A REVIEW ON COST ESTIMATION MODALITIES IN SOFTWARE DEVLOPMENT

B. M. G. Prasad[#1]

Research Scholar
Department of Computer Science & Engineering
(PP.COMP.SCI & ENG.0064C)
Rayalaseema University,Kurnool
Andhra Pradesh,India-518007
bmgprasad@gmail.com

P. V. S. Sreenivas[#2]

[2]Professor
Department of Computer Science & Engineering
Sreenidhi Institute of Science and Technology,
Hyderabad, Telangana, India-501301
pvssrinivas23@gmail.com

ABSTRACT

Software estimation is a management activity that used to accurately estimate the size of the software project, the efforts, the schedule and the cost of the project, keeping some important aspects in mind, such as budget, planning, control, the tools to use and the appropriate historical data. The cost estimate can be defined as the approximate judgment of the costs of a project. Cost estimation will never be an exact science, because there are many variables involved in the calculation for a cost estimate, such as human, technical, environmental and political. In addition, any process that involves a significant human factor can never be exact, because human beings are too complex to be totally predictable. In addition, the software development for any project of reasonable size will inevitably include several tasks with complexities that are difficult to judge due to the complexity of the software systems.

Keywords: Cost estimation, effort estimation, COCOMO, expert judgment.

INTRODUCTION

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 project managers as well as system analysts and developers. Estimation is defined as “The action appraising assessing or valuing” or “The process of forming an approximate notion of numbers, quantities, magnitudes etc. without actual enumeration or measurement”. From these definitions it follows that task of estimation is not easy to do precisely. There are many software cost estimation methods are available. No one method is necessarily better or worse than the other, in fact, their strengths and weaknesses are often complimentary to each other. Estimating a effort required for software development is the most challenging and annoying job that requires expertise, experience as well as good understanding of process, project management,

metrics and most important use of proper estimation models, tools and techniques. Good software estimation models can significantly help the software project manager, project stakeholders to make informed decisions about biding values, planning the project, resource management, and delivering the project on time within budget. However, if managers use inaccurate estimation model [9] for making decisions that may be a recipe for disaster.

SOFTWARE ESTIMATION MODALITIES

The Software Engineering Body of Knowledge (SWEBOK) has identified Knowledge Areas (KAs) such as software requirements, software design, software construction, software testing, software maintenance, software configuration management, software engineering management, software engineering process, software engineering tools and method and software quality [8]. Software Engineering Management KA addresses management and measurement including Software Project Planning, which further addresses Effort, Schedule and Cost Estimation. Based on the breakdown of tasks, inputs and outputs the expected effort range required for each task is determined using calibrated estimation model based on historical size-effort data available; otherwise method like expert judgment is applied.

Boehm (1981) discusses seven techniques of software cost estimation [10]:

(1) Algorithmic cost modeling	A model is developed using historical cost information which relates some software metric (usually its size) to the project cost. An estimate is made of that metric and the model predicts the effort required.
(2) Expert judgement	One or more experts on the software development techniques to be used and on the application domain are consulted. They each estimate the project cost and the final cost estimate is arrived at by consensus.
(3) Estimation by analogy	This technique is applicable when other projects in the same application domain have been completed. The cost of a new project is estimated by analogy with these completed projects.
(4) Parkinson's Law	Parkinson's Law states that work expands to fill the time available. In software costing, this means that the cost is determined by available resources rather than by objective assessment. If the software has to be delivered in 12 months and 5 people are available, the effort required is estimated to be 60 person-months.

(5) Pricing to win	The software cost is estimated to be whatever the customer has available to spend on the project. The estimated effort depends on the customer's budget and not on the software functionality.
(6) Top-down estimation	A cost estimate is established by considering the overall functionality of the product and how that functionality is provided by interacting sub-functions. Cost estimates are made on the basis of the logical function rather than the components implementing that function.
(7) Bottom-up estimation	The cost of each component is estimated. All these costs are added to produce a final cost estimate.

