International Journal of Research and Development in Applied Science and Engineering (IJRDASE)
ISSN: 2454-6844

Software Cost Estimation Techniques – A Review of Literature

Dheeraj Kapoor
Computer Science & Engineering,
A.I.E.T., Lucknow, India
dheeraj.kapoor04@gmail.com

R. K. Gupta
Computer Science & Engineering,
A.I.E.T., Lucknow, India
kailashcs09@gmail.com

***Abstract*—Software Cost Estimation is process of predicting the effort required to develop a software system. Cost estimation will remain complex problem and researches should indulge to approach new technique for this task. The accuracy of the software project cost estimation has a direct and significant impact on the quality of the firm's software investment decisions. Accurate cost estimation can reduce the unnecessary costs and increase the organization's efficiency. For this reason, many estimation models have been proposed over the last 20 years. This paper focuses on the brief literature review of the various cost estimation techniques including latest trends in this field. This is an area that has been attracting a great deal of attention from researchers in the field.**

***Keywords*—Software Cost Estimation, Estimation models, Effort**

1. Introduction

Software Cost Estimation is process of predicting the effort required to develop a software system [1]. The major categories of model are algorithmic and non-algorithmic having its own strength and weakness. Selection is a key factor of accuracy. The estimates depend on effort, project duration, cost etc. Cost estimation will remain complex problem and researches should indulge to approach new technique for this task. Models based on Artificial Intelligence techniques should be used for more accurate estimation. Thus software cost estimation or software effort estimation is the process of predicting the effort required to develop a software system. Software engineering cost models and estimation techniques are used for a number of purposes including; budgeting, trade off and risk analysis, project planning and authority, and software improvement investment analysis. The accuracy of the software project cost estimation has a direct and significant impact on the quality of the firm's software investment decisions. Accurate cost estimation can reduce the unnecessary costs and increase the organization's efficiency. For this reason, many estimation models have been proposed over the last 20 years. Unfortunately, despite the large body of experience with estimation models, the perfection of these models is still far from being satisfactory. Software development effort estimation with the aid of artificial neural networks (ANN) attracted considerable research interest especially at the beginning of the nineties. A key factor in selecting a cost estimation model is the accuracy of its metrics, since these models rely on metrics as their input.

The main contribution of this paper is to provide brief literature survey of the use of various cost estimation techniques, along with the latest trends in the use of Artificial Intelligence techniques for Software Development Cost Estimation.

The paper is organized as follows: a review of various techniques for project cost estimation is presented in section 2 and section 3 summarizes existing literature on software project cost estimation. This includes comment on the performance of the estimation models and description of research trends in software cost estimation. The paper closes in Section 4, with conclusion and future research directions.

2. Literature Review of various Modeling Techniques

It has been surveyed that nearly one-third projects overrun their budget and late delivered and two-thirds of all major projects substantially overrun their original estimates. The accurate prediction of software development costs is a critical issue to make the good management decisions and accurately determining how much effort and time a project required for both project managers as well as system analysts and developers. After 20 years research, there are many software cost estimation methods available including algorithmic methods, estimating by analogy, expert judgment method, price to win method, top-down method, and bottom-up method. No one method is necessarily better or worse than the other, in fact, their strengths and weaknesses are often complimentary to each other. To understand their strengths and weaknesses is very important when you want to estimate your projects.

2.1 Algorithmic methods

The algorithmic method is designed to provide some mathematical equations to perform software estimation. These mathematical equations are based on research and historical data and use inputs such as Source Lines of Code (SLOC), number of functions to perform, and other cost drivers such as language, design methodology, skill-levels, risk assessments, etc. The algorithmic methods have been largely studied and there are a lot of models have been developed, such as COCOMO models [4], Putnam model [5], and function points based models[10].

2.1.1 Expert Judgment Method

Expert judgment techniques involve consulting with software cost estimation expert or a group of the experts to use their experience and understanding of the proposed project to arrive at an estimate of its cost. Generally speaking, a group consensus technique, Delphi technique, is the best way to be used. The strengths and weaknesses are complementary to the strengths and weaknesses of algorithmic method.

2.1.2 Estimating by Analogy

Estimating by analogy means comparing the proposed project to previously completed similar project where the project development information id known. Actual data from the completed projects are extrapolated to estimate the proposed project. This method can be used either at system-level or at the component-level.

2.1.3 Top-Down and Bottom-Up Methods

2.1.3.1 Top Down Method

Top-down estimating method is also called Macro Model. Using top-down estimating method, an overall cost estimation for the project is derived from the global properties of the software project, and then the project is partitioned into various low-level components. The leading method using this approach is Putnam model. This method is more applicable to early cost estimation when only global properties are known. In the early phase of the software development, It is very useful because there are no detailed information available.

