Journal of Advances and Scholarly Researches in Allied Education
Vol. 18, Issue No. 4, July-2021, ISSN 2230-7540

Techniques of Software Cost Estimation

Amit Shrivastava[1]*, Dr. Rajeev Yadav[2]

[1] Research Scholar, Shri Krishna University, Chhatarpur M.P.

[2] Professor, Shri Krishna University, Chhatarpur M.P.

Abstract - One of the world's most rapidly expanding sectors right now is computer software. Effective strategies for predicting software costs are clearly needed to reduce expenses and make software more competitive in light of the rising price of software development. Software cost assessment has made use of a variety of computational methodologies. As a result, it may be difficult for software managers to allocate resources or keep expenses under control if estimates of development time and cost vary widely. Predicting the amount of time and effort it will take to implement a software system is the primary goal of software cost estimate. It's a crucial job that aids the software industry in properly managing the software development process they're involved in.

Keywords - Techniques, Software, Cost Estimation, etc.

----------X----------

INTRODUCTION

A software cost estimate is a prediction of the amount of time and effort required to complete a software project's development. Accurate estimations of project growth, delivery time-to-market accuracy, and cost control can only be achieved if they are estimated accurately. This is regarded the first and most important phase in the software development process. In the long run, an accurate estimate of a current software project will help the business better plan its future software initiatives as well. For the reasons outlined above, software effort estimate has drawn the attention of several scholars for the better part of two decades. (1)

As a result, a project's total software cost is calculated by adding together estimates for both the time and work required to produce the program. Work hours and the number of people involved in the soft project development are included in the overall effort. Even in the early days of software project development, companies of this type encountered a problem with inaccurate estimates of project work and project duration. The availability of imprecise information about the software project to be built at the time of its estimating process was and continues to be a good reason for this. Project managers can only get an accurate picture of how their software development project is progressing if they have a better estimate of the final result. However, this in general provides the organization with a better understanding of resource consumption, which will lead to a better timetable for future projects. There have been various methods of software cost estimating presented over the years, ranging from algorithmic to non-algorithms to data mining to algorithmic met heuristics, but none of them have fulfilled the accepted benchmark for accuracy in software development project estimation. In addition to this, a variety of artificial neural network-based models and their hybrids have been created, and have demonstrated improvements in the stated. A mix of artificial neural network and several met heuristic optimization methods was used in this study to construct an estimation model for software cost estimation that is accurate. (2)

BASIS OF ESTIMATION

There must be a clear and explicit explanation of how the cost estimate is derived, the information on further details necessary, including their amount and sort, as well as their variable nature depending on the application. Cost estimates for activities incorporate additional information such as the following:

- A complete estimate basis documentation showing how the estimate wasprepared.
- A full blueprint of estimate coverage representing what to estimate.
- A thorough record of assumptions to be made.
- Record of constraints.
- Definition of self-assurance scale of final estimate.
- Declaration of possible range of estimates that can be possible like an IT item is supposed to cost from a range of 1000 (INR) plus minus 10(INR)

In other words, if the information outlined above about the estimated is properly defined and

recorded, it becomes easier to go through extensive assessments and receive easy revisions of prospective estimations. A shaky basis for estimation, on the other hand, renders activity estimations pointless and futile. (3)

TIME OF ESTIMATION

In the software development process, software cost estimating is always seen as an iterative or umbrella activity that encompasses all stages and is intimately linked to every other activity, never as a stand-alone task. It is from requirements collecting for the software project that estimates are built, and these requirements never appear to be definite and unambiguous, therefore this ambiguity is carried over to software estimates as well. In addition, since other procedures, tools, and project qualities have a significant impact on requirements, this adds even more ambiguity to the software cost estimating process. During the course of a software development project, it is essential to continuously improve software cost estimates in order to account for new information as it becomes available. It is also necessary to produce many cost estimates during the software development process for larger projects. The following must be included in all of these cost estimates: (4)

- A roughly done pre-requirements guesstimate
- An initial but concrete official estimate to be derived from the requirements of software project to be developed.
- Other middle level estimates for accommodating changes in requirements.
- A cost build-up using ancient historical data of the similar projects.

Cost estimation and design activities appear to interact often in the early stages of the software development life cycle, which suggests a strong relationship between the two. For this reason, software cost estimation is more closely associated with management activities such as risk management, resource allocation, and resource allocation planning in the latter stages of the software development life cycle.