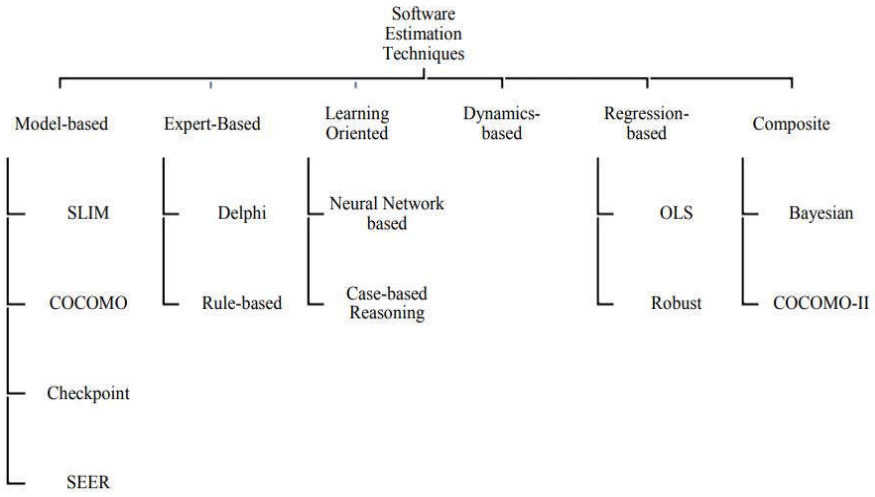

Fig 1: Software Effort and cost estimation techniques and tools

SLIM is based on Putnam's analysis of the lifecycle terms of so-called Rayleigh distribution of project personnel versus time. In SLIM, Productivity is used to link the basic Rayleigh manpower distribution model to software characteristics of size and technology factors.

The system enables a software planner to perform the following functions in an interactive session:

(1) *calibrate* the local software development environment by interpreting historical data supplied by the planner;

(2) create an *information model* of the software to be developed by eliciting basic software characteristics, personal attributes, and environmental considerations; and

(3) conduct software *sizing*--the approach used in SLIM is a more sophisticated, automated version of the LOC costing technique.

Checkpoint is knowledge-based software project estimation tool. It has proprietary database of about 8000 software projects. It uses Function Points as its primary input and focuses on areas that need to be managed to improve software quality and productivity. Checkpoint predicts effort at four levels of granularity: project, phase, activity and task. Estimates also include resources, deliverables, defects, costs and schedules.

The **PRICE-S** model was originally developed for use internally on software projects that were part of Apollo moon program. This tool estimates schedule, cost and resources for software projects of all types such as business processing systems, communication systems, space management systems, software components to extremely complex systems. This tool also uses function point method and certified by IFPUG.

ESTIMACS is a "macro- estimation model" that uses a function point estimation method enhanced to accommodate a variety of project and personnel factors.

The ESTIMACS tool contains a set of models that enable the planner to estimate

1. system development effort,
2. staff and cost,
3. hardware configuration,
4. risk,
5. the effects of "development portfolio."

SEER-SEM [3] For all types of software development and maintenance project, this tool is useful for estimating cost, labor, staffing, schedule, reliability and risk .This is a decision support tool. It follows the methodologies used in parametric models.

SELECT [2] Estimator was released in 1998. It is designed for large scale distributed systems development. It is object oriented, basing its estimates on business objects and components. It assumes incremental development life-cycle. The nature of its inputs allows the model to be used at any stage of the software development life-cycle, most significantly even at the feasibility stage when little detailed project information is known. In later stages, as more information is available, its estimates become correspondingly more reliable. The actual estimation technique is based upon ObjectMetrix developed by

Object Factory. It works by measuring size of a project by counting and classifying the software elements within a project. Applying the qualifier and technology adjustments to the base metric effort for each project element produces an overall estimate of effort in person-days, by activity. Using the total one man effort estimate, schedule is determined as a function of number of developers input as an independent variable.

