)/2 = 11.51(pm)$$

OLD COCOMO II MODEL
For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = A * (size)^{B+\sum w_i} * \prod EM_{jopt} = 12.82(pm)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = A * (size)^{B+\sum w_i} * \prod EM_{jpesst} = 12.70(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = \left(\mathrm{E}_{\mathrm{opt}} + E_{pesst}\right)/2 = 12.76(pm)$$

ii. SIZE: 15, 000 lines of codes

NEW MODEL
For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = (A - (A * e_r)) * (size)^{B+\sum w_i} * \left(\prod EM_{jopt} - \left(\prod EM_{jopt} * e_r\right)\right) = 17.43(pm)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = (A - (A * e_r)) * (size)^{B+\sum w_i} * \left(\prod EM_{jpesst} - \left(\prod EM_{jpesst} * e_r\right)\right) = 17.26(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = \left(\mathrm{E}_{\mathrm{opt}} + E_{pesst}\right)/2 = 17.35(pm)$$

OLD COCOMO II MODEL
For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = A*(size)^{B+\sum w_i} * \prod EM_{jopt} = 19.31(pm)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = A*(size)^{B+\sum w_i} * \prod EM_{jpesst} = 19.13(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = (\mathrm{E}_{\mathrm{opt}} + E_{pesst})/2 = 19.22(pm)$$

iii. SIZE: 20, 000 lines of codes

NEW MODEL

For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = (A-(A*e_r))*(size)^{B+\sum w_i} * \left(\prod EM_{jopt} - \left(\prod EM_{jopt} * e_r\right)\right) = 23.31(pm)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = (A-(A*e_r))*(size)^{B+\sum w_i} * \left(\prod EM_{jpesst} - \left(\prod EM_{jpesst} * e_r\right)\right) = 23.08(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = (\mathrm{E}_{\mathrm{opt}} + E_{pesst})/2 = 23.20(pm)$$

OLD COCOMO II MODEL

For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = A*(size)^{B+\sum w_i} * \prod EM_{jopt} = 25.82(mp)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = A*(size)^{B+\sum w_i} * \prod EM_{jpesst} = 25.57(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = (\mathrm{E}_{\mathrm{opt}} + E_{pesst})/2 = 25.70(pm)$$

iv. SIZE: 50, 000 lines of codes

NEW MODEL

For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = (A-(A*e_r))*(size)^{B+\sum w_i} * \left(\prod EM_{jopt} - \left(\prod EM_{jopt} * e_r\right)\right) = 58.81(pm)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = (A-(A*e_r))*(size)^{B+\sum w_i} * \left(\prod EM_{jpesst} - \left(\prod EM_{jpesst} * e_r\right)\right) = 58.23(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = (\mathrm{E}_{\mathrm{opt}} + E_{pesst})/2 = 58.52(pm)$$

OLD COCOMO II MODEL

For Optimistic Effort:

$$\mathrm{E}_{\mathrm{opt}} = A*(size)^{B+\sum w_i} * \prod EM_{jopt} = 65.13(pm)$$

For pessimistic Effort:

$$\mathrm{E}_{\mathrm{pesst}} = A*(size)^{B+\sum w_i} * \prod EM_{jpesst} = 64.52(pm)$$

For Most Likely Effort:

$$\mathrm{E}_{mokly} = (\mathrm{E}_{\mathrm{opt}} + E_{pesst})/2 = 64.83(pm)$$

3.1.3 Calculation of Development Cost (Dev_Cost)

Suppose that a software developer charges his/her client ₦25,000 Nigerian currency for burden labour rate per month upon developing software of 10, 000 lines of code, then the software development cost ($\mathrm{Dev\_Cost}_{\mathrm{opt}}$) will be as calculated below:

Old Model (COCOMO II):

$\mathrm{Dev\_Cost}_{\mathrm{opt}}$ = **$\mathrm{E}_{\mathrm{opt}}$** * ₦25, 000* 1

=12.82 * ₦25, 000*1 = ₦320, 500 per month

If the delivery time is one year (schedule) then the optimistic development cost will be ₦3, 846, 000.00.00.

New Model (IBCOCOMO):

$\mathrm{Dev\_Cost}_{\mathrm{opt}}$ = **$\mathrm{E}_{\mathrm{opt}}$** *₦25, 000* 1

= 11.58 * ₦25, 000* 1 = ₦295, 500 per month

If the delivery time is one year (schedule) then the optimistic development cost will be ₦3, 546, 000. For the rest of the sizes, the development costs are shown in Table 13.

4.1 RESULTS

Table 12 shows the summary of the efforts estimated for the various software sizes for optimistic, pessimistic and most likely. The bar chart in **Figure 1.0** illustrates the relationship between the new and old COCOMO II models.

Table 13 on the other hand shows the development cost (Dev_Cost) estimated using the two models

Table 12: Summary of Efforts Estimated by the New and Old Models

EFFORT	MODEL		SIZE
	IBCOCOMO (NEW)	**COCOMO II (OLD)**	
Optimistic	11.58 (pm)	12.82 (pm)	**10 kloc**
Pessimistic	11.46 (pm)	12.70 (pm)	
Most Likely	11.53 (pm)	12.76 (pm)	
Optimistic	17.43 (pm)	19.31 (pm)	**15 kloc**
Pessimistic	17.26 (pm)	19.13 (pm)	
Most Likely	17.35 (pm)	19.22 (pm)	
Optimistic	23.31 (pm)	25.82 (pm)	**20kloc**
Pessimistic	23.08 (pm)	25.57 (pm)	
Most Likely	23.20 (pm)	25.70 (pm)	
Optimistic	58.81 (pm)	65.13 (pm)	**50kloc**
Pessimistic	58.23 (pm)	64.52 (pm)	
Most Likely	58.52 (pm)	64.33 (pm)	