2.1.3.2 Bottom-up Estimating Method

Using bottom-up estimating method, the cost of each software components is estimated and then combine the results to arrive at an estimated cost of overall project. It aims at constructing the estimate of a system from the knowledge accumulated about the small software components and their interactions. The leading method using this approach is COCOMO's detailed model.

2.1.4 COCOMO Models

One very widely used algorithmic software cost model is the Constructive Cost Model (COCOMO). Estimates from the basic COCOMO model can be made more accurate by taking into account other factors concerning the required characteristics of the software to be developed, the qualification and experience of the development team, and the software development environment. COCOMO model is a regression model. It is based on the analysis of 63 selected projects. The primary input is KDSI.

2.1.5 Putnam model

Another popular software cost model is the Putnam model. The form of this model is:
Technical constant C= size * B1/3 * T4/3
Total Person Months B=1/T4 *(size/C)3
T= Required Development Time in years
Size is estimated in LOC.
The Putnam model is very sensitive to the development time: decreasing the development time can greatly increase the person-months needed for development.

2.1.6 Function Point Analysis Based Methods

The Function Point Analysis is another method of quantifying the size and complexity of a software system in terms of the functions that the systems delivers to the user. A number of proprietary models for cost estimation have adopted a function point type of approach, such as ESTIMACS and SPQR/20.

2.2 Non- Algorithmic methods

2.2.1 Neural Networks

Neural networks are nets of processing elements that are able to learn the mapping existent between input and output data. The neuron computes a weighted sum of its inputs and generates an output if the sum exceeds a certain threshold. This output then becomes an excitatory (positive) or inhibitory (negative) input to other neurons in the network. The process continues until one or more outputs are generated [18]. Neural network is known of its ability in tackling classification problem.

2.2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a computational method that optimizes a problem
by iteratively trying to improve a candidate solution with regard to a given measure of quality. Such methods are commonly known as Meta Heuristics as they make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions.

2.2.3 Genetic Programming

Genetic programming is one of the evolutionary methods for effort estimation. Evolutionary computation techniques are characterized by the fact that the solution is achieved by means of a cycle of generations of candidate solutions that are pruned by the criteria 'survival of the fittest' [24]. When GA is used for the resolution of real-world problems, a population comprised of a random set of individuals is generated. The population is evaluated during the evolution process. For each individual a rating is given, reflecting the degree of adaptation of the individual to the environment. A percentage of the most adapted individuals is kept, while that the others are discarded. The individuals kept in the selection process can suffer modifications in their basic characteristics through a mechanism of reproduction. This mechanism is applied on the current population aiming to explore the search space and to find better solutions for the problem by means of crossover and mutation operators generating new individuals for the next generation. This process, called reproduction, is repeated until a satisfactory solution is found [6].

2.2.4 Fuzzy Logic

Fuzzy logic is a valuable tool, which can be used to solve highly complex problems where a mathematical model is too difficult or impossible to create. It is also used to reduce the complexity of existing solutions as well as increase the accessibility of control theory [21]. The development of software has always been characterized by parameters that possess certain level of fuzziness. Study showed that fuzzy logic model has a place in software effort estimation [16]. The application of fuzzy logic is able to overcome some of the problems which are inherent in existing effort estimation techniques [7]. Fuzzy logic is not only useful for effort prediction, but that it is essential in order to improve the quality of current estimating models [22].
Fuzzy logic enables linguistic representation of the input and output of a model to tolerate imprecision [17]. It is particularly suitable for effort estimation as many software

attributes are measured on nominal or ordinal scale type which is a particular case of linguistic values [2].

3. Review of Existing Papers

Martin Shepperd and Chris Schofield, (1997), described a technique based upon the use of analogy sometimes referred to as case based reasoning. They compared the use of analogy with prediction models based upon stepwise regression analysis for nine datasets, a total of 275 projects. A striking pattern emerges in that estimation by analogy produces a superior predictive performance in all cases when measured by MMRE and in seven out of nine cases for the Pred(25) indicator. Moreover, estimation by analogy is able to operate in circumstances where it is not possible to generate an algorithmic model, such as the dataset Real-time where all the data was categorical in nature or the Mermaid N dataset where no statistically significant relationships could be found.

Hasan Al-Sakran, (2006) proposed a new hybrid software cost estimation model, which integrates case-based reasoning and multi-agent technology. The study described the application of cased-based reasoning to estimating the cost for developing software project using multi-organization databases integrated with mobile agent technology. The proposed system may be used to produce estimates for new projects by software organizations that do not have historical projects cost data or just starting up their software business.

