Volume 6, Issue 1, January 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Comparative Study of Software Cost Estimation Techniques

Rekha Tripathi, Dr. P. K. Rai
Study Centre for Computer Application
A.P.S.University, Rewa [M.P], India

Abstract: Effective software cost estimation is the most challenging and important activities in software development. Developers want a simple and accurate method of efforts estimation. Estimation of cost before starting of work is a prediction and prediction always not accurate. This paper provides a detail overview of existing software cost estimation models and techniques. It's also includes the recent machine learning techniques for cost estimation field. This paper presents the strength and weakness of various cost estimation methods. This paper focuses on some of the relevant reasons that cause inaccurate estimation.

Keywords: KDLOC - Kilo of delivered lines of code, PM - Person Months, COCOMO - Constructive Cost Estimation FPA – Function point analysis, SDLC - software development life cycle, FC – Function counts, LOC -Line of code

I. INTRODUCTION

Software cost estimation play an important role in software engineering, often determining the success or failure of contract negotiation and project execution. The main goal of software cost and effort estimation is to scientifically estimate the required workload and its corresponding costs in the life cycle of software system. Accurate cost estimates activity is critical to developers and customers.

Accurate cost estimation is important for following reasons [11][15]:

- It can be used to classify and prioritize development projects with respect to complete business plan.
- It can help to find out what resources to commit to the project and how well these resources will be used
- It can help to assess the impact of changes and supporting for preplanning. Projects can be easier to manage and control when resources are matched to real needs.
- Customers expect accurate development costs to be in line with estimated costs.

Software cost estimation activity historically has been a major difficulty in software development. Several reasons have been identified that affects the cost estimation process such as: [15]

- Cost of software development estimate is difficult. The first steps in estimate are to understand and define the system to be estimated.
- A cost estimate done early in the project life cycle is generally based on less precise inputs and less detailed design specifications.
- Software development involve many inter related factors, which affect development effort and productivity, and whose relationships are not well understood.
- Incomplete, inaccurate or inconsistent historical database of cost measurement.
- Lack of trained estimators.
- Software is intangible, invisible, and intractable so it is more difficult to understand and estimate a product or process that cannot be seen and touched.

II. ALGORITHMIC METHODS

These methods are designed to provide some mathematical equations to perform software cost estimation. These mathematical equations are based on research and historical data and use some inputs for example Source Lines of Code, number of functions to perform, and some cost drivers like as language, design methodology, skill-levels, risk assessments, etc. Algorithmic methods developed many models such as COCOMO models, Putnam model, and function points based models [11] [14].

A. COCOMO Model

One very widely used algorithmic cost estimation model is the Constructive Cost Model (COCOMO) which was proposed by Boehm [4]. The basic COCOMO model has a simple form:

$MAN\text{-}MONTHS = K_1 * (KDLOC)^{K_2}$

Where K_1 and K_2 are two parameters which are dependent on the application and development environment. 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. Complexity of the software has following factor:

- Reliability
- Data base size
- Required efficiency for memory and execution time
- Capability of analyst and programmer
- Team experience in the application area
- Experience of team in the programming language and computer
- Use of software engineering and tools

This model is a regression model. It is depend on the analysis of 63 selected projects. The primary input is thousand delivered source instruction. The problems are:

1. In early phase of SDLC, the size is estimated with great uncertainty value. So, the accurate cost estimate cannot be arrived at.
2. The estimation equation is derived from the analysis of 63 selected projects. It usually has some problems outside of its particular environment for this reason, the recalibration is necessary.

The first version of COCOMO model has been experiencing increasing difficulties in cost estimating of software developed to new life cycle processes and capabilities including rapid-development process model, reuse-driven approaches, object-oriented approaches and software process maturity initiative.
For solving this problem, the newest version, COCOMO 2.0, was developed. The major capabilities of COCOMO 2.0 are a tailor able family of software size models, involving object points, function points and source lines of code; nonlinear models for software reuse and reengineering . COCOMO model is also serving as a framework for an extensive current data collection and analysis effort to further refine and calibrate the model's cost estimation capabilities.