COCOMO [6] cost and schedule estimation model was originally published in 1981. It became one of the most popular parametric cost estimation model of the 1980s. It has experienced difficulties in estimating the costs of software developed by following new life-cycle processes and capabilities. The COCOMO II research started in 1994 at USC to address the issues on non-sequential and rapid development process models, reengineering, reuse driven approaches and object oriented approaches.

Delphi technique [7] was developed at the Rand Corporation in the late 1940s originally as a way of making predictions about future events. More recently, the technique has been used as a means of guiding group of informed individuals to a consensus of opinion on some issue. Participants are asked to make some assessment regarding an issue, individually in a preliminary round., without consulting the other participants in the exercise. First round results are then collected, tabulated and returned to each participants for a second round, during which participants are again asked to make an assessment regarding the same issue. Abts and Boehm used the technique to estimate initial parameter values for Effort Adjustment Factors appearing in the glue code effort estimation components of the COCOTS integration model. This technique has been used by Chulani and Boehm to estimate software defect introduction and removal rates during various phases of the software development lifecycle. These factors appear in COQUALMO (COnstructive QUALity MOdel), which predicts the residual defect density in terms of number of defects/unit of size.

Learning-oriented techniques [1] include oldest as well as newest techniques applied to estimation activities. Former are represented by case studies, among the most traditional of manual techniques, later are represented by neural networks, which attempt to automate improvements in the estimation process by building models that “learn” from previous experience. Case studies represent inductive learning, whereby estimators and planners try to learn useful general lessons and estimation heuristics by extrapolation from specific examples. They examine in detail elaborate studies describing the environmental conditions and constraints that obtained during the development of previous projects, the technical and managerial decisions that were made and final successes or failures that resulted.

Neural networks are the most common software estimation model-building technique used as an alternative to mean least squares regression. These are estimation models that can be trained using historical data to produce ever better results by automatically adjusting their algorithmic parameters values to reduce the delta between known actual and model predictions.

Dynamics-based techniques [5] explicitly acknowledge that software project effort or cost factors change over the duration of the system development. Factors like deadlines, staffing levels, design requirements, training needs, budget etc. all fluctuate over the course of development and cause corresponding fluctuations in the productivity of project personnel.

Regression-based techniques are used in conjunction with model-based techniques and include standard regression, robust regression etc. Standard regression refers to the classical statistical approach of general linear regression modeling using least squares. This regression technique is used to calibrate COCOMO II.1997. Robust regression alleviates the common problem of outliers in observed software engineering data. Least Median Squares method falls in this category. Parametric cost models such as COCOMO II, SLIM, Checkpoint etc. use some form of regression based techniques due to their simplicity and wide acceptance.

Composite techniques incorporate a combination of two or more techniques to formulate the most appropriate functional form for estimation. Bayesian analysis is mode of inductive reasoning that provides a formal process by which a-priori expert judgment can be combined with sampling data to produce robust a-posteriori model information in a logically consistent manner in making inferences. This is been used in COCOMO II.

PROBLEM STATEMENT

- Software Estimation is considered as a highly important task.
- Estimation is done in stages of software Life Cycle termed as Budgetary, Initial, Progressive and Post Construction.
- Project Manager with Technical Expert prepare estimates that are cross verified. Knowledge of past project, existing empirical models and software tools are used.
- Post-facto data is maintained as a historical knowledge-base.
- The deviation of estimates with actual values varies by stages but budgetary estimates had variation in the range of 40 – 50 % in later stages range became narrower.

CONCLUSION

The problem that exists in the software cost estimation is the error rate that is estimated finally. The increase in error value leads to increased amount of cost in terms of software production which will remain as the

major drawback in software cost estimation process. This review has motivated the research to develop an alternate technique which can improve the cost estimation by reducing the error value.