REASON OF INACCURACY

Many software development organizations throughout the world have a problem with underestimating the amount of time and work required to complete a project, which leads to numerous problems in the finished product. A typical "industry joke" is that software development effort equals the multiplication of working hours into Pie and then adding 30 percent to it, which has been widely mocked. Software development, on the other hand, is a process of incremental improvement that begins with a clear picture of what the final product will look like and how it will function. Having a comprehensive understanding of the software product's needs and its related goals helps to fine-tune the remainder of the software project estimating details. As a result, as a software project progresses through its life cycle, the accuracy of its estimation will improve. (5) When it comes to the estimating process and lack of a clear understanding of the project to be produced, Barry Boehm claims that software cost estimates made in the early phases of software development might be reduced by four factors.

SOFTWARE COST ESTIMATION AND SOFTWARE QUALITY

Any software development product's quality is affected by the software cost estimation. It's common to see the word "software quality" used in a narrow sense and separated from the actual development process. As a result, the notions of "usability," "error-freeness," and "extensibility" are the sole determinants of software quality. Product quality, therefore, disregards both cost and time, which are the two most important considerations. On the other hand, essential software quality, which may really be referred to as software process quality, must encompass these characteristics as well. The best software in terms of process quality is frequently just good enough in terms of product quality. The purpose of this part is to aid in the understanding of software quality as a whole and, in turn, the overall quality of the final product. (6)

REAL ESTIMATION, MEASUREMENT AND ITS RELATIONSHIP

A better software estimating method will never be improved by measuring only the three economic criteria of effort, time, and product. Furthermore, it is impossible to get the treasonable values for the parameters before at least one of the estimations for product and effort and product and time have been performed.

In order to get an accurate estimate, the first step is to build a model of the final product. Modeling effort and time, as well as the model's product metric, must be measured in order to build a model of this sort. The second critical step is to link the parameters of the modeling process to those of a complete manufacturing process, with the goal that the earlier may forecast the later. Not even software engineering can claim credit for the aforementioned standards, which are drawn from other, more archaic fields of engineering like architecture and automobile production.

The architectural blueprints that are drawn out by the architects before any house is completed are a well-known example of this kind of estimating procedure. While their plans are primarily used to meet standards like the minimum allowable distances between its surrounding houses and many others, they are also used as a model for cost and time estimations during the construction of the building process itself. In order to arrive at the predictor value, the building's volume is calculated on the plan. These "magic" parameters are generally chosen and sealed by architects' associations, which are statistically generated and adapted to the project's specific requirements. (7)

The only reasonable approach to estimating is the three-point estimation structure, which is also known as measurement-based estimation. In addition, Albrecht's Function point analysis and Boehm's COCOMO implicitly employ this similar method.

PSYCHOLOGICAL AND ORGANIZATIONAL BARRIERS TO RATIONAL ESTIMATION

This subject can be better understood if we consider the persons who are usually engaged in the estimating process.

- A manager of the software producing unit (SPU).
- The software products client and
- The project manager.
- When it comes to promoting the SPU, the SPU's manager has a vested interest. Because of this, he is prepared to offer the software product at any price the consumer is willing to pay. In turn, the buyer is always prepared to spend as little as possible.

As a further concern, the ability to accurately anticipate project costs at an early stage might have a substantial impact on project deployment in the future. To put it another way, knowing the exact cost of a software project as early and logically as feasible would have a significant influence on future decisions about the start or end of similar software initiatives. (8)

SOFTWARE COST ESTIMATIONAND RISK MANAGEMENT

Estimating the cost and size of software is a prediction issue of enormous importance. The scope of software cost assessment is not limited to software development corporations, but also includes research and development organizations, aircraft organizations, and many other organizations that generate software. Solid software project size estimation is critical to successful cost estimation since the size of the project is the most important element in determining the cost of software. Slightly seeking the faultless method for estimating accurate size and cost, a more genuine approach to enhance estimation is to minimize the risks allied with inadequate sizing and inaccurate costing of software in parallel. By providing information on how to mitigate risk while making decisions on various pricing and size possibilities, this section is intended to assist experienced cost experts in better understanding the causes of ambiguity and risk. These two methodologies have the power of increasing accountability for holding risks associated to software cost estimations, which is a prevailing impression in any such form of examination.