B. Putnam Model

The Putnam model is an empirical effort estimation model. Putnam used his productivity levels observations to derive the software equation:
Technical constant C= size * $B^{1/3}$ * $T^{4/3}$
Total PM $B=1/T^4$ *$(size/C)^3$
T = Required Development Time in years
Size = estimated in LOC
Where: C = parameter dependent on the development environment and is determined on the basis of historical data of the past projects.
Rating for C=2,000 is poor C=8000 is good C=12,000 is excellent.
This model is very sensitive to the development time, decreasing the development time can greatly increase the person-months needed for development [6][12]. One significant problem with this model is that it is based on knowing, or being able to estimate accurately, the size of the software to be developed. There is often great uncertainty in the software size. It may result in the inaccuracy of estimation.

C. Function Point Analysis

The Function Point Analysis is method of quantifying the size and complexity of a software system in terms of the functions that the systems deliver to the user. A number of proprietary models for cost estimation have adapted to this type of approach, like as ESTIMACS and SPQR/20. This is a measurement which is based on the functionality of the program. It was first introduced by Albrecht [1]. The total number of FP depends on the counts of distinct in terms of format or processing logic types. Following two steps in counting function points:
1) Counting to the user functions: The raw function counts are arrived at by considering a linear combination of five basic software components. These components are external inputs, external outputs, external inquiries, logic internal files, and external interfaces, each at one of three complexity levels: simple, average or complex. The sum of these numbers, weighted according to the complexity level, is the number of FC.
2) Adjusting for environmental processing complexity: The final function points is arrived at by multiplying function count by an adjustment factor that is determined by considering 14 aspects of processing complexity. This adjustment factor allows the function count to be modified by at most 35% or -35%.

D. Linear Models

Commonly this models have the simple structure and trace a clear equation as below:
$$\mathbf{EFFORT} = a_0 + \sum_{i=0}^{n} a_i x_i$$

Where, a_1, a_2 ,an are selected according to the information of project, only allowed values for x_i are -1, 0, +1.[16]

E. Seer-sem Models

This model has been proposed in 1980 by Galorath Inc[9] . Most parameters in seer-sem are commercial and, business projects usually use seer-sem as their main cost estimation method. Size of the software is the most important feature in seer-sem method and a parameter namely S_e is defined as effective size. S_e is computed by determining the five indicators: new size, existing size, reimp redesign and retest as below:

S_e=new size + existing Size(0.4 redesign + 0.25 reimp + 0.35 retest) After computing the S_e the estimated effort is calculated as below**:**

EFFORT= $t_d = D^{-0.2} \times (S_e / C_{te})^{0.4}$ Where D = relevant to the staffing aspects

It is determined based on the complexity degree in staffs structure. C_{te} is computed according to productivity and efficiency of the project method. It is used widely in commercial project. [7]

III. NON ALGORITHMIC METHODS

Non Algorithmic methods use some information about the previous projects which are similar to the under estimate project is required and usually cost estimation process in these methods is done according to the analysis of the previous datasets.

A. Expert Judgment Method

Expert judgment techniques involve consulting with cost estimation expert or a group of the estimation experts to use their experience and understanding of the proposed project to arrive at an estimate of its cost. It is the most usable methods for the cost estimation. Mostly companies used expert judgment method for generating the cost of the product [4] [12] [13]. This method using following estimating steps:

a. Project leader presents each expert with a specification and an estimation form.
b. The experts fill out forms anonymously.
c. Project leader calls a group meeting in which the experts discuss cost estimation issues with the project leader and each other.
d. Project leader prepares and distributes a summary of the cost estimation on an iteration form.
e. Again experts fill out forms, anonymously.
f. Steps d and step e are iterated for as many rounds as appropriate.

B. Estimating by Analogy

Cost estimating by analogy means comparing the proposed project to previously completed similar project where the project development information is known. Actual data from the completed projects are extrapolated to cost estimate the proposed project. Analogy method can be used either at system level or at the component level [12]. This method using following estimating steps:

a. Find out the necessary characteristics of the proposed project.
b. Choose the most similar completed projects whose characteristics have been stored in the historical data base.
c. Find the estimate for the proposed project from the most similar completed project by analogy.

C. Parkinson's Law

Using Parkinson's Law **"**Work expands to fill the available volume"[8], the cost is determined by the available resources rather than based on an objective assessment. [12], If the software has to be delivered in 20 months and 4 people are available, the effort is estimated to be 80 PM. Although it sometimes gives good estimation, this method is not recommended as it may provide very unrealistic estimates. Parkinson's Law does not promote good software engineering practice [15].

