International Conference on Smart Computing and Electronic Enterprise. (ICSCEE2018) ©2018 IEEE

An Empirical Analysis of Cost Estimation Models on Undergraduate Projects Using COCOMO II

Faiza Tahir
Dept. of Software Engineering
University of Gujrat Sialkot Sub Campus
Sialkot, Pakistan
faiza.tahir@uogsialkot.edu.pk

Mahum Adil
Dept. of Software Engineering
University of Gujrat Sialkot Sub Campus
Sialkot, Pakistan
mahum.adil@uogsialkot.edu.pk

***Abstract*—Now a day's software engineering has inclination towards formality. Many formal and mathematical mergers have taken place in theoretical software engineering. Giving mathematical foundation to various aspects of software engineering enhance their practicability and predictability. Cost estimation models like Constructive Cost Model (COCOMO) is one major proven example. COCOMO effort and time calculation is based on mathematical foundations. Several cost drivers having mathematical values also contributed in these calculations. This paper presents a sensitivity analysis using on Constructive Cost Model II (COCOMO II) and their relationships with the projects taken on undergraduate level using web based estimation tool Estimator. The analysis studies the behavior of cost drivers having higher impact on these projects and needs detail understanding and some has low impact. This categorization further used in functional and non-functional requirements level. As which nonfunctional requirement need more and detailed understanding at undergraduate study level**

Keywords*— *COCOMO I, COCOMO II, functional requirements verification, software cost estimation, undergraduate project

I. INTRODUCTION

Software cost estimation is one of the biggest challenge for project managers [1]. Software cost estimation is necessary for not only success of the project but also to ensure project managers about the quality and certainty of their useful risk management strategies which in turn give confidence on the return on investment [2]. As the poor estimation has become a major cause of project failure, Researchers estimated that more than 70% of the projects are failed either on budget overrun or due to unrealistic deadlines [1]-[5]. This higher number of failures proves cost estimation a challenging task. Moreover, these estimations are needed in start of the project mainly at the time of project proposal therefore, proper understanding has to develop in order to get a realistic and appropriate estimates. The main reasons of difficulty in project cost estimation are following.

- Here project cost estimation depicts both project effort estimation in effort required in person per month and time estimation in time required for all the software development life cycle.
- It is an expensive task both in terms of money and time.
- As the project proposal is the first task to initiate the project process, cost estimation comes before the project proposal, and therefore these estimates are required earlier and unfortunately done in a hurried way, overlooking many important factors.
- Experts and analysts providing estimates are often difficult to find.
- Human biasness [1].

There is various software cost estimation models proposed which are divided in two main categories i.e., algorithmic and non-algorithmic models. Algorithmic models uses different statistical analysis like regression analysis and provide estimates on the basis of historical data [1]. Disadvantages of algorithmic models is that they lack in robustness and effectiveness, it is also difficult to get some inputs at the very early stages of projects. Moreover they cannot establish a sound relationship between input factors and doesn't handle categorical values [3]. Examples are Software Life Cycle Management (SLIM) model and the most popular Cost Constructive Model [4]-[12]-[14]-[20]-[21].

Non algorithmic techniques include Parkinson's technique based on Parkinson's law to estimate, price to win put the price in front and then estimate according to it, expert judgment uses experts opinion to analyze the project effort, analogy compare the project with similar other projects to find out the nearest estimates [3]

These models have following problems in it [14].

- Difficulty in handling categorical and uncertain data.
- Difficult to draw conclusion from available data.
- Rapidly changing environment, and customization in software.
- Biasness in opinions of experts and difficult to incorporate it.
- Unavailability of historical data, and limitations of available historical data.

Software computing techniques comes under the non-algorithmic category which includes fuzzy logic systems, genetic algorithm, artificial neural networks, bayesian network, evolutionary computation and other machine learning approaches like regression trees and rule induction, swarm intelligence etc. [1]-[3]-[4]-[12]-[14]-[20]-[21].

Some researchers use the combination of algorithmic and non-algorithmic techniques like COCOMO with analogy [12].

Neural network with fuzzy logic systems [10] and artificial neural network with COCOMO II [15]-[16]-[17]-[18]-[19].

This paper will also present the use of two techniques i.e., expert judgment and COCOMO II to enhance the predictability of COCOMO II for the undergraduate projects of universities. It will identify the major cost drivers along with the predictability and effort affecting these projects

In Section II we will brief description of COCOMO I & II. Section III of the paper presents the methodology and case study. Section IV presents the analysis of the case study using various techniques and Section V concludes the paper.