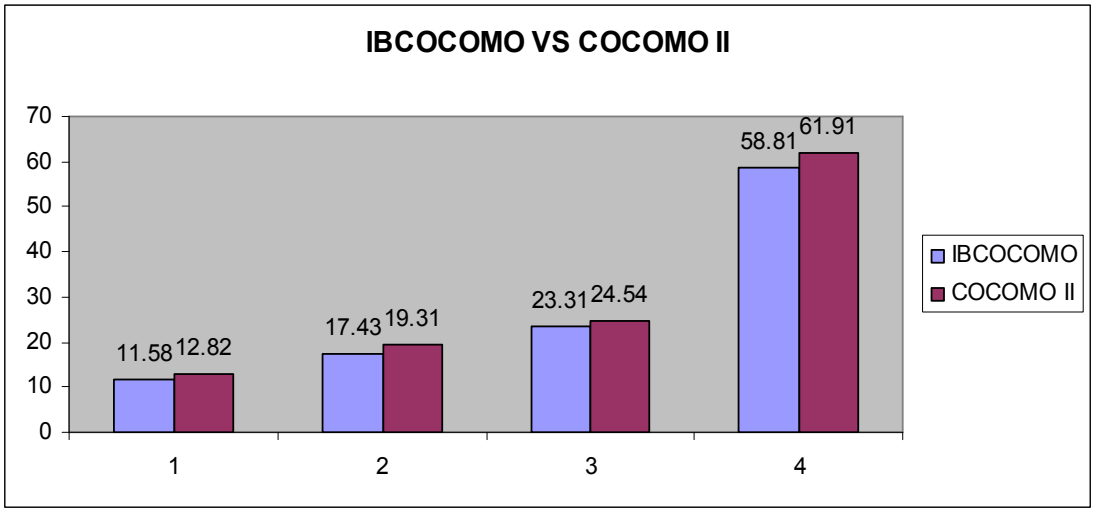

Fig. 1.0: IBCOCOMO and COCOMO II Relationship

Furthermore, for the software development cost, the new model provides lesser cost compared to the old COCOMO II model as illustrates.

Table 13: Summary of Development cost Estimated by the New and Old Models

Development Cost	MODEL		SIZE
	IBCOCOMO	COCOMO II (OLD)	
Optimistic	₦295, 500 per month	₦320, 500 per month	10 kloc
Pessimistic	₦286, 500 per month	₦317, 500 per month	
Most Likely	₦288, 250 per month	₦319, 000 per month	
Optimistic	₦435, 750 per month	₦482, 750 per month	15 kloc
Pessimistic	₦431, 500 per month	₦478, 250 per month	
Most Likely	₦433, 750 per month	₦480, 500 per month	
Optimistic	₦582, 750 per month	₦645, 500 per month	20kloc
Pessimistic	₦577, 000 per month	₦639, 250 per month	
Most Likely	₦580, 000 per month	₦642, 500 per month	
Optimistic	₦1,470, 250 per month	₦1,628, 250 per month	50kloc
Pessimistic	₦1,455, 750 per month	₦1,613, 000 per month	
Most Likely	₦1,463, 000 per month	₦1,610, 250 per month	

5.1 DISCUSSIONS

Table 12 shows the estimated software development efforts obtained from the different estimated software sizes. The table reveals that our new model provides lesser software development efforts than the COCOMO II model does.

It has been a known fact that software development cost is a function of software effort. Higher software development effort leads to higher software development cost. The new model provides smaller efforts compared to COCOMO II, and hence provides smaller software development cost as shown in **Table** 13.

To further present our support for the new model, we use a bar chart to illustrate the comparison. Looking at the bar chart (**figure 1.0**), one observes that the Old model (COCOMO II) produces taller bars than the New model (IBCOCOMO). The implications of having taller bars imply higher efforts, which consequently lead to higher development cost. On the other hand, the shorter bars imply lower efforts, which also lead to lower development costs.

6.1 CONCLUSION

This work presents an overview of the COCOMO software cost estimation models. It introduces a modified version of COCOMO II. The multiplicative (A) and Effort multiplier (EM) were adjusted by the introduction of an adjustment factor (e_f). The new model is an improvement over the previous ones. Software developers will find this model an attractive resource.

REFERENCES

Boehm, B. W., 1981. Software Engineering Economies. Engle wood Cliffs, NJ: Prentice-Hall.

Boehm, B. W., Abts, C., Clark, B. and Devnanni-Chulani, S., 1997. COCOMO II Model Definition Manual. The University of Southern California.

Clark, B., Chulani, S. and Boehm B., 1998. "Calibration Results of COCOMO II 1997" International Conference on software Engineering.

Dan. S., 1997. Software Cost Estimation Bournemouth University Retrieved 20/09/2009 from http://www.ecfc.u-net.com/cost/index.htm

Duke, S. O. O., 2008. Development of a Benchmark Software Cost Estimation Package

Fenton, N. E., 1997. "Software Cost Estimation", Information and Software Technology, Vol. 34 no.10. PP. 627-639.

Fenton, N. E. and Pfleeger, S. L., 1997. Software Metrics: A Rigorous and Practical Approach. PWS Publishing Company.

Johnson, K., 1998. Software Cost Estimation Matrices and Models. Department of Computer Science, University of Calgary, Alberta, Canada T2NIN4 Retrieved on 21/09/2009 from http://sern.ucalgary.ca/courses/seng/621/W98/johnsonk/cost.htm

Park, R. E., 1992. “Software size measurement: A Framework for Counting source statement” CMU-SEI-92 TR -20, Software Engineering Institute, Pittsburg. PA.