International Journal of Engineering Research and Development
***e*-ISSN : 2278-067X, *p*-ISSN : 2278-800X, www.ijerd.com**
Volume 10, Issue 6 (June 2014), PP.11-15

Comparative Analysis Between Fpa and Cocomo Techniques For Software Cost Estimation

Abubaker Ali[1],Noraini Ibrahim[2]
University Tun Hussein Onn Malaysia (UTHM), Parit Raja, 86400 Batu Pahat, Johor
[1]may.91985@yahoo.com
[2]noraini@uthm.edu.my

Abstract:- Software cost estimation is the process of predicting the effort required to develop a software system. The basic input for the software cost estimation is coding size and set of cost drivers; the output is effort in terms of Person-Months. Here, the use of FPA and COCOMO has been proposed for the estimation of software project cost. It was found the FPA results for first case study WBDDDS was 12.506 Persons-Months with the total cost of $65031.2. Then, by using COCOMO, the result was 16.286 Persons-Months with the total cost of $84,687.2. However, COCOMO and FPA results for second case study (SBOBSE) are 19.62 Persons-Months with the total cost of $102,024 and 19.354 Persons-Months with the total cost of $100,640.8, respectively. In conclusion, comparison of the two models is done.

Keywords:- Function point analysis (FPA), Constructive Cost Model (COCOCMO**)**, software cost estimation, lines of code (LOC)

I. INTRODUCTION

The estimating software development cost remains a complex problem, one that continues to attract a considerable amount of research attention. Improving the accuracy of the cost estimation models available to project managers would facilitate a more effective control of time and budgets during the software development. The needs for a reliable and accurate cost estimation in software engineering have been an ongoing challenge for software engineers in the last decade [1], [2] and [3].

The Standish Group Chaos Report recently reported that several software projects (around 66%) are delivered with some delay, over-budget, and many are not even finished. Commonly, the main cause of these problems is the failure of the software development cost estimation (SDCE) [4].

The software cost estimation is the process of predicting cost for the development of the software. The software cost is the amount of cost in either person days or person hours necessary for conducting the tests, the most commonly used methods for predicting software development cost are Function Point Analysis, Constructive Systems Engineering Cost Model (COSYSMO), SEER for Software (SEER-SEM), Putnam model, and Constructive Cost Model (COCOMO) [5].

The function point analysis (FPA) is a method of quantifying the size and complexity of a software system in terms of the functions that the system delivers to the user. The function does not depend on the programming languages or tools used to develop a software project. FPA is a standard method to measure the software development from the user's point of view. The past three decades of the use of FPA have shown that it is a proven method [6], [7], [8].

Boehm develops the Constructive Cost Model (COCOMO). It is based on linear-least-squares regression. Using the line of code (LOC) as the unit of measure for the software size itself contains so many problems. These methods failed to deal with the implicit non-linearity and interactions between the characteristics of the project and effort [2].

II. OVERVIEW OF COST ESTIMATION

As stated by Parkinson's Law which demonstrates that the cost of the task depends to a great extent on what assets are accessible in the association and that the life cycle of the undertaking is to be extended to meet the due date and as per the assets accessible to the association [9].

R. Suggested Analogy, Top down and Bottom up methodologies as three estimation procedures, which the Analogy strategy assessing the task by utilizing verifiable data of at one time finished undertakings and to contrasted and as of now existing data on finished activities. The second approach focuses on the general qualities rather than the useful and non-practical prerequisites of the framework to be created, while the bottom up methodology it recognizes every single segment and afterward consolidates all of them to give the general obliged estimation for the task which is observed that it gives the most point by point estimation [10]

Reported that various of undertakings either wiped out or misses its conveyance dates. Then again, a more than a large portion of the undertakings significantly overwhelms their estimation as demonstrated and

which this methodology focused around a few overviews directed [11]. The related mulled over showed that the viable software estimation is a standout amongst the most essential and troublesome software improvement exercises [9]. The over-assessing framework and the under-evaluating arrangement of a venture are both terrible for diverse reasons, which the overestimating will reason for an undertaking to take in any event as long as it was assessed, inasmuch as the other framework (under-assessing) a task will prompt under staffing, under checking the quality confirmation exertion, short calendar [12]

In view of Galorath and Evans hunt which were discovered reasons down the falls flat of the software ventures throughout their a serious examination of the web locales, which that; lack of common sense the venture, inadequate prerequisites designing, abruptly choices at the early phases of the task and wrong estimations which think about as the most paramount reasons [13].

