ISSN 2320-9984

Volume 4 Issue 3 | September 2016 ISSN: 2320-9984 (Online)

International Journal of Modern Engineering & Management Research

Website: www.ijmemr.org

Cost Estimating Model for Evaluation and Problem Solving Tactics

Dr. Sudeep Kishore Sharma
Professor
St. Aloysius Institute of Technology
Jabalpur (M.P.), [INDIA]
Email: sharma.sudeepcs@rediffmail.com

Amaresh Singh
Assistant Professor
St. Aloysius Institute of Technology
Jabalpur (M.P.), [INDIA]
Email: prempsgtech12@gmail.com

***Abstract*—*Software cost estimation is the process of predicting the effort required to develop a software system. Many estimation models have been proposed over the last 30 years. This paper provides a general overview of software cost estimation methods including the recent advances in the field. The main questions to be answered in the paper are: (I) What are the reasons for overruns of budgets and planned durations? (2) What are the prerequisites for estimating?(3) How can software development effort be estimated? (4) What can software project management expect from SCE models, how accurate are estimations which are made using these kind of models, and what are the pros and cons of cost estimation models? software, cost estimation, project control, software cost estimation model.*

Keywords:*— *Software Engineering, Estimating model, Technology, Computation.

1. INTRODUCTION

Estimation is the process of finding an estimate, or approximation, which is a value that can be used for some purpose even if input data may be incomplete, uncertain, or unstable. [1] Estimation determines how much money, effort, resources, and time it will take to build a specific system or product. Estimation is based on:

- Past Data/Past Experience
- Available Documents/Knowledge
- Assumptions
- Identified Risks

The main goal of software project cost and effort estimation is to scientifically estimate the required workload and its corresponding costs in the life cycle of software system. Software cost estimation is a complex activity that requires knowledge of a number of key attributes that affect the outcomes of software projects, both individually and in concert. The most critical problem is the lot of data is needed, which is often impossible to get in needed quantities. Figure 1 Hence, Software cost and effort estimation has become a challenge for IT industries. In this paper, several existing methods for software project effort, cost estimation are illustrated and their aspects are discussed. Also, it describes software metrics used for software project cost estimation. This paper summarizes existing literature on software project cost estimation[2]. The paper includes comment on the performance of the estimation models and description of research trends in software cost estimation.

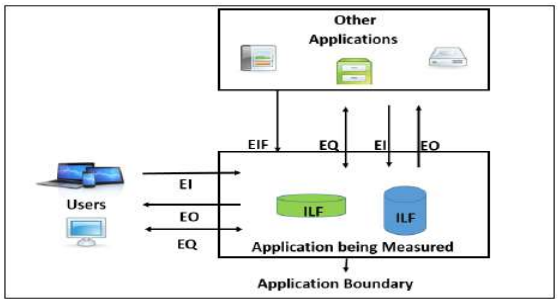

Figure 1 systematic representation of data estimation

Reasons for effort estimation vary, some of the most frequent being:

- *Project approval.* There must be a decision on project launching on the part of an organization, preceded by effort estimation required for successful completion of the project.[3]
- Project *management.* Project managers are responsible for planning and managing of the project. Both these activities require effort estimation as per respective phases in order for project to be completed.
- *Understanding by the development team members.* In order that development team could perform efficiently, it is necessary for its members to understand their individual roles as well as overall activities of the team as a whole.
- *Defining* of *project task,* that can be used for this purpose, is done by means of effort estimation.

The accuracy of effort estimation is as current an issue for resarchers today as it was 25 years ago when it was launched by Brooks (1975) in his work "The Mythical Man Month". Even today these estimations are mainly unreliable, with no proof of significant progress that has been made in their improvement, despite considerable funds and activities that have been invested to that purpose. Different authors classify effort estimation methods differently.[4]

Following classifications have been adopted for this purpose: Empirical parametric (algorithmic) estimation models; Empirical non-parametric estimation models; Expert estimation; Analogue estimation models; Downward estimation; Upward estimation".

A good software cost estimate should have the following attributes:

- It is conceived and supported by the project manager and the development team.
- It is accepted by all stakeholders as realizable.
- It is based on a well-defined software cost model with a credible basis.
- It is based on a database of relevant project experience (similar processes, similar technologies, similar environments, similar people and similar requirements).
- It is defined in enough detail so that its key risk areas are understood and the probability of success is objectively assessed. Software cost estimation historically has been a major difficulty in software development.

