Volume 4, No. 5, May 2013
Journal of Global Research in Computer Science
ISSN-2229-371X
REVIEW ARTICLE
Available Online at www.jgrcs.info

A SYSTEMATIC REVIEW OF COST ESTIMATION MODELS

G. Rajkumar[1], Dr.K.Alagarsamy[2]
[1]Assistant Professor, N.M.S.S.Vellaichamy Nadar College, Madurai, India.
[2]Associate Professor, Computer Centre, Madurai Kamaraj University, Madurai, India

Abstract: Software estimation models should support managerial decision making in software projects. Software cost estimation is the process of predicting the cost and effort required to develop a software system. Many estimation models have been proposed over the last twenty to thirty years. Many software companies track and analyze project performance by measuring cost estimation accuracy. A high estimation error is frequently interpreted as poor estimation skills. One of the problems in software cost estimation is how to evaluate estimation models. A key factor is the accuracy of its estimates in selecting a cost estimation model. Unfortunately, despite the large body of experience with estimation models, the accuracy of these models is not satisfactory. The paper includes comment on the performance of the estimation models and description of several approaches to cost estimation. This paper summarizes several classes of software cost estimation models and techniques.

INTRODUCTION

Software cost estimation is an essential part of most software development. Unfortunately, software development cost estimation is difficult and inaccurate. In spite of the availability of many estimation methods and guidelines, there is still a need for improvement. One means of reducing cost estimation error is through analysis of cost estimation error, e.g., through the identification of estimation processes that lead to more accurate estimates. Since the early 1950s, Software engineering cost models and estimation techniques are used for a number of purposes.

These include Budgeting, risk analysis, Project Planning and control and Software improvement and investment analysis. In recent years, software has become the most expensive component of computer system projects. Accurate software cost estimates are critical to both developers and customers. [7]Research on software cost estimation started independently in a number of companies and military organizations that built large software systems. They can be used for generating request for proposals, contract negotiations, scheduling, monitoring and control. Underestimating the costs may result in management approving proposed systems that then exceed their budgets, with underdeveloped functions and poor quality, and failure to complete on time. The purpose of software cost estimation is to:

a. [2]Define the resources needed to produce, verify, and validate the software product, and manage these activities.
b. Software Quality Assurance
c. [1]Quantify, insofar as is practical, the uncertainty and risk inherent in this estimate.
d. Test-bed development
e. Development Environment Support
f. Software system-level test support, including development and simulation software
g. Administration and Support Costs
h. Independent Verification & Validation (IV&V)
i. Other review or support charges

Accurate cost estimation is important because:

a. Help to classify and prioritize development projects with respect to an overall business plan.
b. Used to determine what resources to commit to the project and how well these resources will be used.
c. Used to assess the impact of changes and support replanning.
d. [3]Projects can be easier to manage and control when resources are better matched to real needs.

ESTIMATION METHODS

All the cost estimation methods are based upon some form of analogy: Historical Analogy, Expert Judgment, Models, etc., the role these methods play in generating an estimate depends upon where one is in the overall life-cycle.

Historical analogy estimation methods are based upon using the software size, cost or effort of a comparable project from the past. When the term analogy is used in this document, the comparison is made using measures or data that has been recorded from completed software projects. Analogical estimates can be made at high levels using total software project size and/or cost for individual Work Breakdown Structure (WBS) categories in the process of developing the main software cost estimate. Generally, it is necessary to adjust the size or cost of the historical project, as there is rarely a perfect analogy. This is especially true for high-level analogies. [10]Analogy models are the simplest type of estimating models. They are used to estimate cost by comparing one program with a similar past program or programs, thereby avoiding issues with expert judgment bias. The advantage of the analogy method is that it is based on experience. However, the method is limited because, in most instances, similar programs do not exist. [8]The steps using estimating by analogy are, Characterizing the proposed project; Selecting the most similar completed projects whose characteristics have been stored in the historical data base' Deriving the estimate for the proposed project from the most similar completed projects by analogy.