- Risks and uncertainties can be identified in the early stages of the process.
- It is important to conduct a thorough analysis of the whole software cost estimating process in order to identify areas where risk mitigation may be improved.

An increase in the tenacity of analysts in documenting uncertainty is a byproduct of the first one. The second one, on the other hand, is the one that really addresses and eliminates risks throughout the software cost estimating process. (9)

MERITS, DEMERITS AND RISKS IN SIZE ESTIMATION

In order to construct a system, different languages and other system compositions, such as using fresh, recycled, and modified system code, have different virtues and disadvantages when it comes to determining system size. Risks connected with each decision can be mitigated if competent sizing-estimation procedures are used. However, while utilizing a sizing approach, various global factors must be taken into account. They're discussed in the following paragraphs:

- It is possible to count the amount of code lines or requirements in a project, although this is not recommended. The advantages include the simplicity of counting and counting automation, the freedom of programming language selection, the flexibility of storing in a historical database, and the flexibility of regulating organizations' understanding of how the system works. (10) Flaws include issues counting early in the development phase, reliance on

programming or specifications style and inconsistencies among languages.

- Applicability points or function points can be counted. At the beginning of the software development process, these objects may be easily specified, but tracking their evolution is difficult because of their theoretical nature. Easy creation from a clear specification and consistency among intermediate products are some of the advantages (such as design or early code modules). Analysts' interpretation of the estimated constructions and the complexity of calculating embedded system size are two disadvantages.

- Particularly when it comes to new sizing procedures, there is no empirical evidence. It's possible to use a novel scaling approach for every new methodology, but without any empirical data to indicate acceptable input variables.

- Relying on previous project knowledge and expertise. Most of the estimating methods rely on some level of access to historical data from previous projects. This trust may be used to govern previous projects' instructions, thereby reducing the risk of inconsistent input values. Furthermore, a project's resemblances to other projects may hide crucial distinctions. It is also not possible to obtain historical data in the necessary format for the new sizing approach.

- To keep an eye on progress throughout time. Customers' and developers' expectations can be better managed if progress can be tracked by hiding the size of files. Nonetheless, several size models were explored at the beginning of the development process, but not in the middle of the development process. Obviously, a size estimate based on a single objective scenario may no longer be relevant if the aim changes.

- The model's calibration. According to the organization's or development's style, standardization adjusts the model. A properly calibrated model is likely to be more accurate than a model that was previously calibrated. However, the traditional approach will never be able to accurately estimate new or fundamentally different projects.

There are risks at every stage of a project's lifecycle, and they must be tied to specific actions or dates. Voids are brought into the estimating process by default when any choice, no matter how minor or large, such as the decision to build a certain software module, is made. This choice merely multiplies the possibility of mistake by the risk factor. It is much more difficult to provide accurate cost estimates at the beginning of a software development project because of the inherent uncertainty of the process. (11) That's why appropriate risk identification and mitigation are so critical.

COST ESTIMATION ISSUES

There is a certain sequence of actions that must be followed in order to get an accurate estimate of the cost of software development. However, the precision of the cost assessment process is hampered by several difficulties. The following are some of the difficulties that have been identified:

Requirement Issues

As a necessary first step in every new software development project, gathering requirements for new code is an important part of any project's success. There are two types of requirements: functional and non-functional. These characteristics characterize how a system behaves, and they serve as a foundation for all subsequent stages of project activity. The accuracy of software sizing and cost estimation is directly related to the accuracy of the requirements, which are created by the requirements themselves. Inaccurate estimations are caused by issues with requirements. If errors in the requirements definition are not caught and passed on to the next level, they might become more difficult to rectify. (12)

Software Sizing Issues

The first step in software cost estimates is to estimate the amount of the deliverables that must be created. However, the process of calculating the size of a software project is called software sizing. It is reliant on a lot of factors to determine the size of software. Programs that are expected to accomplish a large number of difficult tasks with a higher degree of reliability are, by definition, larger in size. Because of this, the size estimation requires a good understanding of the project scope, its complexity, and relationships. (13)

Software Metric Issues

In software engineering, it's an integral part of the software cost estimating process. Measuring the progress of software development is an important part of software metrics. There is a strong likelihood that there are many possible software metrics based on all of the conceivable software objects and all of the possible properties of these things. The foundation of every measurement program must be a comprehensive and all-encompassing measurement strategy, with the metrics focusing on a specific product, resource, process, or project goal.

