Pragmatic Peer Review Project Contextual Software Cost Estimation – A Novel Approach

Manoj Kumar Panda

HEAD OF THE DEPT,CE,IT & MCA

NUVA COLLEGE OF ENGINEERING & TECH NAGPUR , MAHARASHTRA,INDIA

Abstract

Software cost estimation is the process of predicting the effort required to develop a software system. Many estimation models have been proposed over the last 30 years. This Chapter provides a general overview of software cost estimation methods including the recent advances in the field. As a number of these models rely on a software size estimate as input, we first provide an overview of common size metrics. We then highlight the cost estimation models that have been proposed and used successfully. Models may be classified into 2 major categories: algorithmic and non-algorithmic. Each has its own strengths and weaknesses. A key factor in selecting a cost estimation model is the accuracy of its estimates. Unfortunately, despite the large body of experience with estimation models, the accuracy of these models is not satisfactory. The Chapter includes comment on the performance of the estimation models and description of several newer approaches to cost estimation. Keywords: project estimation, effort estimation, cost models. It can be used to determine what resources to commit to the project and how well these resources will be used. It can be used to assess the impact of changes and support re planning. Projects can be easier to manage and control when resources are better matched to real needs. Customers expect actual development costs to be in line with estimated costs. Software cost estimation involves the determination of one or more of the following

Key words :- Work break down structure (WBS),WA, Adjusted Function Points (AFP), Project Delivery Rate (PDRU), Project Elapsed Time (PET), Resource Level (RL) and Average Team Size (ATS)

1.1 Introduction

In recent years, software has become the most expensive component of computer system projects. The bulk of the cost of software development is due to the human effort, and most cost estimation methods focus on this aspect and give estimates in terms of person-months. Accurate software cost estimates are critical to both developers and customers. 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. Overestimating may result in too many resources committed to the project, or, during contract bidding, result in not winning the contract, which can lead to loss of jobs. Accurate cost estimation is important because:

It can help to classify and prioritize development projects with respect to an overall business plan.

Estimates:

effort (usually in person-months)project duration (in calendar time)

Cost (in Rupees)

Most cost estimation models attempt to generate an effort estimate, which can then be converted into the project duration and cost. Although effort and cost are closely related, they are not necessarily related by a simple transformation function. Effort is often measured in person months of the programmers, analysts and project managers. This effort estimate can be converted into a dollar cost figure by calculating an average salary per unit time of the staff involved, and then multiplying this by the estimated effort required. Practitioners have struggled with three fundamental issues:

Software cost estimation model to use

Software size measurement to use – lines of code (LOC), function points (FP), or feature point.

1.1 A Good Estimation

The widely practiced cost estimation method is expert judgment. For many years, project managers have relied on experience and the prevailing industry norms as a basis to develop cost estimate. However, basing estimates on expert judgment is problematic:

Project Delivery Rate (PDRU):-

This approach is not repeatable and the means of deriving an estimate are not explicit. It is difficult to find highly experienced estimators for every new project. The relationship between cost and system size is not linear. Cost tends to increase exponentially with size. The expert judgment method is appropriate only when the sizes of the current project and past projects are similar. Budget manipulations by management aimed at avoiding overrun make experience and data from previous projects questionable.

Resource level(RL) :- we must find the average resources available and the optimum use of the resources is must including the manpower e.g. the total available computer system add the uninterruptible energy supply to it and the manpower must be in buffer .

Work Breakdown Structure (WBS):- in project management and systems engineering, is a deliverable oriented decomposition of a project into smaller components. It defines and groups a project's discrete work elements in a way that helps organize and define the total work scope of the project.

Work Breakdown Structure(WBS):- element may be a product, data, a service, or any combination. A WBS also provides the necessary framework for detailed cost estimating and control along with providing guidance for schedule development and control

In the last three decades, many quantitative software cost estimation

models have been developed. They range from empirical models such as Boehm's COCOMO models to *analytical* models such as those in. An *empirical model* uses data from previous projects to evaluate the current project and derives the basic formulae from analysis of the particular database available. An *analytical model*, on the other hand, uses formulae based on global assumptions, such as the rate at which developer solve problems and the number of problems available. Most cost models are based on the size measure, such as LOC and FP, obtained from size estimation. The accuracy of size estimation directly impacts the accuracy of cost estimation.

Although common size measurements have their own drawbacks, an organization can make good use of any one, as long as a consistent counting method is used. A good software cost estimate should have the following attributes. It is conceived and supported by the project manager and the development team. It is accepted by all stakeholders as realizable.

It is based on a well-defined software cost model with a credible basis.