II. COCOMO I & COCOMO II

Several software cost estimation techniques and models has been introduced to predict the effort, COCOMO is the most famous in all of them. It has been introduced by Barry Boehm in 1981. COCOMO is composed of three sub models; basic, intermediate and advanced. Size is the main input of all these models. COCOMO II is the revised form of COCOMO I. It has been introduced in 2000s and accommodates the various new software development life cycle models. It consists of three sub models. Application composition model, early design model and post architectural model. The main inputs of these models are size and cost drivers which may affect development of the projects [3]. The 17 cost drivers in these models are shown as ratings scale from extra high to very low. These ratings then converted into numerical values and use in formula to predict the effort in person per month. These cost drivers are divided into 4 categories or project, product, personnel and platform.

To take the size of the project, line of code (LOC), source line of code (SLOC), kilo line of code (KLOC), function points (FP), use case points and object points are some examples of metrics used in COCOMO II. There are other factors like scale factors which is used in post architectural model. These factors have some special characteristics and influence on the project that is why their values are calculated as exponent of the size as their increasing and decreasing have significant effect on the project's effort [3].

The main benefits of COCOMO are

- This provide a basic estimate to project managers for negotiations with customer in order to save themselves from budget overrun and infeasible deadlines.
- This gives a basic work break down structure of different concurrent or independent activities of the project.
- It provides a mechanism of project control and monitoring to know every phase of the project and its progress which in turn improves the performance of the project this will improve the process management gradually [2].

COCOMO II has undergone several improvements, modifications and calibrations. Many researchers have calibrated different cost factors to improve the model. Some proposed new cost drivers like task management and other management related factors, with the claim that poor management will lead to increase in development effort [7]-[11].

With claim that improvement in one cost driver like process maturity may lead from 4% to 11% predictability of COCOMO II introduce process maturity as main cost factor [8] and calibrating the values of constants for embedded system software cost estimation [9].

Some researchers tried to enhance the predictability of software cost estimation through artificial intelligence techniques, like neural networks, fuzzy logic with multi agent system [13] and combination techniques like neuro-fuzzy logic system [10] and calibrating the cost drivers by using evolutionary algorithms like BAT algorithm [5]-[6].

Some researchers suggest that mood, anger and socialization factor must be included in COCOMO II to increase the predictability of the model [13].

To get the accurate software cost estimation it is necessary to know which cost drivers effecting most to the projects. As the effect of these cost drivers is direct on the estimation therefore, their understanding is necessary to improve them [4] as discussed in (1) and (2).

$$PM = A * (size)\ b * EM \quad (1)$$

$$b = .91 * .01 * SF \quad (2)$$

A. Research hypothesis

As stated earlier, COCOMO II provides a significant support in cost estimation by using line of code as a major parameter. Cost drivers plays vital role in project estimation. Their influence on a particular project may add more time and complexity of that project. Therefore, in this paper following points will be addressed.

- Figure out cost drivers effects on projects.
- Effects of scaling factors on projects.
- COCOMO II effort and expert judgment comparisons.

III. METHODOLOGY

Many tools available in the market which incorporate full functionality of COCOMO. However we develop another tool for software cost estimation "Estimator". This tool incorporates not only all the models and functionalities of COCOMO I & COCOCMO II but it also support the SLIM model. It calculates the size through function point, use case points or object points metrics and then converts them in KLOC. Estimator asks the software development life cycle of the project first and then provides the previous estimates accordingly.

The methodology of this paper is to discuss the software cost estimation and the behavior of its cost drivers on the projects taken by undergraduate students of university level. The project's main characteristics being observed are as follows:

- The projects are either web based systems or mobile applications.

- The projects are of short duration, no project has taken time for over a year.
- Projects consume more time in implementation and testing phases.
- Projects use waterfall, incremental and agile all three approaches for software development.
- The present approach to find out software cost estimation is subjective based on Experts judgment.
- The projects size in less than 10,000 lines of code.
- Number of students can take the project are 2-4.

With the above observations following objectives can be set.

- What input factors and cost drivers have greater impact on the project and needs detail understanding of the students.
- What input factors and cost drivers have less impact and if removed can only make a trivial effect on the overall cost estimation.
- When the personnel factor are pre-determined what other factors having high leverage on the projects.
- Improving cost factor values will improve the predictability of COCOMO II.
- Web based application cost estimation in general.