In the of late, Boehm, B. remarked that there are substantial amounts of cost dissection strategies accessible, yet was found that not generally sheltered to utilize. The least complex strategy is to build cost appraise in light of the ordinary costs or profit rates of past activities. A percentage of the basic strategies are helpful if the new extend does not have any cost-discriminating contrasts from the past activities. Nonetheless, they are unsafe if the discriminating variable of the cost driver has been disposed of [14].

III. RELATED WORK

A. Cost Estimation Techniques

This system is shaped to give a numerical methodology to do the software estimate. These scientific mathematical statements are focused around examination and chronicled data and utilization inputs, for example, Source Lines of Code (SLOC), and some other cost drivers like dialect, technique, hazard appraisals, aptitude levels, and so forth. These algorithmic models have been widely taken a shot at. A few models have been shaped focused around it selves, for example, COCOMO models, capacity point and Putnam displays and known as based models [15]. There are numerous courses in the writing to gauge the cost. Essentially, the cost estimation techniques are characterized into two gatherings, which are number crunching and non-math [16]. In this study number juggling strategy will use to talk about the assessment of the cost.

B. Constructive Cost Model (COCOMO)

The starting evaluation, which is additionally alluded to as the ostensible appraisal, is dictated by the static single variable model comparison utilizing KLOC as the measure of size. The sort decides the starting exertion in individual month. The constants (a, b) rely on upon the kind of the undertaking as demonstrated in table 1 and emulating three classes of software [15].

EFFORT = a* (KLOC) ^{b} (1)

Table I: Basic COCOMO Calculating [15]

Development Mode	Basic Effort Equation
Organic	**2.4**
Semi Detached	**3.0**
Embedded	**3.6**

1. Intermediate COCOMO

This model processes advancement exertion of the software as a system size capacity and a set of cost drivers. These incorporate subjective evaluations of the fittings, work force and undertaking traits, and items. The cost drivers mentioned below can be put into groups.

1. Product attributes
   a. Required software reliability
   b. Size of application data base
   c. Complexity of the product
2. Hardware attributes
   a. Run-time performance constraints
   b. Memory constraints
   c. Volatility of the virtual machine environment
   d. Required turnaround time
3. Personnel attributes
   a. Analyst capability
   b. Software engineer capability
   c. Applications experience
   d. Virtual machine experience

e. Programming language experience

4. Project attributes
   a. Use of software tools
   b. Application of software engineering methods
   c. It extends from a high to a low in the matters of worth. An exertion multiplier focused around the evaluations is chosen from the tables, which was distributed by Boehm, and an exertion appraisal component (EAF) is gotten as an item from these multipliers. The typical values for EAF range from 0.9 to 1.4. The intermediate COCOMO model takes the form:

$$EFFORT = a^* (KLOC)^{b} * EAF \qquad (2)$$

Where, effort in person-months and KLOC are the estimated number of delivered lines of code for the project.

Table II: Intermediate COCOMO Calculating [15]	
Development Mode	Intermediate Effort Equation
Organic	3.2
Semi Detached	3.0
Embedded	2.8

C. Function Point Analysis (FPA)

There are two parts in the model that are Unadjusted Function Point (UFP) and Adjusted Function Point (AFP). The UFP consists of five components. They are External Inputs (EI), External Outputs (EO), External Inquires (EQ), Internal Logical Files (ILF), and External Interface date (EIF) [17].