Project Elapsed Time (PET):-The project elapsed time is the duration of total time we must finish the project or the stipulated time during that time at any cost we must deliver the product to the client

It is based on a database of relevant project experience (similar processes, similar technologies, similar environments, similar people and similar requirements).It is defined in enough detail so that its key risk areas are understood and the probability of success is objectively assessed. Software cost estimation historically has been a major difficulty in software development. Several reasons for the difficulty have been identified: Lack of a historical database of cost measurement Software development involving many interrelated factors, which affect development effort and productivity, and whose relationships are not well understood Lack of trained estimators and estimators with the necessary expertise Little penalty is often associated with a poor estimate.

1.2. Process Of Estimation

Estimation is an important part of the planning process. For example, in the top-down planning approach, the cost estimate is used to derive the project plan:

1.2.1.The project manager develops a characterization of the overall functionality, size, process, environment, people, and quality required for the project.

1.2.2 A macro-level estimate of the total effort and schedule is developed using a software cost

Estimation Model.

1.2.3 The project manager partitions the effort estimate into a top-level work breakdown structure. He also partitions the schedule into major milestone dates and determines a staffing profile, which together forms a project plan.

1.2.4The actual cost estimation process involves seven steps:

1.2.5 Establish cost-estimating objectives

1.2.6 Generate a project plan for required data and resources

1.2.7. Pin down software requirements

1.2.8. Work out as much detail about the software system as feasible

1.2.9. Use several independent cost estimation techniques to capitalize on their combined strengths

1.2.10. Compare different estimates and iterate the estimation process

1.2.11. After the project has started, monitor its actual cost and progress, and feedback results to project management.

Average Team Size(ATS):- the total team size dependent on the total project manhours if the average project duration is two years then we must take the sufficient number of human resource in to our project if again the dead line is nearing then the manpower must be enhanced as per the real time situation .

No matter which estimation model is selected, users must pay attention to the following to get best results:coverage of the estimate (some models generate effort for the full life-cycle, while others do not include effort for the requirement stage)calibration and assumptions of the model sensitivity of the estimates to the different model parameters deviation of the estimate with respect to the actual cost.

2 The Outputs Of This Step Are As Follows:

2.1 Assumptions made to revise estimates

2.2 Methods used to revise estimates

2.3 Revised size, effort, schedule, and cost estimates

2.4 Revised functionality and procurements

2.5 Updated WBS

2.6 Revised risk assessment

Review and Approve the Estimates

The purpose of this step is to review the software estimates and to obtain project and line management approval.

3. Conduct A Peer Review With The Following Objectives:

3.1 Confirm the WBS and the software architecture.

3.2 Verify the methods used for deriving the size, effort, schedule, and cost. Signed work agreements may be necessary.

3.3 Ensure the assumptions and input data used to develop the estimates are correct.

3.4 Ensure that the estimates are reasonable and accurate, given the input data.

3.5 Formally confirm and record the approved software estimates and underlying

Assumptions for the project:

4. The software manager, software estimators, line management, and project management approve the software estimates after the review is complete and problems have been resolved. Remember that costs cannot be reduced without reducing functionality.

The outputs of this step are as follows:

1 Problems found with the estimates
2 Reviewed, revised, and approved size, effort, schedule, cost estimates, and assumptions
3 Work Agreement(s), if necessary Track, Report, and Maintain the Estimates
The purpose of this step is to check the accuracy of the software estimates over time, and provide the estimates to save for use in future software project estimates.

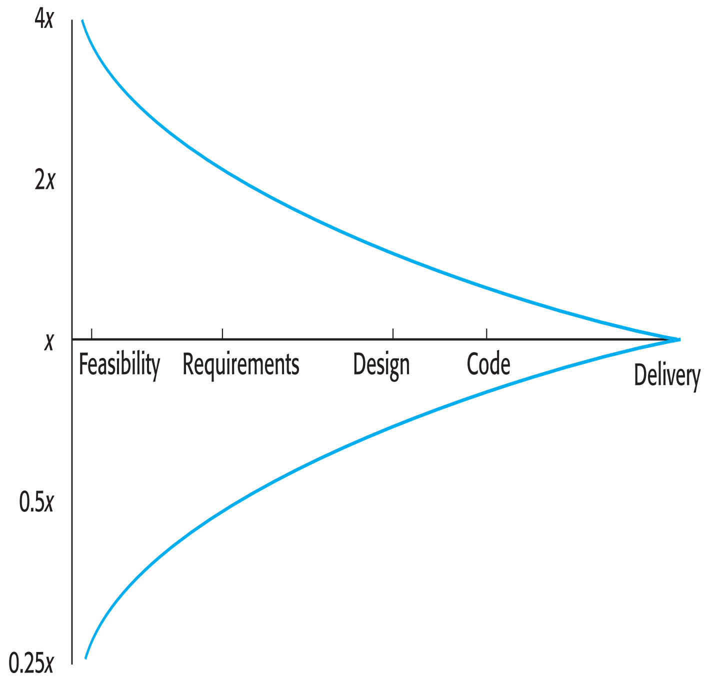
1. Track the estimates to identify when, how much, and why the project may be overrunning or under-running the estimates. Compare current estimates, and ultimately actual data, with past estimates and budgets to determine the variation of the estimates over time. This allows estimators to see how well they are estimating and how the software project is changing over time.