For estimating software cost in an organization, post architectural model of COCOMO II is most appropriate model [9].

IV. Result and Discussion

Projects selected for testing purposes, are from undergraduate projects taken by students in their final year of study. There are total 50 projects tested of the graduating batch (Table I), taken from two big universities of Pakistan (COMSATS Institute of Information Technology and University of Gujrat). The result shows (Table I) LOC of projects at average are 10K. However there are some exceptions where LOC can be 21 and 25K. This shows most of students at undergraduate level must prepare themselves well for programming and programming courses must be taught and practiced in detail. In (Table I) function points are mainly between the ranges of 100-300; this shows the amount of functionality each project has to deliver. Therefore the analysis and requirement elicitation phase of the project must be started before the last year of their studies to work more on requirement collection and refinements. The cost drivers used to adjust function points are 14. However most influenced cost drivers are data communication, performance, complex processing, online updates, operational ease, installation ease, transaction, distributed data processing, heavily used configuration, reusability, end user efficiency, multiple sites deployment and facilitate changes are least affected to the projects. The main reason behind this is most of the projects are mobile apps, few are web projects in this category of projects data processing, transaction rate and performance is important concerns. Moreover some projects used artificial intelligence methods and statistical analysis techniques, which need complex processing. As these projects are taken on low level therefore heavily used configuration and multiple sites deployment are least effected issues. Scale factors in COCOMO II post architectural model are five. In which team cohesion and architecture/risk resolution effects most to projects and precedentedness and process maturity are least concerned issues at this undergraduate level. Cost drivers specific to post architectural model are 15. In which personnel attribute analyst capability, applications experience, programmer capability, virtual machine experience and language experience are kept constant for all the projects which are rated as very low. As at this level students are not expected to have any kind of professional experience regarding applications and languages. Some students are working first time with the domain and language and analysis capabilities are also low at this level. Therefore these attributes are kept low. Product attributes like reliability, database size and complexity are kept low to very low. The reason behind is; at this level students are not certain about nonfunctional requirements which is an issue to be addressed by educationists. Platform attributes are kept from low to nominal due to the same problem as stated in nonfunctional requirements. Project attributes, modern programming techniques, software tools and development schedule are deliberately kept low for all the projects. And the reason is that's programming practices and schedule is already fixed; which students cannot deviate in their projects. In the end, the effort needed in person per month is in the range of 4-10. However, this count included participating students, clients (where applicable), supervisor and co-supervisor, management and project evaluating committee.

These results are a bit over estimated when compared with expert judgment approach. Two experts are hired for project estimation. Both experts having almost 5 years of experience in software development in the domain of web applications and mobile apps. All the projects are presented to them and get estimated. Their estimations (Table I) are somewhere inclined with the estimator results and in some projects under estimated. The reason may be because the experts can see the projects in a professional perspective and may overestimate the student level.