Several reasons for the difficulty have been identified:

- Lack of a historical database of cost measurement [5-7]
- Software development involving many interrelated factors, which affect development effort and productivity, and whose relationships are not well understood
- Lack of trained estimators and estimators with the necessary expertise
- Little penalty is often associated with a poor estimate

The four basic steps in Software Project Estimation are:

- Estimate the size of the development product.
- Estimate the effort in person-months or person-hours.
- Estimate the schedule in calendar months.
- Estimate the project cost in agreed currency.

2. SOFTWARE METRICS

This section provides some background information on software conventional metrics used in software cost and effort estimation. **A. Size Oriented Metrics:** i) Source Lines of Code (SLOC): is software metric used to measure the size of software program by counting the number of lines in the text of the program's source code. This metric does not count blank lines, comment lines, and library. SLOC measures are programming language dependent.[8] They cannot easily accommodate nonprocedural languages. SLOC also can be used to measure others, such as errors/KLOC, defects/KLOC, pages of documentation/KLOC, cost/KLOC. ii) Deliverable Source Instruction (DSI): is similar to SLOC. The difference between DSI and SLOC is that "if-then-else" statement, it would be counted as one SLOC but might be counted as several DSI. **B. Function Oriented Metrics:** Function Point (FP): FP defined by Allan Albrecht at IBM in 1979, is a unit of measurement to express the amount software functionality [9]. Function point analysis (FPA) is the method of measuring the size of software. The advantage is that it can avoid source code error when selecting different programming languages. FP is programming language independent, making ideal for applications using conventional and nonprocedural languages. It is based on data that are more likely to be known early in the evolution of project.

Function types are as:

- ***External Inputs (EI) :*** it originates from user or transmit- ted from another application. External Outputs (EO) : it is derived data within application that provides information to the user[10].
- ***E x t e r n a l Enquiries (EQ) :*** it is online i/p that results in the generation of some immediate s/w response in the form of an online output.
- ***Internal Logical Files (ILF) :*** is logical grouping of data that resides within the applications boundary and maintained via EI.
- ***External Interface Files (EIF) :*** is logical grouping of data that resides external to application but provides information that may be of use to the application.

3. ESTIMATION ACCURACY

Accuracy is an indication of how close something is to reality. Whenever you generate an estimate, everyone wants to know how close the numbers are to reality. You will want every estimate to be as accurate as possible, given the data you have at the time you generate it. And of course you don't want to present an estimate in a way that inspires a false sense of confidence in the numbers.

Important factors that affect the accuracy of estimates are[11-13]:

- The accuracy of all the estimate's input data.
- The accuracy of any estimate calculation.
- How closely the historical data or industry data used to calibrate the model matches the project you are estimating.
- The predictability of your organization's software development process.
- The stability of both the product requirements and the environment that supports the software engineering effort.
- Whether or not the actual project was carefully planned, monitored and controlled, and no major surprises occurred that caused unexpected delays.

4. COSTS OVER SHOOTS OF SOFTWAREDEVELOPMENT

Costs has become a topic of growing importance. This is not surprising. It often happens that software is more expensive than estimated and completion is later than planned. Moreover it turns out that much software does not meet the demands of the customer. There are a number of examples of such automation projects[14]. The development costs of the automation of the education funding in The Netherlands proved to be three times as much as expected. Delays and wrong payments are a daily occurrence *(Volkskrant*, 24 June 1987). The development of the software for the purpose of the house-rent subsidies, produced to government order, proved to be twice as much as planned (NRC *Handelsblad*, 28 February1989). In September 1989 the Dutch media announced as front page news the results of a governmental audit concerning the automation for the police. It proved to be an expensive disaster. The development costs of a computerized identifying system wereUS$43 million instead of the estimated US$21 million. Further more the system did not answer the formulated goals. The findings of a well-known Dutch consultancy organization (Berenschot) were that the costs of the automation of the registration of the Dutch population at the municipal offices were more than twice as much as were estimated (Volkskrant, 5 January 1990). A few years ago the estimates of the costs were about US$25million. New calculations show that there is a deficit of more than US$30 million.

5. SOFTWARE SIZING

The software size is the most important factor that affects the software cost. This sectiondescribes five software size metrics used in practice. The line of code and function point are themost popular metrics among the five metrics[15].