The main advantages of this method are, The estimation are based on actual project characteristic data; The estimator's past experience and knowledge can be used which is not

easy to be quantified; The differences between the completed and the proposed project can be identified and impacts estimated.

Using this method, we have to determine how best to describe projects. The choice of variables must be restricted to information that is available at the point that the prediction required. Possibilities include the type of application domain, the number of inputs, the number of distinct entities referenced, the number of screens and so forth.

Even once we have characterized the project, we have to determine the similarity and how much confidence can we place in the analogies. Too few analogies might lead to maverick projects being used; too many might lead to the dilution of the effect of the closest analogies. Finally, we have to derive an estimate for the new project by using known effort values from the analogous projects. Possibilities include means and weighted means which will give more influence to the closer analogies.

***Expert judgment*:** The [5] majority of research work carried out in the software cost estimation field has been devoted to algorithmic models. However, by an overwhelming majority, expert judgment is the most commonly used estimation method. In its crudest form the expert judgment method involves consultation with one or more local experts who are knowledgeable about the development environment or application domain to estimate the effort required to complete a software project.

The method relies heavily on the experience of their knowledge in similar development environments and historically maintained databases on completed projects and the accuracy of theses past projects. The advantages of this method are, The experts can factor in differences between past project experience and requirements of the proposed project; The experts can factor in project impacts caused by new technologies, architectures, applications and languages involved in the future project and can also factor in exceptional personnel characteristics and interactions, etc. The disadvantages include, This method cannot be quantified; It is hard to document the factors used by the experts or experts-group; Expert may be some biased, optimistic, and pessimistic, even though they have been decreased by the group consensus; The expert judgment method always compliments the other cost estimating methods such as algorithmic method.

Delphi Approach or Wideband Delphi technique attempts to gather the opinions of a group of experts with the aim of producing an accurate unbiased estimate. It is a structured technique of expert judgment and is essentially a form based technique involving a multistep procedure: Experts are issued the specification and estimation form by the coordinator.

a. A group meeting is held to discuss the product and estimation issues.
b. Experts produce an independent estimate.
c. Estimates are returned indicating the median estimate and the expert's personal estimate.
d. Another group meeting is held to discuss results.
e. Experts prepare a revised independent estimate.
f. Steps 3-6 are repeated until a consensus is reached by the panel of experts.

Advantages of the Delphi Estimation Process:

a. [12]Free of social pressure, personality influence, and individual dominance
b. Allows sharing of information and reasoning among participants
c. Conducive to independent thinking and gradual formulation
d. Respondent panel provides broad analytical perspective on problems and issues
e. Can be used to reach consensus among groups hostile towards each other

Disadvantages of the Delphi Estimation Process:

a. Judgments are those of a selected group, and may not represent prevailing opinion
b. Tendency to eliminate extreme positions and force middle-of-the-road consensus
c. More time-consuming than nominal group process
d. Requires skill in written communication
e. Requires adequate time and participant commitment (may require 30 to 45 days to complete entire process)

Algorithmic Method: 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, Putnam model, and function points based models. General advantages of methods are, it is able to generate repeatable estimations; It is easy to modify input data, refine and customize formulas; It is efficient and able to support a family of estimations or a sensitivity analysis; It is objectively calibrated to previous experience. General disadvantages are, It is unable to deal with exceptional conditions, such as exceptional personnel in any software cost estimating exercises, exceptional teamwork, and an exceptional match between skill-levels and tasks; Poor sizing inputs and inaccurate cost driver rating will result in inaccurate estimation; Some experience and factors cannot be easily quantified.

Bottom-up approach, each component of the software system is separately estimated and the results aggregated to produce an estimate for the overall system. The requirement for this approach is that an initial design must be in place that indicates how the system is decomposed into different components. The advantages of this methods are, It permits the software group to handle an estimate in an almost traditional fashion and to handle estimate components for which the group has a feel; It is more stable because the estimation errors in the various components have a chance to balance out. The disadvantages are, It may overlook many of the system-level costs (integration, configuration management, quality assurance, etc.) associated with software development; It may be inaccurate because the necessary information may not available in the early phase;

It tends to be more time-consuming; It may not be feasible when either time and personnel are limited.