Stein Grimstad, et. al, (2006) took up a real-world example where common analysis of estimation error leads to a flawed conclusion, together with a review of published estimation error analysis in research studies. It was suggested that there is a need for better analyses of software cost estimation error. It was concluded that the checklist to identify non-studied factors with a potential biasing impact on the measured estimation error, the emphasis on proper estimation terminology, and the support on isolation strategies are useful. The framework does, however, not replace good analysis skill.

Vahid Khatibi, Dayang N. A. Jawawi, (2011) Here most of the present estimation techniques have been illustrated systematically. It was found that there is no estimation method which can be present the best estimates in all various situations and each technique can be suitable in the special project. It is necessary understanding the principals of each estimation method to choose the best. 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).

Zulkefli Mansor, Saadiah Yahya, Noor Habibah Hj Arshad , (2011) This paper aimed to discuss success factors that influence in traditional and agile cost estimation process for software development project. Further, this paper presented the success factors that bring to the successful of traditional and agile cost estimation in software development project. By considering these factors, it was found that agile cost estimation process can produce more accurate result and can reduce effort, time and cost.

Narendra Sharma, Ratnesh Litoriya, (2012) The aim of this research work is to identify the important cost drivers in the past project data with the help of data mining tool weka. These results suggest that building data mining and machine learning techniques into existing software estimation techniques such as COCOMO can effectively improve the performance of a proven method. While the best combination of data mining techniques were not consistent across the different stratifications of data, it showed that there are different populations of software projects and that rigorous data collection should be continued for improving the development of accurate cost estimation models.

Jyoti G. Borade, (2013) This paper focussed 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. It was concluded that conventional estimation techniques focuses only on the actual development effort furthermore whereas this paper 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.

K. Subba Rao, et. al. (2013, In this paper, an enhanced model of Holdback through neural network was constructed. The model takes the advantage of Holdback for effort estimation and neural network for learning capability. Here two most popular approaches were suggested to predict the software effort estimation. One is the K-fold cross validation method which has been already proven and successfully applied in the software effort estimation field and the other is Holdback cross validation algorithm in neural network that has been extensively used in lieu of cost estimation and have demonstrated its strength in predicting problem. It was found that the use of artificial neural network algorithm for the proposed model and the cost estimation algorithms are efficient to find the values of project estimates.

Soumyabrata Mukherjee, Bishnubrata Bhattacharya, Suvajit Mandal, (2013) Many models and metrics that have been proposed over the last 30 years has been summerized in this paper. Also it provides an overview of software cost estimation tools which is essential for estimating.

Geetika Batra, Kuntal Barua, (2013) This paper highlights general overview of cost estimation different techniques and metrics including latest trends in this field. Identification of problems and solutions to those problems is motive of this review. The review also shows that many reviewers and researchers state that assessment of cost gradually increases or decreases. Though it is an essential task the ignorance is not acceptable. Awareness of project managers and selection of methods are responsible for over budget. Each estimation techniques like COCOMO, SLIM model have it own prospects to be good and at the same time suffered with pitfalls. Estimation by algorithmic, non algorithmic, top-bottom approach or bottom-up approach etc shows their own

significance in different manner in the field of software cost estimation. The suggestions of using combination of different techniques and models can be much more efficient as alone model and method are not much effective in estimation.

Isa Maleki, Laya Ebrahimi, Saman Jodati, Iraj Ramesh,(2014) In this paper, SCE was investigated using the FL and the capabilities of COCOMO model. Therefore, this paper analyzes the FL in SCE using membership functions of Triangular, Trapezoidal, Gaussian, Generalized Bell and Sigmoidal. Functions tested and evaluated on a set of software projects dataset and showed that they have better performance than COCOMO model better.

Swati Waghmode, Dr.Kishor Kolhe, (2014) In this model, a value shrinking technique based on multilayer feed-forward neural network, and auto-associative clustering has been proposed. The Kernel component analysis is log-linear regression functions calibrated with large data set with ordinary least squares. It showed that Kernel component analysis can improve the estimation model accuracy by shrinking the input variables into an equivalent pattern and removing irrelevant variable, based on the COCOMOII data set. It also showed that the models obtained by applying Kernel component analysis are more persistent, acceptable and dependable.

Lalit V. Patil, et. Al., (2014) Here a hybrid approach has been proposed, which consists of Functional Link Artificial Neural Network (FLANN) and COCOMO-II with training algorithm. FLANN reduces the computational complexity in multilayer neural network. It does not have any hidden layer, and it has fast learning ability.