There are still issues, such as selecting the correct metrics for a project, and others of a similar nature. It has been a while since we've seen a study that accurately reflects the problems of metrics in object-oriented software and function point findings.

Software Complexity Issues

The complexity of a software design is measured by how difficult it is to understand and verify. Software cost estimation is complicated by the presence of both dependent and independent factors, both of which have an impact on the accuracy with which costs for software development projects may be estimated. The size of a project's team has a direct correlation to its complexity, and a smaller team means better control over the project's timetable. (14)

CONCLUSION

The current status of software engineering operations is accurate software cost assessment, which is of course a difficult procedure. The calibration of the software data may be used to determine the accuracy of individual software effort estimate models, and this has been proven using hybrid estimation models in this research. Using performance validation metrics like the magnitude of relative error (MRE), mean relative error (MMRE), and median relative error (MRE), the improvement has been demonstrated and clearly compared to the classic COCOMO II and other previously developed models (MdMRE). the suggested input selection and artificial neural network approach delivers greater performance because the inclusion of input selection procedure results in a model that is more stable since budding co linearity between attributes is decreased. As a result, a model with a smaller number of attributes is preferred over a model with a larger number of features.

REFERENCES

1. Chinchu Mary Jose, Ambili S(2016), "Prediction of Cost of Quality Using Artificial Neural Network In Construction Projects"

2. Javad Pashaei Barbin and Hassan Rashidi (2015), "A decision support system for estimating cost of software projects using a hybrid of multi-layer artificial neural network and decision tree"

3. Gharehchopogh, F. S., Maleki, I., and Talebi, A., "*Using hybrid model of artificial bee colony and genetic algorithms in software cost estimation*". In Application of Information and Communication Technologies (AICT), 9th International Conference on (pp. 102-106). IEEE, 2015

4. Efi Papatheocharous and Andreas S. Andreou (2015), "Software cost estimation using artificial neural networks with inputs selection"

5. Bayram S, Ocal, M., Oral, L.E and Atis, C. (2015). Comparison of multi layer perceptron (MLP) and radial basis function (RBF) for construction cost estimation: the case of Turkey, Journal of Civil Engineering and Management, 22(4), 480-490.

6. Roxas, C.L.C and Ongpeng, J.M.C. (2014). An Artificial Neural Network Approach to Structural Cost Estimation of Building Projects in the Philippines, Proc. DLSU Research Congress 2014 De La Salle University, De La Salle University, Manila, Philippines, 2014.

7. Mukherjee S.: Optimization of Project Effort Estimate Using Neural Network IEEE International Conference on Advanced Communication Control and Computing Technologies, pp 406-410,(2014).

8. Lhee, S.C., Flood, I. and Issa, R. (2014). Development of a two-step neural network-based model to predict construction cost contingency, Journal of Information Technology in Construction, 1, 399-411.

9. Hany El-Sawah (2014), "Comparative study in the use of neural networks for order of magnitude cost estimating in construction"

10. Bardsiri, A. K., and Hashemi, S. M.,"*Software effort estimation: A survey of well-known approaches*". International Journal of Computer Science Engineering (IJCSE), 3(1), 46-50. 2014

11. Nassif, A. B., Ho, D., and Capretz, L. F.,"*Towards an early software estimation using log-linear regression and a multilayer perceptron model*", Journal of Systems and Software, 86(1), 144-160. 2013

12. Naik, G.M. and Kumar, M. (2013). Project Cost and duration optimization Using Soft Computing Techniques, Journal of Advanced Management Science, 1(3), 299-203.

13. Kumari S.: Performance Analysis of the Software Cost Estimation Methods A Review, International Journal of Advanced Research in Computer Science and Software Engineering, Issue 7,vol 3,(2013).

14. Kaushik A., Soni A.K, Soni R.: A Simple Neural Network Approach to Software Cost Estimation, Global Journal of Computer Science and Technology Neural and

Artificial Intelligence, Issue 1, vol 13, Version 1.0,(2013)

Corresponding Author

Amit Shrivastava*

Research Scholar, Shri Krishna University, Chhatarpur M.P.