E. Price-to-win

The cost is estimated to be the best price to win the project. The cost estimation is based on the customer's budget instead of the software functionality. For example, if a reasonable estimation for a project costs 100 PM but the customer can only effort 60 PM. It is common that the estimator is asked to modify the estimation to fit 60 PM effort in order to win the project. This is again not a good practice since it is very likely to cause a bad delay of delivery or force the estimation team to work overtime[10][15].

F. Top-Down Estimating Method

Top-down estimating method is known as Macro Model. Using this 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 mechanism or components. The method using this approach is Putnam model. Top-Down method is more applicable to early cost estimation when only global properties are known. In the early phase of the software cost estimation, top-down is very useful because there is no detailed information available [4] [13].

G. Bottom-up Estimating Method

Using bottom-up cost estimating method, the cost of each software component is estimated and then combines the results to arrive at an estimated cost of overall project. Bottom-up method aims at constructing the estimate of a system from the knowledge accumulated about the small software components and their interactions. The method using this approach is COCOMO's detailed model [4].

IV. MACHINE LEARNING METHODS

Most techniques about cost estimation use statistical methods, which are not able to present reason and strong results. This approach could be appropriate because they can increase the accuracy of estimation by training rules of estimation and repeating the run cycles. It is categorized into two main methods, neural networks and fuzzy methods which are discussed in the next subsections.

A. Neural networks

Neural networks include several layers which each layer is composed of several elements called neuron. Neurons, by investigating the weights defined for inputs, produce the outputs. Outputs will be the actual effort, which is the main goal of estimation.

Back propagation neural network is the best selection for software estimation problem because it adjusts the weights by comparing the network outputs and actual results. In addition, training is done effectively. Majority of researches on using the neural networks for software cost estimation, are focused on modeling the Cocomo method, for example in [2] a neural network has been proposed for estimation of software cost according to the following figure. Figure (1) shows the layers, inputs and the transfer function of the mentioned neural network. Scale Factors(SF) and effort multipliers(EM) are input of the neural network, pi and qj are respectively the weight of SFs and EMs.[16]

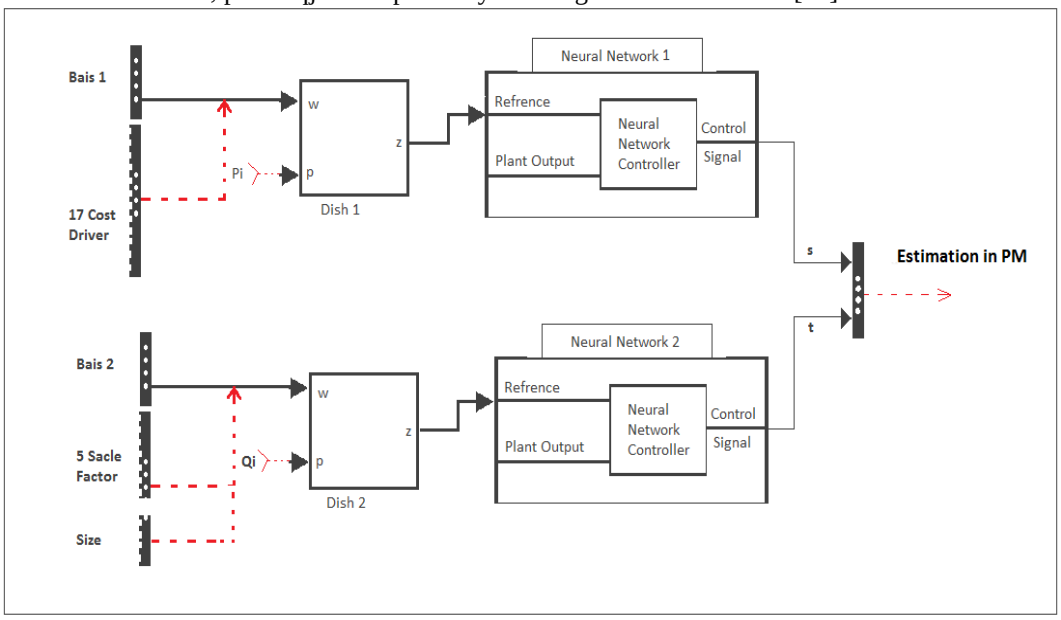

Fig.1 Layers, input & transfer function of neural network [2]