N. Veeranjaneyulu, S.Suresh, Sk.Salamuddin and Hye-jin Kim, (2014) A new classical methodology has been proposed in this paper to estimate the normal software project effort. This methodology is focused around thinking by relationship, fuzzy logic and phonetic quantifiers, which can be viably utilized when the software activities are portrayed by all out as well as numerical information. The new approach enhances the traditional relationship technique while utilizing the clear cut data. From the execution of the results, it is seen that the proposed strategy has successfully evaluated the normal effort for the software project datasets.

4. Conclusion

This paper presents a literature review of the use of various cost estimation techniques and survey of journals. This literature review is very useful, since it brings a better understanding of the field of study, and this is an important contribution of this paper. From the literature review it can be concluded that this subject attracts a great deal of interest by researchers. Researchers have developed different models for estimation but there is no estimation method which can present the best estimates in all various situations and each technique can be suitable in the special project. There are many software cost estimation methods available including algorithmic methods, estimating by analogy, expert judgment method, top-down method, and bottom-up method. No one method is necessarily better or worse than the other, in fact, their strengths and weaknesses are often complimentary to each other. In an absolute sense, none of the models perform particularly well at estimating software development effort, particularly along the RMSE dimension. Future research directions include the study of ways to select the best metrics for cost estimation, with special reference to computational intelligence techniques. The existence of features with different frequencies is a concern, and methods that will help how to envisage this problem will be made use of for future research work.

References:

[1]. N. Veeranjaneyulu, S.Suresh, Sk.Salamuddin3 and Hye-jin Kim, (2014), " Software Cost Estimation on e-Learning Technique using A Classical Fuzzy Approach", International Journal of Software Engineering and Its Applications Vol. 8, No. 11 (2014), pp. 217-222.

[2]. Stein Grimstad, et. al, (2006), "A Framework for the Analysis of Software Cost Estimation Accuracy", *ISESE'06*, ACM 1-59593-218-6/06/0009.

[3]. Lalit V. Patil, et. Al., (2014), " Develop Efficient Technique of Cost Estimation Model for Software Applications", *International Journal of Computer Applications (0975 – 8887) Volume 87 – No.16, February 2014.*

[4]. Jyoti G. Borade, (2013), " Software Project Effort and Cost Estimation Techniques", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 8, pp 730-739.

[5]. K. Subba Rao, et. al. (2013), "Software Cost Estimation in Multilayer Feed forward Network using Random Holdback Method", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 10, pp 1309-1328.

[6]. Vahid Khatibi, Dayang N. A. Jawawi, (2011), " Software Cost Estimation Methods: A Review", Journal of Emerging Trends in Computing and Information Sciences, Volume 2 No. 1, pp 21-29.

[7]. Swati Waghmode1, Dr.Kishor Kolhe, (2014), "A Novel Way of Cost Estimation in Software Project Development Based on Clustering Techniques", International Journal of Innovative Research in Computer and Communication Engineering, Vol. 2, Issue 4, PP. 3892-3899.

[8]. Isa Maleki, Laya Ebrahimi, Saman Jodati, Iraj Ramesh,(2014), "Analysis Of Software Cost Estimation Using Fuzzy Logic", International Journal in Foundations of Computer Science & Technology (IJFCST), Vol.4, No.3,, PP. 27-41.

[9]. Hasan Al-Sakran, (2006), "Software Cost Estimation Model Based on Integration of Multi-agent and Case-Based Reasoning", Journal of Computer Science 2 (3): 276-282.

[10]. Soumyabrata Mukherjee, Bishnubrata Bhattacharya, Suvajit Mandal, (2013), " A Survey On Metrics, Models & Tools Of Software Cost Estimation", International Journal of Advanced Research in Computer Engineering &Technology (IJARCET) Volume 2, Issue 9,, PP. 2620-2625.

[11]. Geetika Batra, Kuntal Barua, (2013), "A Review on Cost and Effort Estimation Approach for Software Development", International Journal of Engineering and Innovative Technology (IJEIT) Volume 3, Issue 4,, PP. 290-293.

[12]. Narendra Sharma, Ratnesh Litoriya, (2012), " Incorporating Data Mining Techniques on Software Cost Estimation: Validation and Improvement", International Journal of Emerging Technology and Advanced Engineering, Volume 2, Issue 3,, pp. 301-309.

[13]. Zulkefli Mansor, Saadiah Yahya, Noor Habibah Hj Arshad , (2011), " Review on Traditional and Agile Cost Estimation Success Factor in Software Development Project", International Journal on New Computer Architectures and Their Applications (IJNCAA) 1(3): 942-952.

[14]. Martin Shepperd and Chris Schofield, (1997), "Estimating Software Project Effort Using Analogies", IEEE transactions on software engineering, vol. 23, no. 12,, pp. 736-743.