Volume 2, No. 7, July 2011
Journal of Global Research in Computer Science
ISSN-2229-371X
REVIEW ARTILCE
Available Online at www.jgrcs.info

TO DESIGN A SIMULATOR FOR PERFORMANCE COMPARISION OF COST MODELS

Manpreet Kaur
Department of Computer Science and Systems Engineering
University-CDLU, SIRSA, India
manpreet.beaty@gmail.com

Abstract: In Cost Estimation Models we estimate the cost. Software Cost Estimation is the process of predicting the amount of effort required to build a software system. In this dissertation, I will design a simulator that compare the cost models i.e COCOMO81 and COCOMO2.0 and find the which one is better in terms of cost, effort, persons per month and source lines of codes. In this work, random number generator is used for input. The output is shown in the form of graphs. All work is done in java language. The performance of cost models is a series of simulations reveal that the proposed scheme provides a better solution.

Keywords: Cocomo81**,** Cocomo2.0, Software Cost estimation, Software Effort Estimation, SLOC, Function Points.

INTRODUCTION

Software cost estimation is one of the most critical tasks in managing software projects. There is inevitable gap between the estimated costs and the actual costs derived from software projects and hence accurate cost estimates are highly desired during the early stages of development. The precision of the effort estimate is very important for software industry because both overestimates and underestimates of the software effort are harmful to software companies. If a manager's estimate is too low, then the software development team will be under considerable pressure to finish the product quickly. On the other hand, if a manager's estimate is too high, then too many resources will be committed to the project. In point of fact, estimating software development effort remains a complex problem and it is very important to investigate novel methods for improving the accuracy of such estimates. The most popular techniques used for software cost estimation is algorithmic models such as COCOMO [4, 5, 6].

Software cost estimation is the process of predicting the amount of effort required to build a software system. Over the years many have attempted to determine a priori what the cost of a developing a specific application will be. Why has it been so important? Not only is the budget on the line, but many times a manager's job or reputation as well. The make or buy decision must be made.

What cost is this that we are trying to estimate, determine or "predict"? We know that the cost of developing software, up until the point that it is accepted, is only a fraction of the total cost of the system over the typical life cycle of the product. However, for the purpose of this study, we will exclude the maintenance costs, and will speak only of the development costs up until acceptance. This position is consistent with that taken by those having done research in this field. Though, many membership functions were used in the literature [10] to represent the cost drivers, many of them are not appropriate to clear the vagueness in the cost drivers.

We will first review and discuss the most main published methods (lines of code, function points, and objects), and some basic terminology relating them, followed by a discussion of current trends, and finally the implications of these trends for software cost estimation.

LITERATURE REVIEW

Frank Freiman, while at RCA:

In the 1960s Frank Freiman developed the concept of parametric estimating, and this led to the development of PRICE model for hardware. This was the first generally available computerized estimating tool. It was extended to handle software in 1970s.

Berry W. Boehm:

First, Barry Boehm (developer of the COCOMO model, a model eventually se!ected for this study) has written a widely cited book entitled Software Engineering Economics [5] in which he provides an analysis of eight important models. This list was used to generate candidates.

The prototypical model of this type is the Constructive Cost Model (COCOMO) developed by Berry W. Boehm in the late 1970s and described in his classical book “Software Engineering Economics”. Various implementations of COCOMO continue to be widely used throughout the world. PRICE so, a software cost estimation model, was developed in the late 1970s by Robert Park.

Allan Albrecht:

The Function Points. Measurement method was developed by Allan .Albrecht at IBM and first published in 1976 [2]. Albrecht was interested in the general problem of productivity measurement in systems development and created the Function Points method as an alternative to estimating SLOC. Albrecht’s Function Points are at a more macro level than SLOC, capturing things like the number of input transaction types and the number of unique reports. He

believes Function Points offer several significant advantages over SLOC counts.

ESTIMACS:

Model developed by Howard Rubin of Hunter College and marketed by Management and Computer Services during the period when these data were being collected [1,3]. Both of these models were selected, bringing the total number of models compared to four.