Top-down approach is the opposite of the bottom-up method. An overall cost estimate for the system is derived from global properties, using either algorithmic or non-algorithmic methods. The total cost can then be split up among the various components. This approach is more suitable for cost estimation at the early stage. The advantages of this method are, It focuses on system-level activities such as integration, documentation, configuration management, etc., many of which may be ignored in other estimating methods and it will not miss the cost of system-level functions; It requires minimal project detail, and it is usually faster, easier to implement. The disadvantages are, It often does not identify difficult low-level problems that are likely to escalate costs and sometime tends to overlook low-level components; It provides no detailed basis for justifying decisions or estimates.

COCOMO (Constructive Cost Model) models[4] cost and schedule estimation model was originally published in [Boehm 1981]. It became one of most popular parametric cost estimation models of the 1980s. But COCOMO '81 along with its 1987 Ada update experienced difficulties in estimating the costs of software developed to new life-cycle processes and capabilities. The COCOMO II research effort was started in 1994 at USC to address the issues on non sequential and rapid development process models, reengineering, reuse driven approaches, object oriented approaches etc. COCOMO II was initially published in the Annals of Software Engineering in 1995 [Boehm *et al.* 1995]. The model has three sub models, Applications Composition, Early Design and Post-Architecture, which can be combined in various ways to deal with the current and likely future software practices marketplace. A primary attraction of the COCOMO models is their fully-available internal equations and parameter values. Over a dozen commercial COCOMO '81 implementations are available.

The models have been widely accepted in practice. In the COCOMOs, the code-size S is given in thousand LOC (KLOC) and Effort is in person-month. The basic COCOMO model is simple and easy to use. As many cost factors are not considered, it can only be used as a rough estimate.

COCOMO model is a regression model. It is based on the analysis of 63 selected projects. The primary input is KDSI. [11]The problems are: In early phase of system life-cycle, the size is estimated with great uncertainty value. So, the accurate cost estimate can not be arrived at; For this reason, the recalibration is necessary.

SOFTWARE ESTIMATION

Software project plans include estimates of cost, resources, product size, schedules, staffing levels, and key milestones. The software estimation process is discussed in the following steps for developing software estimates. Establishing this process early in the life-cycle will result in greater accuracy and credibility of estimates and a clearer understanding of the factors that influence software development costs. This process also provides methods for project personnel to identify and monitor cost and schedule risk factors. The estimation method described is based upon the use of:

a. Multiple estimates
b. Data-driven estimates from historical experience
c. Risk and uncertainty impacts on estimates
d. [8]Different estimation methods may use different data. This results in better coverage of the knowledge base for the estimation process. It can help to identify cost components that cannot be dealt with or were overlooked in one of the methods
e. Different viewpoints and biases can be taken into account and reconciled. A competitive contract bid, a high business priority to keep costs down, or a small market window with the resulting tight deadlines tends to have optimistic estimates. A production schedule established by the developers is usually more on the pessimistic side to avoid committing to a schedule and budget one cannot meet.

The selection of Estimation methods:

From the above comparison, we know [13]no one method is necessarily better or worse than the other, in fact, their strengths and weaknesses are often complimentary to each other. According to the experience, it is recommended that a combination of models and analogy or expert judgment estimation methods is useful to get reliable, accurate cost estimation for software development.

Table 1: Strengths and Weakness of different methods

Methods	Strengths	Weaknesses
Expert Judgment	Expert with the relevant experience can provide good estimation – Fast estimation.	Dependent on the Expert.
Analogy	Based on actual project data and past experience.	Similar projects may not exist – Historical data may not be accurate.
Top – down	Faster and easier than bottom-up method.	Less accurate than other methods.
Bottom-up	Based on detailed analysis.	Difficult to perform the estimate early in the life cycle.
Delphi	Sharing information	More time-consuming than nominal group process.

