Research Paper Engineering

Review on Cost Estimation Methods for Software Development

* Ms. Shivani Rathore ** Mr. Ghanshyam Rathore
*** Mr. Anil Khandekar

* (M.E. Student), IIST, Indore

** IIST, Indore

*** (Guide), IIST, Indore

ABSTRACT

Cost Estimation is an essential part for good Engineering management. Software Engineering cost estimation provides the important link between the general concept and technical concept of economical analysis. Accurate cost estimation helps to complete project within time and budget. This paper summarizes the overview of several cost estimation models and techniques.

Keywords:

I. Introduction

The emphasis on software cost estimation has been increasing gradually over last three decades. The accurate prediction of software development cost is a critical issue, but without reasonably accurate cost estimation capability, project manager cannot determine how much time and manpower should take.

Software cost estimation is a continuing activity which starts at the proposal stage and continuous through the life time of project. The estimation process includes size estimation, effort estimation, developing initial project schedules and finally estimating overall cost of the project.

Cost estimation methods are basically of two types:-

1. Algorithmic Based

- COCOMO Model
- Putnam Model
- Function Point Analysis.

2. Non- Algorithmic Based

- Expert Judgment
- Analogy Costing
- Top Down and Bottom Up
- Parkinson's law
- Pricing to win.

The main aim of this paper is to provide a review of all these types of models and methods.

II. FEATURES

A. Non Algorithmic Based Estimation Methods

1. EXPERT JUDGMENT: - This method involves one or more experts for consulting. Expert provides estimates using their own methods and experience. Process iterates until some consensus is reached. Deliphi Technique or PERT will be used to resolve the inconsistencies in the estimates.

The estimating steps using this method-

- Manager provides specifications and estimation form to all experts.
- Group Meeting will be held between Manager and all Experts in which they discuss on estimation issues.
- Experts fill out forms anonymously.
- Manager prepares and distributes a summary of estimation on iteration form.
- Manager calls a group meeting, in which they discuss those points in which they are varied widely.
- Experts fill out forms, again anonymously, and steps 4 and 6 are repeated until as appropriate.

2. ANALOGY COSTING: - The cost of project is computed by comparing the project to a similar project in the same application domain. Actual data from completed projects are taken to estimate the proposed project. This method can be used either at system-level or component-level.

The steps using estimating the analogy are:-

- Find out the characteristics of the proposed project.
- Select the completed projects whose characteristics are mostly same as proposed project and have been stored in database.
- Find out the estimate for the proposed project from the most similar completed project by analogy.

3. TOP DOWN and BOTTOM UP methods: - In top down method, cost estimation is based on the global properties of the software project and then the project is partitioned into various low-level components. Putnam model used this approach. It is used in early phase of the software development and where no detailed information is available.

In bottom up approach, cost estimation of all components of project is calculated independently and then combined them to find out the overall cost estimation of project. COCOMO Model used this approach.

4. PARKINSON'S LAW: - Using Parkinson's Law "work expands to fill the available volume", according to available resources the cost of project is determined rather than project specifications. This method is rarely used because sometimes it provides good estimation.

5. PRICING TO WIN: - In this method cost estimation is based on costumer's budget instead of the software functionality. This is again not a good practice because it cause a bad

delay to a delivery or force the development team to work over time.

B. Algorithmic Based Estimation Methods

In this process cost are analysed using mathematical formula linking costs or inputs with metrics to produce an estimated output. The formula used in formal models arises from the analysis of historical data. There is variety of different models available, the best known are Boehm's COCOMO Model, Putman's SLIM Model and Albrecht's function points.

1. COCOMO MODEL: - Constructive Cost Model (COCOMO) is widely used algorithmic software cost model. It has following hierarchy-

- Model 1 (Basic COCOMO Model):- The basic COCOMO model computes software development effort and cost as a function of program size expressed in estimated lines of code(LOC). The basic steps in this Model are:-

a) Obtain an initial estimate of the development effort from the estimate of thousands of delivered lines of source code (KLOC).
b) Determine a set of 15 multiple factors from different attributes of the project.
c) Adjust the effort estimate by multiplying the initial estimate with all the multiplying factors.

The initial estimate (also called nominal estimate) is determined by an equation of the form used in the static single-variable models, using KLOC as the measure of size. To determine the initial effort E in person-months the equation used is of the type-

$E= a^* (KLOC)^b$

The value of constants a and b depend on the project type.