Table III: Function Types and Weights [17]			
Function Type	Simple	Average	Complex
External Input	3	4	6
External Output	4	5	7
Internal Files	7	10	15
External Files	5	7	10
External Inquiry	3	4	6

There are 14 GSCs components that affect the length of the project energy and each can be ranked from"0"- no influence to"5"- necessary. GSCs consist connected with 14 factors called f1, f2... f14. These types of factors are outlined in Table only two. 5. The sum of Table III components are then multiplied with given Formula 2. 4, which constitute this Adjustment Factor (AF) defined within the range. (0.65, -1.35) [17].

$$AF = 0.65 + 0.01 \sum_{i=1}^{14} fi \qquad (3)$$

Table IV: General System Characteristics (GSCS)[17]	
1	**Data Communications**
2	**Distributed Functions**
3	**Performance**
4	**Heavily Used Configuration**
5	**Transaction Rate**
6	**Online Data Entry**
7	**End User Efficiency**
8	**Online Update**
9	**Complex Processing**
10	**Reusability**
11	**Installation Ease**
12	**Operational Ease**
13	**Multiple Sites**
14	**Facilitate Change**

Then, the Unadjusted FP will be then multiplied by the UFP to develop the AFP count as given inside Equation 4. The AFP value is definitely within 35% in the original UFP physique. A diagram, which shows the method of computing FP, is given in Figure 2.1 [17].

$$\text{Adjusted FP} = \text{Unadjusted FP} \times \text{AF} \qquad (4)$$

IV. METHODOLOGY

Methodology shown, The COCOMO and FPA, the software cost to apply the estimation of the Constructive Cost Model (COCOMO) technique and using the basic COCOMO and Intermediate COCOMO in the case study 1 (Sugar Bun Online Bakery System, E-Sugarbun (SBOBSE)). Subsequently, by evaluating the software cost to apply the estimation of the Constructive Cost Model (COCOMO) technique in the case study 2 (Web-Based Dog's Diseases Diagnosis System (WBDDDS)). There are also five steps for FPA to apply this technique. Besides that, the Function Point Analysis (FPA) estimation technique and the Development Project Function Point Count and Adjustment Factor in FPA were used as well in the case study 1 (SBOBSE). The software cost was evaluated to apply the estimation using the Function Point Analysis (FPA) estimation technique in the case study 2 (WBDDDS). Then, the software cost estimation was compared using FPA and COCOMO for both case studies.

V. RESULTS AND COMPARATIVES

In this section, a summarized on the application of the FPA and COCOMO will be presented on two different projects.

1. Web Based Dog's Diseases Diagnosis System (WBDDDS)

Table V: Comparison of FPA & COCOMO in first case study(WBDDDS)

	FPA	COCOMO
Man-months	12.506	16.286
Total Cost	$65031.2	$84687.2

Table 4.35 denotes that the FPA is better than COCOMO based on a first case study where person-month in FPA is 12.506 with total cost in FPA is $65031.2 but in COCOMO, the person–month is 16.286 with total cost in COCOMO is $84687.2.

2. Sugar Bun Online Bakery System, E-Sugarbun (SBOBSE)

Table VI: Comparison of FPA & COCOMO in second case study (SBOBSE)

	FPA	COCOMO
Man-months	19.62	19.354
Total Cost	$102,024	$100,640.8

Table 4.35 denotes that the COCOMO is better than FPA based on a second case study where person-month in FPA is 19.62 person-months and total cost in FPA $102,024 total cost. However, person-month in COCOMO is 19.354 person-months and its total cost is $100,640.8.

A. Evaluate The Results Of Comparison the Estimation Method

Based on the two case studies, FPA is better cost estimation then COCOMO in the first case study (WBDDDS); the person-month was 12.506 person-months with the total cost of $65031.2, while in COCOMO, 16.286 person-months with the total cost of $84687.2. However, in the second case study (SBOBSE), it was found that in the COCOMO was better than FPA, the person-month in FPA was 19.62 and the total cost was $102,024, but the person-month in COCOMO was 19.354, and the total cost was $100,640.8. To conclusion, there is no preferred in the technical as mentioned in the above results and thus are by predilection as used by the study.