TABLE I. UNDERGRADUATE PROJECT RESULT USING COCOMO II

Sr. No	Project Name	UFC	VAF	FP	KLOC	Scale Factors	Effort	Expert Judgment
1	Teenage Activity Tracker for Parents	143	0.9	166	8.8	1.2	7	4
2	Book Trading and Tutor Finder	90	0.8	131	6.9	1.2	5	4
3	Smart Home	100	0.9	141	7.5	1.1	5	3
4	Android Application Evaluator	63	0.8	95	5	1.1	3	2
5	Easy Share	82	0.8	116	6	1	4	2
6	Easy Route Finder	123	0.9	136	7.6	1.1	5	3
7	Mobile Application for Land Area Marking and Division	146	0.8	161	8.5	1.1	6	3
8	TIME@CHAT	58	0.8	99	5.2	1.2	4	3
9	My TV	151	0.9	172	9	1.1	6	3
10	Easy to search	282	0.8	288	15	1.1	8	6
11	Pi Controller	266	0.9	290	15	1.1	12	8
12	smart informer	155	0.8	166	8.8	1.1	7	3
13	XT Android File Manager	244	0.8	248	13	1.1	9	4
14	Knowledge skills and abilities	132	0.9	172	9.2	1	6	2
15	Informer	99	0.9	148	7.8	1.1	5	3
16	Hamsafar	255	0.8	260	13.7	1.1	10	7
17	Android Navigation App for Blinds	65	0.9	113	6	1.1	4	2
18	ilEARNING	447	0.9	411	21.8	1.1	16	5
19	3D Boutique Management System	193	0.9	207	8.7	1.1	6	3
20	Online Recruitment System	310	0.8	289	15	1.1	11	5
21	VEHICLE DEFENDER	442	1.0	478	25	1.1	18	6
22	My Assistant	93	0.8	109	5.8	1.1	4	1
23	CIIT Emergency Alert App	279	0.9	262	14	1.1	10	2
24	EARTHNET	324	0.9	320	17	1.1	12	6
25	fyp Repository	308	0.8	258	14	1.1	12	8
26	Punjab Patient Portal	157	0.9	130	7.2	1.2	5	3
27	Home Maintenance	153	0.8	125	6.7	1.2	5	3
28	iRide	137	0.8	106	5.6	1.1	4	3
29	Real Estate	125	0.8	97	5.2	1.2	4	3
30	Remote Access Mobile Using SMS	138	0.8	109	5.7	1.1	4	2
31	Mobile Theft Monitoring	164	0.8	136	7.2	1.1	5	2
32	Lodgment AR	187	0.8	155	8.2	1.1	6	3
33	Face Recognition using Real Time Data	126	0.8	99	5.2	1.2	4	2
34	Virtual Agent of UOG Admission	110	0.8	91	4.9	1.1	3	2
35	Apna Thekaydaar	133	0.9	113	5.9	1.1	4	2
36	Ofline File Sharing	131	0.8	110	5.8	1.1	4	2
37	IS-TeleHealth	111	0.9	96	5.2	1.2	4	2
38	Online Traffic Challan System	128	0.8	106	5.7	1.1	4	2
39	Online Buying and Selling	149	0.9	128	6.9	1.2	5	2
40	Physician Online	110	0.8	89	4.8	1.1	3	2
41	Secure and Smart Blood Transfusion	123	0.9	104	5.6	1.1	4	2
42	Outcome Based Education	121	0.9	102	5.5	1.1	4	2
43	Urdu Based ERP for SME	227	0.8	188	10	1.1	7	6
44	Biometric Voting System	123	0.9	113	6.1	1.1	4	3
45	Smart Voting System	108	0.9	92	5	1.1	3	3
46	Help on Spot	145	0.8	113	5.9	1.1	4	3
47	Smart Vegetable Garden Using IOT	132	0.8	110	5.9	1.1	4	3
48	Project Tracking System	169	0.8	141	7.6	1.1	5	3
49	Smart Child Guardian	172	0.9	158	8.5	1.1	6	3
50	Routing Protocol Design In Delay Tolerant Network	162	0.9	137	7.4	1.1	5	3

A. Open Issues

Following are the open issues.

1. COCOMO I & II both take LOC as their primary parameter. LOC estimation at the level of requirement elicitation and planning cannot accurate. Also LOC metrics does not support reusability and automated tools that generate code through specifications. At the level of design, LOC estimation can be done with the help of other metrics like design-code expansion. But LOC estimation is early phases of project life cycle is still an issue to be address.
2. Function points and object points are used instead of LOC. These points' methodologies though proved fruitful but they are highly subjective and cannot support complex systems (agent, robotics) and real time systems etc.
3. Projects used highly specialized algorithm like data mining or machine learning or any evolutionary algorithms, cannot be estimated through function points. COCOMO II though contains project attributes as effort adjustment factor but these measures are also subjective and may vary from expert to expert.
4. COCOMO I & II also doesn't use any latest technology like 3D modeling or any computer vision techniques and do not accommodate any kind of statistical analysis.

V. Conclusion

This paper has studies the final year projects of undergraduate students of universities of Pakistan. The scheduled time for these projects is usually one year from its inception till deployment. Effort estimation of these projects is conducted through the use of tool Estimator. The resultant effort and line of code measures are then compared with expert's judgments. Factors effecting most to these projects are team cohesion, risk resolution, data processing, and performance and transaction rate. Other factors which have low leverage on projects are also mentioned. Some issues which still need to be addressed are also subjectivity of measures and not supporting to latest technology requirements of the projects. Recommendations are also given under the light of these observations for which areas and courses taught at undergraduate level must be given preference

References

[1] I. Attarzadeh and S. H. Ow, "Improving estimation accuracy of the COCOMO II using an adaptive fuzzy logic model," 2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), Taipei, 2011, pp. 2458-2464.