Use of Estimation Methods:

It is very common that we apply some cost estimation methods to estimate the cost of software development. But what we have to note is that it is very important to continually re-estimate cost and to compare targets against actual expenditure at each major milestone. This keeps the status of the project visible and helps to identify necessary corrections to budget and schedule as soon as they occur.

At every estimation and re-estimation point, iteration is an important tool to improve estimation quality. The estimator can use several estimation techniques and check whether their estimates converge. The other advantages are as following, Different estimation methods may use different data. This results in better coverage of the knowledge base for the estimation process. It can help to identify cost components that cannot be dealt with or were overlooked in

one of the methods; Different viewpoints and biases can be taken into account and reconciled. A competitive contract bid, a high business priority to keep costs down, or a small market window with the resulting tight deadlines tends to have optimistic estimates. A production schedule established by the developers is usually more on the pessimistic side to avoid committing to a schedule and budget one cannot meet.

It is also very important to compare actual cost and time to the estimates even if only one or two techniques are used. It will also provide the necessary feedback to improve the estimation quality in the future. Generally, the historical data base for cost estimation should be set up for future use.

Identifying the goals of the estimation process is very important because it will influence the effort spent in estimating, its accuracy, and the models used. Tight schedules with high risks require more accurate estimates than loosely defined projects with a relatively open-ended schedule. The estimators should look at the quality of the data upon which estimates are based and at the various objectives.

CONCLUSION

Software cost estimation is simple in concept, but difficult and complex in reality. The difficulty and complexity required for successful estimates exceed the capabilities of most software project managers. As a result, manual estimates are not sufficient for large applications. This paper has presented an overview of some software estimation techniques, providing an overview of several popular estimation models currently available. The important lesson to take from this paper is that no one method or model should be preferred over all others. The search for reliable, accurate and low cost estimation methods must continue. Also, more studies are needed to improve the accuracy of cost estimate for maintenance projects. The conclusion is that no single technique is best for all situations, and that a careful comparison of the results of several approaches is most likely to produce realistic estimates.
Some recommendations:

a. Do not depend on a single cost or schedule estimate.
b. Use several estimating techniques or cost models, compare the results, and determine the reasons for any large variations.
c. Document the assumptions made when making the estimates.
d. Monitor the project - the accuracy of the estimate.
e. Effective software process can be used to increase accuracy in cost estimation in a number of ways.
f. Maintaining a historical database

REFERENCES

[1]. "Handbook for Software Cost Estimation", Prepared by Karen Lum, Michael Bramble, Jairus Hihn, John Hackney, Mori Khorrami and Erik.

[2]. http://www.ceh.nasa.gov/downloadfiles/ Web%20 Links/cost_hb_public-6-5.pdf

[3]. "Software Cost Estimation" by Hareton Leung and Zhang Fan.

[4]. "Software Development Cost Estimation Approaches – A Survey", Barry Boehm, Chris Abts, Sunita Chulani.

[5]. http://www.ecfc.u-net.com/cost/index.htm

[6]. http://www.cs.odu.edu/~zeil/cs451/ Lectures/ 04mgmt/costest/costest_htsu3.html#subsubsect:parkinson

[7]. Software Cost Estimation in 2002, Capers Jones, Software Productivity Research Inc., Artemis Management Systems.

[8]. http://www.computing.dcu.ie/~renaat/ca421/ LWu1.html.

[9]. http://www.levela.com/software_cost_ estimating_swdoc.htm

[10]. Software Development Cost Estimating Guidebook, Software Technology Support Center Cost Analysis Group

[11]. The Comparison of the Software Cost Estimating Methods, *Liming W.*

[12]. http://www.cs.uwf.edu/~wilde/gump/ delphi.htm

[13]. http://www.compapp.dcu.ie/~renaat/ ca421/LWu1.html