B. Fuzzy Method

The systems, which work based on the fuzzy logic try to simulate human behavior and reasoning. In many problems, where decision making is very difficult and conditions are vague, fuzzy systems are an efficient tool in such situations. Fuzzy technique always supports the facts that may be ignored. Following four stages in the fuzzy approach:

Stage 1: produce trapezoidal numbers for the linguistic terms.
Stage 2: develop the complexity matrix by producing a new linguistic term.
Stage 3: determine the productivity rate and the attempt for the new linguistic terms.
Stage 4: determine the effort required to complete a task and to compare the existing method.

For example in [3] COCOMO technique has been implemented by using fuzzy method. The Fig (2) displays all following mentioned steps.

Step (1) fuzzification has been done by scale factors, cost drivers and size.
Step (2) principals of COCOMO are considered.
Step (3) defuzzification is accomplished to find the effort [16].

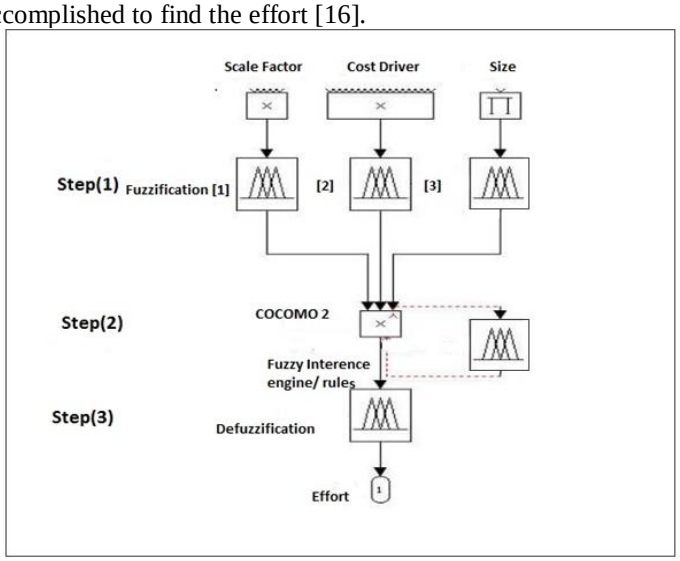

Fig.2 An example of using Fuzzy method [3]

V. COMPARISON OF VARIOUS COST ESTIMATION TECHNIQUES

Table 1 Comparison of Various Estimation Methods

S.No.	Method	Type	Strength	Weakness
1	COCOMO	Algorithmic	Clear results, it's very common	A lot of data is required, It is not suitable for any project
2	Function point	Algorithmic	Language free, its results are better than source line of code	Mechanization is hard to do , it is not considered quality output
3	Putnam model	Algorithmic	A Probabilistic model, it's used in a very large project	For only use large projects
4	Seer-Sem model	Algorithmic	Used in a very large projects	it's required 50 input parameters which are increases the complexity and uncertainty
5	Linear model	Algorithmic	It's a best method of prediction using linear regression technique	Little difference between actual and predicted results and error is also needed to calculate.
6	Expert judgment	Non-Algorithmic	Fast prediction, adapt for especial projects	Success depend on expert, usually is done incomplete
7	Analogy	Non-Algorithmic	Works based on actual experience and especial expert is not important	Much information about past projects is required in some situations there are no similar project
8	Parkinson	Non-Algorithmic	Correlates with some experience	Reinforces poor practice
9	Price to win	Non-Algorithmic	It's often gets the contract	Generally produces large over runs
10	Top - down	Non-Algorithmic	Requires minimal project detail, usually faster and easier to implement and system level focus	Less detailed basis and less stable
11	Bottom - down	Non-Algorithmic	More detailed basis, more stable and encourage individual commitment	May overlook system level costs , requires more effort, a lot of time consuming
12	Neural network	Machine learning model	Consistent with unlike databases, power of reasoning	There is no guideline for designing, performance depends on large training data
13	Fuzzy	Machine learning model	Training is not required, flexibility	Maintaining the degree of meaningfulness is difficult, hard to use