- Model 2 (Intermediate COCOMO Model):- Intermediate COCOMO Model computes software development effort as a function of program size and set of " cost drivers " that include subjective assessment of the products, hardware, personnel and project attributes.

The basic model is extended to consider a set of "cost driver attributes" that can be grouped into four major categories:

1. Product attributes
a. required software reliability
b. size of application data base
c. complexity of the product

2. Hardware attributes
a. run-time performance constraints
b. memory constraints
c. volatility of the virtual machine environment
d. required turnaround time

3. Personnel attributes
a. analyst capability
b. software engineer capability
c. applications experience
d. virtual machine experience
e. programming language experience

4. Project attributes
a. use of software tools
b. application of software engineering methods
c. required development schedule Each of the 15 attributes is rated on a 6 point scale that ranges from "very low" to "extra high" (in importance or value). Based on the rating, an effort multiplier is determined from tables published by Boehm, and the product of all effort multipliers results is an *effort adjustment factor* (EAF). Typical values for EAF range from 0.9 to 1.4.

The intermediate COCOMO model takes the form:

$E = a_i KLOC^{b_i} \times EAF$

Where *E* is the effort applied in person-months and *KLOC* is the estimated number of delivered lines of code for the project.

- Model 3 (Advanced COCOMO Model):- The advanced COCOMO Model incorporates all characteristics of the intermediate version with an assessment of the cost driver's impact on each step (analysis , design, etc) of the Software Engineering Process.

2. PUTNAM MODEL: - The Putnam model is an empirical software effort estimation model. Putnam used his observations about productivity levels to derive the software equation:

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

Where: C is a parameter dependent on the development environment and It is determined on the basis of historical data of the past projects.

Rating: C=2,000 (poor), C=8000 (good) C=12,000 (excellent).

One of the distinguishing features of the Putnam model is that total effort decreases as the time to complete the project is extended.

3. FUNCTION POINT ANALYSIS: - 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 deliver 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.

This is a measurement based on the functionality of the program and was first introduced by Albrecht. The total number of function points depends on the counts of distinct (in terms of format or processing logic) types.

There are two steps in counting function points:

a) Counting the user functions. The raw function counts are arrived at by considering a linear combination of five basic software components: 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 function counts (FC).
b) Adjusting for environmental processing complexity:- The final function points is arrived at by multiplying FC by an adjustment factor that is determined by considering 14 aspects of processing complexity. This adjustment factor allows the FC to be modified by at most 35% or -35%.

III. Conclusions

This paper has presented an overview of a variety of software estimation techniques, providing an overview of several popular estimation models currently available. To produce a meaningful and reliable estimate, the cost estimation process needs to be thoroughly arranged and carefully followed.

It seems that all estimation methods are specific for some specific type of projects. It would be best when estimating the cost that more than one estimation method be used to get a global view on the possible cost.

In recent year research, researchers worked with another field along with the software engineering like data mining for improving the accuracy of software cost estimation process.

The future work is to study new software cost estimation methods and models that can be help us to easily understand the software cost estimation process.

REFERENCES

[1] CAPER JONES. “ESTIMATING SOFTWARE COST” TATA MC- GRAW -HILL | Edition | [2] Roberto Meli, Luca Santillo “FUNCTION POINT ESTIMATION | METHODS: A COMPARATIVE OVERVIEW” Data Processing Organization. | [3]Murali Chemuturi “Analogy based Software Estimation” Chemuturi | Consultants | [4] Disaggregating and Calibrating the CASE Tool Variable in COCOMO II | Jongmoon Baik, Member, IEEE, Barry Boehm, Fellow, IEEE, and Bert M. Steece. | [5] An Integrated Approach to Software Engineering by Pankaj Jalote. | [6] Software Engineering by ROGER S. PRESSMAN, McGRAW.HILL INTERNATIONAL EDITION. | [7] Neural Network Approach for Software Cost Estimation | Nasser Tadayon | Department of Computer and Software Engineering | Embry Riddle Aeronautical University | [8] The Consistency of Empirical Comparisons of Regression and Analogy-based Software Project Cost Prediction | Carolyn Mair and Martin Shepperd | Brunel University, UK | [9] Effort Prediction in Iterative Software Development Processes – Incremental Versus Global Prediction Models.