In recent decade, many software effort estimation techniques have been proposed to evaluate their estimation performances. Some of these widely used techniques include the estimation by expert [7], analogy-based estimation [9],algorithmic method [11], rule induction [8], artificial neural network [12]

RELATED WORK

In this paper I will do the Comparision between cocomo81 and cocomo2.0. All the work is done in java language.Comparision between cocomo81 and cocomo2.0 is shown in different graphs. Cost and effort depends upon cost drivers values and inputs. Inputs are generated using random numbers.

In this paper inputs are taken as source lines of code, avg lines per program, programs, adaptation factor and labour per hour. It calculates the total cost, month duration, person month effort. For input we have to just click on random button. This dissertation compare the total cost, total effort and duration of cocomo81 and cocomo2.0. Comparision is depends upon the cost drivers values. We can set the cost drivers value using radio buttons.

It is important to stress that uncertainty at the input level of the COCOMO model yields uncertainty at the output [7]. This becomes obvious and, more importantly, bears a substantial significance in any practical endeavor. Cost drivers are often expressed through an unclear category which needs subjective assessment. The effort multipliers and scale factors of the COCOMO were described in natural language as very low, low, nominal, high, very high and extra high and these were represented by fixed numerical values [6]. More conventionally, the problem of software cost estimation using COCOMO relies on a single (numeric) value of cost driver of a given software project to predict the effort. But it is not an appropriate way to fix numerical number to each of these scales.

CONCLUSIONS AND FUTURE RESEARCH

A crucial issue for project managers is the accurate and reliable estimates of the required software development effort, especially in the early stages of the software development life cycle. Software effort drivers usually have properties of uncertainty and vagueness when they are measured by human judgment. Cost drivers in algorithmic software cost estimation are often expressed through linguistic assessments and they usually represent high level concepts for which a single, precise measurement scale is not available. This motivates the use of simulator to estimation inputs and their assessment procedures.

In this paper, it is projected an improved approach to estimate the software project effort by the use of simulator rather than classical intervals in the COCOMO model. In conclusion, the success of any software project relies on accurate estimations and a soft-computing technique such as simulator is a feasible choice as an estimation model for improving estimation accuracies.

REFERENCES

[1] Rubin, H.A. Macroestimation of software development parameters: The Estimacs system. In SOFTFAIR Conference on Software Deuelopment Took, Techniques and AIternatiues (Arlington, Va., July 25-28). IEEE Press, New York, 1983. pp. 109-118.

[2] IEEE Softw. 3,4 (July 19861, 70-75).Albrecht, A.J. Measuring application development productivity.

[3] Robin, H.A. Using ESTIMACS E. Management and Computer Services, Valley Forge, Pa., Mar. 1984.

[4] Boehm, B.W., Royce, W.W., Le COCOMO Ada, Genie logiciel & Systemes experts, 1989.

[5] Boehm, B.W. Software Engineering Economics. Prentice-Hall, Englewood Cliffs, N.J.. 1981.

[6] Boehm, B.W., et al., "Cost models for future software life cycle processes: COCOMO2.0," Annals of Software Engineering on Software Process and Product Measurement, Amsterdam, 1995.

[7] Jorgenson M, Sjoberg D.I.K., The impact of customer expectation on software development effort estimates. International Journal of Project Management, 22(4) :317–325.

[8] Jeffery R, Ruhe M,Wieczorek I, "Using public domain metrics to estimate software development effort," In Proceedings of the 7th International Symposium on Software Metrics, (April 04 - 06, 2001), IEEE Computer Society, Washington, DC, pp 16–27.

[9] Chiu NH, Huang SJ, "The adjusted analogy-based software effort estimation based on similarity distances,"Journal of Systems and Software, Volume 80, Issue 4, April 2007, Pages 628-640.

[10] Musflek P, Pedrycz W., Succi G., Reformat M, 2000,"Software cost estimation with fuzzy models," Applied Computing Review, Vol. 8, No. 2, pages 24-29.

[11] Kaczmarek J, Kucharski M, "Size and effort estimation for applications written in Java," Journal of Information and Software Technology, Volume 46, Issue 9, 1 July 2004, Pages 589-601.

[12] Heiat A, "Comparison of artificial neural network and regression models for estimating software development effort," Journal of Information and Software Technology, Volume 44, Issue 15, 1 December 2002, Pages 911-922.