According to the above comparison, it is evident that no one method is necessarily better or worse than the other, in fact, their advantage and disadvantage are often complimentary to each other. The algorithmic methods are based on mathematics and some experimental equations. They are usually hard to learn and they need much data about the current project state. But if enough data is available, these methods present the reliable results. On the other hand for non algorithmic methods it is necessary to have the enough information about the similar type of previous projects, because these methods perform the cost estimation by analysis of the historical data. They are easy to learn because they follow the human behavior. According to the estimation experience, it is recommended that a combination of models and expert judgment estimation methods are useful to get reliable, accurate cost estimation for software development. We should use expert judgment method or analogy method for known projects and projects parts if the similarities of them can be got, since it is fast and under this circumstance, reliable. For large, lesser known projects, it is better to use algorithmic methods. In this case, many researchers recommend the estimation models that do not require source line of code as an input. If we approach cost estimation by parts, we may use expert judgment for some known parts. In this way we can take advantage of both the rigor of models and the speed of expert judgment method or analogy. Because the advantages and disadvantages of each technique are complementary, a combination will reduce the weakness of any one technique, augment their individual strengths and help to cross-check one technique against another.

VI. CONCLUSION

In this paper, we have discussed a comparative study of different types of software cost estimation techniques and also described the advantages and disadvantages of these techniques. This paper presents some of the relevant reasons that cause inaccurate estimation. To produce a meaningful and reliable cost estimate, we must improve our understanding of software project attributes and their causal relationships. It has been seen that all cost estimation methods are specific for some specific type of projects. It is very difficult to decide which method is better than to all other methods because every method or model has an own significance or importance. In recent year researchers have worked with another field along with the software engineering like data mining and machine learning techniques for improving the accuracy

of software estimation process. The future work is to study new software cost estimation technique that can help us to easily understand the software cost estimation process.

REFERENCES

[1] A.J. Albrecht and J.E. Gaffney, "Software function, source lines of code, and development effort prediction: a software science validation" IEEE Transactions on Software Engineering, , pp. 639–647, 1983.

[2] Attarzadeh,I. Siew Hock Ow, "Proposing a New Software Cost Estimation Model Based on Artificial Neural Networks",IEEE International Conference on Computer Engineering and Technology (ICCET) , Volume: 3, Page(s): V3-487 - V3-491 2010.

[3] Attarzadeh, I. Siew Hock Ow,"Improving the accuracy of software cost estimation model based on a new fuzzy logic model",World Applied science journal 8(2):117-184,2010-10-2.

[4] B.W. Boehm",Software Engineering Economics"Prentice- Hall, Englewood Cliffs, NJ, USA, 1981.

[5] COCOMO II Model definition manual, version 1.4, University of Southern California.

[6] Caper Jones" Estimating software cost " tata Mc- Graw -Hill Edition 2007.

[7] Fischman,L, K. McRitchie, and D.D. Galorath," Inside SEER-SEM", CrossTalk, The Journal of Defense Software Engineering, 2005.

[8] G.N. Parkinson" Parkinson's Law and Other Studies in Administration" Houghton-Miffin, Boston, 1957.

[9] Galorath, D. D., & Evans, M. W. " Software sizing, estimation, and risk management: When performance is measured performance improves". Boca Raton, FL: Auerbach,2006.

[10] Hareton Leung, Zhang Fan, "Software Cost Estimation", Article 2001.

[11] Leungh, Zhangf,_"Software cost estimation" in Handbook of software engineering and knowledge engineering' (World Scientific Pub. Co, River Edge, NJ, 2001)

[12] Liming Wu "The Comparison of the Software Cost Estimating Methods" University of Calgary.

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

[14] Oscar Marbán, Antonio de Amescua, Juan J. Cuadrado, Luis García "A cost model to estimate the effort of data mining projects", Universidad Carlos III de Madri (UC3M), Volume33, Issue 1, pp.133-150, March, 2008 .

[15] Sweta Kumari and Shashank Pushkar," Performance Analysis of the Software Cost Estimation Methods", International Journal of Advanced Computer Science and Applications, Vol. 3, 2013.

[16] Vahid Khatibi, Dayang N. A. Jawawi " Software Cost Estimation Methods: A Review" Journal of Emerging Trends in Computing and Information Sciences, Volume 2 No. 1 ,2010-11