Line of Code:

This is the number of lines of the *delivered source code* of the software, excluding comments and blank lines and is commonly known as *LOC* [10]. Although LOC is programming language dependent, it is the most widely used software size metric [16]. Most models relate this measurement to the software cost. However, exact LOC can only be obtained after the project has completed. Estimating the code size of a program before it is actually built is almost as hard as estimating the cost of the program. A typical method for estimating the code size is to use experts' judgement together with a technique called *PERT*[17]. It involves experts' judgment of three possible code-sizes: Sl, the lowest possible size; Sh the highest possible size; and Sm, the most likely size. The estimate of the code-size S is computed as

$$S = \frac{S_l + S_h + 4S_m}{6}$$

Following are some guidelines for achieving reliable estimates:

- Base estimates on similar projects that have already been completed.
- Use relatively simple decomposition techniques to generate project cost and effort estimates.
- Use one or more empirical estimation models for software cost and effort estimation.

6. CONCLUSION

This paper focus on the existing software estimation methods. Also, presented background information on software project models and software metrics to be used for effort and cost estimation. No model can estimate the cost of software with high degree of accuracy. Estimation is a complex activity that requires knowledge of a number of key attributes. At the initial stage of a project, there is high uncertainty about these project attributes. In the past four decades a great number of different models and effort estimation methods have been developed. This clearly indicates the awareness among the researchers of the need to improve effort estimation in software engineering. Unfortunately, the fact remains that even though, all the effort invested by the researchers yielded no result as they wished for and, even today, effort estimation still 422 remains rather unreliable. Whenever estimations are made one actually looks to the future, As we learn that BBNs are especially useful when the information about the past and/or the current situation is vague, incomplete, conflicting, and uncertain. Conventional estimation techniques focus only on the actual development effort furthermore, this paper also described test effort estimation. In fact, testing activities make up 40% total software development effort. Hence, test effort estimation is crucial part of estimation process.

REFERENCES:

[1] Heemstra, F J, Kusters, R and van Genuchten, M 'Selections of software cost estimation models' Report TUE/ BDK University of Technology Eindhoven (1989)

[2] Chen Qingzhang, Fang Shuojin, Wang Wenfu, ”Development of the Decision Support System for Software Project Cost Estimation”, World Congress on Software Engineering, IEEE, 2009.

[3] Y. F. Li, M. Xie, T. N. Goh, ”A Study of Genetic Algorithm for Project Selection for Analogy Based Software Cost Estimation, IEEE, 2007.

[4] Yinhuan Zheng, Yilong Zheng, Beizhan Wang, Liang Shi, “Estimation of software projects effort based on function point”, 4th International Conference on Computer Science and Education, 2009.

[5] Pichai Jodpimai, Peraphon Sophatsathit, and ChidchanokLursin-sap, “Analysis of Effort Estimation based on Software Project Models”, IEEE, 2009.

[6] Noth, T and Kretzsclunar, M Estimation of software development projects Springer-Verlag (1984) (In German)

[7] Hecmstra, F J How expensive is software? Estimation and control of software-development Kluwer (1989) (In Dutch)

[8] Druffel, L E 'Strategies for a DoD Software initiative' CSS DUSD(RAT) Washington, DC (1982)

[9] Conte, S D, Dunsmore, H F and Shen, V Y Software engineering metrics and models Benjamin Cummins (1986) 10 Reifer, D J 'The economics of software reuse' Proc. 14th Annual ISPA Conf., New Orleans (May 1991)

[10] Ying Wang, Michael Smith, “Release Date Prediction for Telecommunication Software Using Bayesian Belief Networks”, E Canadian Conference on Electrical and Computer Engineer- ing, IEEE, 2002.

[11] Khaled Hamdan, Hazem El Khatib, Khaled Shuaib, “Practical Software Project Total Cost Estimation Methods”, MCIT 10, IEEE, 2010.

[12] JairusHihn, Hamid Habib-agahi, “Cost Estimation of Software Intensive Projects:A Survey of Current Practices”, IEEE, 2011.

[13] Khaled Hamdan, Mohamed Madi, “Software Project Effort: Different Methods of Estimation”, International Conference on Communications and Information Technology (ICCIT), Aqaba., IEEE,2011.

[14] S. Bibi, I. Stamelos, L. Angelis,

"Bayesian Belief Networks as a Software Productivity Estimation Tool", IEEE.

[15] Matthias Kerstner, "Software Test Effort estimation Methods", 2 February 2011.

[16] Nancy Merlo Schett, "Seminar on software cost estimation", University of Zurich, Switzerland, 2003.

[17] Chetan Nagar, "Software efforts estimation using Use Case Point approach by increasing Technical Complexity and Experience Factors, International Journal on Computer Science and Engineering (IJCSE) Vol. 3 No. 10 October 2011.