VI. CONCLUSION AND FUTURE WORK

Software cost estimation is the process of predicting the effort required to develop a software system. The basic input for the software cost estimation is coding size and set of cost drivers; the output is effort in terms of Person-Months. Here, the use of FPA and COCOMO has been proposed for the estimation of software project cost. It was found the FPA results for first case study WBDDDS was 12.506 Persons-Months with the total cost

of $65031.2. Then, by using COCOMO, the result was 16.286 Persons-Months with the total cost of $84,687.2. However, COCOMO and FPA results for second case study (SBOBSE) are 19.62 Persons-Months with the total cost of $102,024 and 19.354 Persons-Months with the total cost of $100,640.8, respectively. In conclusion, comparison of the two models is done. In present study most of the present estimation techniques have been showed methodically. Since software project managers are used to select the best estimation method based on the conditions and status of the project, characterize and comprising of estimation techniques can be useful for decreasing of the project failures. There is not any estimation method, which can be present; the best estimates in all diverse situations and each technique can be suitable in the special project, it is necessary for understanding the principles of each estimation method to choose the best. Because performance of each estimation method depends on several parameters such as complexity of the project, duration of the project, expertise of the staff, development method and so on. Some evaluation metrics and an actual estimation example have been presented in this paper just for describing the performance of an estimation method (for example COCOMO)[13].This work can be extended comparison the new methods for estimation by using COCOMO II and FPA for two case studies

REFERENCES

[1]. Attarzadeh, I., & Ow, S. H. (2010, February). A novel soft computing model to increase the accuracy of software development cost estimation. In Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on (Vol. 3, pp. 603-607). IEEE.

[2]. Boehm, B. W. (1979, September). Software engineering-as it is. In Proceedings of the 4th international conference on Software engineering (pp. 11-21). IEEE Press.

[3]. Boehm, B., Clark, B., Horowitz, E., Westland, C., Madachy, R., & Selby, R. (1995). Cost models for future software life cycle processes: COCOMO 2.0.Annals of software engineering, 1(1), 57-94.

[4]. Braga, P. L., Oliveira, A. L., & Meira, S. R. (2008, March). A GA-based feature selection and parameters optimization for support vector regression applied to software effort estimation. In Proceedings of the 2008 ACM symposium on Applied computing (pp. 1788-1792). ACM.

[5]. Matson, J. E., Barrett, B. E., & Mellichamp, J. M. (1994). Software development cost estimation using function points. Software Engineering, IEEE Transactions on, 20(4), 275-287.

[6]. Felfernig, A., & Salbrechter, A. (2004). Applying function point analysis to effort estimation in configurator development. In International Conference on Economic, Technical and organisational aspects of Product Configuration Systems, Kopenhagen, Denmark (pp. 109-119).

[7]. M. Junhai and M. Lingling, "Comparison study on methods of software cost estimation," in e-Business and Information System Security (EBISS), 2010 2nd International Conference on, 2010, pp. 1-4.

[8]. Jeng, B., Yeh, D., Wang, D., Chu, S. L., & Chen, C. M. (2011). A Specific Effort Estimation Method Using Function Point. Journal of Information Science and Engineering, 27(4), 1363-1376.

[9]. Tagra, D. (2011). Cost Estimation For Commercial Software Development Organizations. Dalhousie University Halifax, Nova Scotia: Ph.D. Thesis.

[10]. Pandian, C. R. (2004). Software Metrics: A Guide to Planning. Analysis, and Application. Auerbach Publications, 46.

[11]. Steve McConnell. Software Project Survival Guide. Microsoft Press, 1998. pp. 40-45, Nov./Dec. 2006, doi:10.1109/MITP.2006.149

[12]. K. Kavoussanakis and T. Sloan. UKHEC Report on Software Estimation, Dec 2004

[13]. Khatibi, V. & A.Jawawi, Dayang N. (2010). Software Cost Estimation Methods: A Review. Journal of Emerging Trends in Computing and Information Sciences, vol3(1), pp. 21 – 29.

[14]. Boehm, B. (2000). Safe and simple software cost analysis. IEEE software,17(5), 14-17

[15]. Kumari, S. & Pushank, S. (2013). Performance analysis of the software cost Estimation Methods: A Review. International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3(7), pp. 229 - 238.

[16]. H. Leung and Z. Fan, (2002) Software Cost Estimation. Handbook of software engineering and knowledge engineering, world scientific publications company. pp 1- 14

[17]. Kemerer, C. F. (1987). An empirical validation of software cost estimation models. Communications of the ACM, 30(5), 416-429.