2. Document changes between the current and past estimates and budgets.

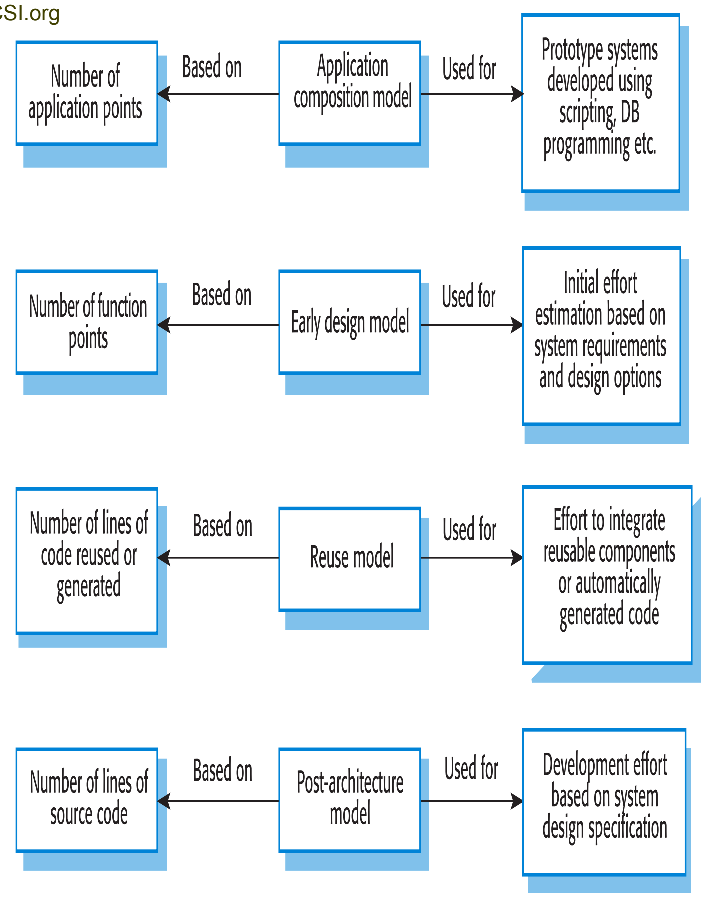
3. In order to improve estimation and planning, archive software estimation and actual data each time an estimate is updated and approved, usually at each major milestone. It is recommended that the following data be archived:

4 Project contextual and supporting information

− Project name
− Software organization
− Platform
− Language
− Estimation method(s) and assumptions
− Date(s) of approved estimate(s)
4.1 Estimated and actual size, effort, cost, and cost of procurements by WBS work element
4.2 Planned and actual schedule dates of major milestones and reviews
4.3 Identified risks and their estimated and actual impacts
The outputs of this step are as follows:
4.4 Updated tracking comparisons of actual and estimated data
4.5 Evaluation of the comparisons
4.6 Updated size, effort, schedule, cost estimates, and risk assessment
4.7 Archived software data, including estimates and actuals



(Figure .1)


Number of application points
Based on
Application composition model
Used for
Prototype systems developed using scripting, DB programming etc.
Number of function points
Based on
Early design model
Used for
Initial effort estimation based on system requirements and design options
Number of lines of code reused or generated
Based on
Reuse model
Used for
Effort to integrate reusable components or automatically generated code
Number of lines of source code
Based on
Post-architecture model
Used for
Development effort based on system design specification

(Figure 2)

CONCLUSION

In this approach we aimed at addressing the problem of large variances found in available historical data that are used in software cost estimation. Project data is expensive to collect, manage and maintain. Therefore, if we wish to lower the dependence of the estimation to Computational Intelligence in Software Cost Estimation: Evolving Conditional Sets of Effort Value Ranges 17 the need of gathering accurate and homogenous data, we might consider simulating or generating data ranges instead of real crisp values.