[2] B. W. Boehm and R. Valerdi, "Achievements and Challenges in Cocomo-Based Software Resource Estimation," in IEEE Software, vol. 25, no. 5, pp. 74-83, Sept.-Oct. 2008.

[3] G. Kumar and P. K. Bhatia, "Automation of Software Cost Estimation using Neural Network Technique," in International Journal of Computer Applications 98(20):11-17, July 2014.

[4] D. Damor and B. Tanwala, "Improved Software Cost Estimation Scenario using WBS with COCOMO II – A Review," in International Institution for Technology Research and Development, vol. 1, 2015

[5] N. Gupta and K. Sharma, "Optimizing intermediate COCOMO model using BAT algorithm," 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, 2015, pp. 1649-1653

[6] S. D. Chulani, B. Clark and B. Boehm, "Calibrating the COCOMO II Post Architecture Model," in 20th International conference on Software engineering,1998

[7] M. A. Yahya, R. Ahmad and S. P. Lee, "Effects of software process maturity on COCOMO II's effort estimation from CMMI perspective," 2008 IEEE International Conference on Research, Innovation and Vision for the Future in Computing and Communication Technologies, Ho Chi Minh City, 2008, pp. 255-262.

[8] R. (Dick) Fairley, "The influence of COCOMO on software engineering education and training", Journal of Systems and Software, vol. 80, no. 8, pp. 1201-1208, 2007.

[9] R. Dillibabu and K. Krishnaiah, "Cost estimation of a software product using COCOMO II", 2000 model - a case study. International Journal of Project Management, vol. 23, no. 4, (2005), pp. 297–307

[10] X. Huang, D. Ho, J. Ren and L. Capretz, "Improving the COCOMO model using a neuro-fuzzy approach", Applied Soft Computing, vol. 7, no. 1, pp. 29-40, 2007

[11] B. Dixon, R. K. Smith, A. Parrish and J. Hale, "Enhancing the Cocomo Estimation Models," in IEEE Software, vol. 17, no. , pp. 45-49, 2000.

[12] Z. Mansor, Z. Kasirun, N. Arshad and S. Yahya, "E-cost estimation using expert judgment and COCOMO II", 2010 International Symposium on Information Technology, 2010.

[13] M. Kazemifard, A. Zaeri, N. Ghasem-Aghaee, M. Nematbakhsh and F. Mardukhi, "Fuzzy Emotional COCOMO II Software Cost Estimation (FECSCE) using Multi-Agent Systems", Applied Soft Computing, vol. 11, no. 2, pp. 2260-2270, 2011.

[14] Z. Muzaffar and M. Ahmed, "Software development effort prediction: A study on the factors impacting the accuracy of fuzzy logic systems", Information and Software Technology, vol. 52, no. 1, pp. 92-109, 2010.

[15] R. Sarno, J. Sidabutar and Sarwosri, "Improving the accuracy of COCOMO's effort estimation based on neural networks and fuzzy logic model," 2015 International Conference on Information & Communication Technology and Systems (ICTS), Surabaya, 2015, pp. 197-202.

[16] I. Attarzadeh, A. Mehranzadeh and A. Barati, "Proposing an Enhanced Artificial Neural Network Prediction Model to Improve the Accuracy in Software Effort Estimation," 2012 Fourth International Conference on Computational Intelligence, Communication Systems and Networks, Phuket, 2012, pp. 167-172.

[17] M. Madheswaran and D. Sivakumar, "Enhancement of prediction accuracy in COCOMO model for software project using neural network," Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Hefei, 2014, pp. 1-5.

[18] S. Wagner, "An Approach to Global Sensitivity Analysis: FAST on COCOMO," First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), Madrid, 2007, pp. 440-442.

[19] Jun Liu, Zheng Xu, Jianzhong Qiao and Shukuan Lin, "A defect prediction model for software based on service oriented architecture using EXPERT COCOMO," 2009 Chinese Control and Decision Conference, Guilin, 2009, pp. 2591-2594

[20] P. Musilek, W. Pedrycz, Nan Sun and G. Succi, "On the sensitivity of COCOMO II software cost estimation model," Proceedings Eighth IEEE Symposium on Software Metrics, 2002, pp. 13-20

[21] Y. S. Seo, K. A. Yoon and D. H. Bae, "Improving the Accuracy of Software Effort Estimation Based on Multiple Least Square Regression Models by Estimation Error-Based Data Partitioning," 2009 16th Asia-Pacific Software Engineering Conference, Penang, 2009, pp. 3-1