REFERENCES

1. Ali, A, Qadri, S, Muhammad, SS, Abbas, J, TariqPervaiz, M & Awan, S 2010, “Software Cost Estimation through Entity Relationship Model”, vol. 6, no. 11, pp. 47-51.
2. Al-Sakran, H 2006, “Software cost estimation model based on integration of multi-agent and case-based reasoning”, vol. 2, no. 3, pp. 276-282.
3. Du, WL, Ho, D & Capretz, LF 2010, “Improving software effort estimation using neuro-fuzzy model with SEER-SEM”, vol. 10, no. 12.
4. Jiang, R 2009, “Required characteristics for software reliability growth models”, in Software Engineering, 2009. WCSE'09. WRI World Congress on, vol. 4, pp. 228-232.
5. Kitchenham, B, Mendes, E & Travassos, GH 2007, “Cross versus within-company cost estimation studies: A systematic review”, vol. 33, no. 5, pp. 316- 329.
6. Malik, A, Pandey, V & Kaushik, A 2013, 'An Analysis of Fuzzy Approaches for COCOMO II', International Journal of Intelligent Systems and Applications (IJISA), vol. 5, no. 5, p. 68.
7. Mittal, H & Bhatia, P 2007, 'A comparative study of conventional effort estimation and fuzzy effort estimation based on Triangular Fuzzy Numbers', International Journal of Computer Science and Security, vol. 1, no. 4, pp. 36-47.
8. Lee, W-T, Hsu, K-H, Lee, J & Kuo, JY 2012, 'Applying Software Effort Estimation Model Based on Work Breakdown Structure', in Genetic and Evolutionary Computing (ICGEC), 2012 Sixth International Conference on, pp. 192-195.
9. Reddy, CS & Raju, K 2009, 'Improving the accuracy of effort estimation through fuzzy set representation of size', Journal of Computer Science, vol. 5, no. 6, p. 451.
10. Syed Ali Abbas, XiaoFeng Liao, Afshan Azam and Ayesha Kulsoom Khattak, 2012, “Neural Net Back Propagation and Software Effort Estimation: A comparison based perspective”, ARPN Journal of Systems and Software, Vol. 2, No.6, pp.195-207.

About Authors:

Dr. P.V.S. Srinivas is presently serving as a Professor in Computer Science and Engineering, at Srinidhi Institute of Science and Technology, Hyderabad,Telanagana. He has got his Masters followed by Ph.D. in computer Science and Engineering in the area of Computer Networks from JNTU Hyderabad in the year 2003 and 2009 respectively. His main research interests are Wireless Communication, Mobile Computing and Mobile Ad hoc Networks. His research focuses in on "Designing an Effective and Assured Communication in MANETs" and improving QoS in MANETs. He is also serving as a Chief Panel Consultant in the area of wireless communications for a Hyderabad based company by name a SCADA METER SOLUTIONS Pvt Ltd. He has published 32 research papers in different refereed International Journals and conferences in India as well as abroad. He is also serving as an Editor-in-Chief for an International Journal IJWNC and also a peer reviewer for 3 International Journals.

Mr. B.M.G. Prasad presently serving as Professor in Computer Science and Engineering at Holy Mary Institute of Technology and Science, Hyderabad, Telangana. He is having 18 years of teaching experience in various Engineering Colleges. He has done his B.Tech at Vijaya Nagar Engineering College, Bellary in 1997. He completed his M.Tech at J.N.T.U College of Engineering, Anatapur in 2003 and now he is persuing his Ph.D in Computer Science and engineering under the guidence of Dr. P.V.S. Srinivas,, in RAYALASEEMA UNIVERSITY,KURNOOL A.P State. He has published papers in the I-Manager's International Journal and Indian National Science Congress, Tiruvananthapuram in the Computer Networks area. And his areas of interest are Software Engineering, Wireless Networks, Image Processing etc.