The theory of conditional sets was applied in the present work with Genetic Algorithms (GAs) on empirical software cost estimation data. GAs are ideal for providing efficient and effective solutions in complex problems; there are, however, several trade-offs. One of the major difficulties in adopting such an approach is that it requires a thorough calibration of the algorithm's parameters. We have tried to investigate the relationship between software attributes and effort, by evolving attribute value ranges and evaluating estimated efforts. The algorithm promotes the best individuals in the reproduced generations through a probabilistic manner. Our methodology attempted to reduce the variations in performance of the model and achieve some stability in the results. To do so we approached the problem from the perspective of minimizing the differences in the ranges and the actual and estimated effort values to decisively determine which attributes are the most important in software cost estimates.

We used the ISBSG repository containing a relatively large quantity of data; nevertheless, this data suffers from heterogeneity thus presents low quality level from the perspective of level of values. We formed three different subsets selecting specific cost attributes from the ISBSG repository and filtering out outliers using box-plots on these attributes. Even though the results are of average performance when using the first two datasets, they indicated some importance ranking for the attributes investigated. According to this ranking, the attributes Added Count (AC) and File Count (FC) were found to lay among the most significant cost drivers for the ISBSG dataset. The third dataset included Adjusted Function Points (AFP), Project Delivery Rate (PDRU), Project Elapsed Time (PET), Resource Level (RL) and Average Team Size (ATS). These attributes may be measured early in the software life-cycle, thus this dataset may be regarded more significant than the previous two from a practical perspective. A careful and stricter filtering of this dataset provided prediction improvements, with the yielded results suggesting small value ranges and fair estimates for the mean effort of a new project and its deviation. There was also an indication that within different areas of the data, significantly different results may be produced. This is highly related to the scarcity of the dataset itself and supports the hypoChapter that if we perform some sort of clustering in the dataset we may further minimize the deviation differences in the results and obtain better effort estimates.

Although the results of this work are at a preliminary stage it became evident that the approach is promising. Therefore, future research steps will concentrate on ways to improve performance, examples of which may be: (i) Pre-processing of the ISBSG dataset and appropriate clustering into groups of projects that will share similar value characteristics. (ii) Investigation of the possibility of reducing the attributes in the dataset by utilizing a significance ranking mechanism that will promote only the dominant cost drivers. (iii) Better tuning of the GA's parameters and modification/enhancement of the fitness functions to yield better convergence. (iv) Optimization of the trial and error weight factor assignment used in the present.

BIBLIOGRAPHIES & REFERENCES

1. A. J. Albrecht, and J. E. Gaffney, "Software function, source lines of codes, and development effort prediction: a software science validation", *IEEE Trans Software Eng.* SE-9, 1983, pp.639-648.
2. U. S. Army, *Working Schedule Handbook, Pamphlet No.* 5-4-6, Jan 1974.
3. J. D. Aron, *Estimating Resource for Large Programming Systems*, NATO Science Committee, Rome, Italy, October 1969.
4. R.K.D. Black, R. P. Curnow, R. Katz and M. D. Gray, *BCS Software Production Data*, Final Technical Report, RADC-TR-77-116, Boeing Computer Services, Inc., March 1977.
5. B. W. Boehm, *Software engineering economics*, Englewood Cliffs, NJ: Prentice-Hall, 1981.
6. B.W. Boehm et al "The COCOMO 2.0 Software Cost Estimation Model", *American Programmer*, July 1996, pp.2-17.
7. L. C. Briand, K. El Eman, F. Bomarius, "COBRA: A hybrid method for software cost estimation, benchmarking, and risk assessment", *International conference on software engineering*, 1998, pp. 390-399.
8. G. Cantone, A. Cimitile and U. De Carlini, "A comparison of models for software cost estimation and management of software projects", in *Computer Systems*: *Performance and Simulation*, Elisevier Science Publishers B.V., 1986.
9. W. S. Donelson, "Project planning and control", *Datamation*, June 1976, pp. 73-80.
10. N. E. Fenton and S. L. Pfleeger, *Software Metrics*: *A Rigorous and Practical Approach*, PWS Publishing Company, 1997.
11. D. V. Ferens, and R. B. Gumer, "An evaluation of three function point models of estimation of software effort", *IEEE National Aerospace and Electronics Conference*, vol. 2, 1992, pp. 625-642.
12. G. R. Finnie, G. E. Wittig, AI tools for software development effort estimation*, Software Engineering and Education and Practice Conference*, IEEE Computer Society Press, pp. 346-353, 1996.
13. M. H. Halstead, *Elements of software science*, Elsevier, New York, 1977.
14. P. G. Hamer, G. D. Frewin, "M.H. Halstead's Software Science – a critical examination", *Proceedings of the 6th International Conference on Software Engineering*, Sept. 13-16, 1982, pp. 197-206.
15. F. J. Heemstra, "Software cost estimation", *Information and Software Technology*, vol. 34, no. 10, 1992, pp. 627-639.