AN IMPROVED METHOD FOR ESTIMATING SOFTWARE COSTS IN AGILE SOFTWARE DEVELOPMENT USING SOFT COMPUTING TECHNIQUES

Dr. Sachin Joshi

Assistant Professor, Bundelkhand University, Jhansi (U.P).

Abstract

The management & estimation of agile projects is stimulating works for many software companies for their high failure rates. To develop successful software projects. Proper prediction of projects overall effort & cost evaluation is a very important task. The numbers of development models over the last few decades have evolved through software projects. Hence, to complete an exact estimation of exertion & taken a toll for diverse program ventures which is based on distinctive improvement models are having innovative & new steps of software development is a significant task which is to be done. Proper forecast is always helpful for a successful software project with extreme budget and with no delay but the percentage of any misconduct in the overall development cost & effort lead the estimation of project failure in the terms of delivery time, feature or budget. Software companies have adopted different various development models which are based on the organization and requirement of project. While the changes of adaptability in a software projects. ASD (Agile Software Development) has ended up a more well-known and fruitful system for advancement on the final decade. The client is included for the improvement utilizing a spry system as member. Such methods are more adaptable and nearness of bio-intelligence increments their precision. In this paper we proposed a COCOMO (Constructive Cost Model) and SLIM technique for cost estimation of better software projects. Profit or loss estimation forecast to new project is carried out with the help of historical data of company. In the machine learning to predict forecast using historic data Naïve Bayes algorithm plays vital role and provides great accuracy. The proposed model yields better results in terms of MMRE.

Keywords: COCOMO model, Cost Estimation, SEERA dataset, Naïve Bayes, prediction.

► *Correspondence Author: Dr. Sachin Joshi*

Introduction

Software project management is collection of two activities: Project Planning and Project Monitoring and control. Planning is predicting the activities that must be done before starting development work. Once venture work is begun it is the obligation of venture supervisor to screen the work and see the objective tall quality of computer program must be created with moo taken a toll and inside a time and budget. The input for the arranging is SRS-Software Prerequisite Detail Report and yield is venture arrange basically incorporates Fetched estimation and Plan estimation. Computer program taken toll estimation is the method of anticipating the sum of time required to construct a package. But after some years' program advancement begun getting to be commercial as increasingly computer program ventures were begun to be requested by diverse sort of clients (Uncommonly for defense purposes). Measure of ventures begun getting to be enormous with the increment in functionalities of can be extending. Root of dexterous computer program advancement strategies around 2000 incorporates a solid foundation history in computer program improvement. Spry computer program advancement has been accepted around the world by diverse computer program improvement organizations [7]. A spry strategy permits for reacting to change put by the client at any point of advancement. This empowers ceaseless testing and upkeep. Thus a profoundly adaptable and quality item is created. Brief advancement cycle and early working item model increment its efficiency. Moreover, dynamic client inclusion makes selection simple for any alter which leads to keenness. Due to persistent alter within the dexterous improvement strategies, it is difficult to induce a precise gauge other than conventional strategies of estimation have as of now fizzled to allow precise and exact arrangement to gauges. Later inquires about have appeared that delicate computing strategies are getting to be very reasonable for taking care of issues like fetched estimation, optimization, machine learning, estimating, etc. Numerous delicate computing methods like Mamdani FIS, ANNs (i.e. Relapse Neural Systems, Outspread Premise Capacities (RBF), Counter Propagation Neural Network (CPNN), etc.), MMRE Bio-Inspired Procedures (i.e. PSO, GA, etc.) are being connected effectively for assessing taken a toll and exertion in dexterous computer program advancement environment is utilized in existing framework.

Software cost estimation methods utilize historical datasets of previous projects to estimate project costs and effort. These methods must encompass the ever-changing software development landscape and the impact of country-specific environmental and cultural factors on software development practice. Research in empirical software engineering and cost estimation has recognized that the relevance and the representativeness of the dataset are important for accurate and realistic cost estimation

Literature Review

A literature review allows one to get an insight into the different aspects of the problem being studied. It explores innovative computational methods, shines light on how to enhance data collecting performance and proposes strategies to maximize data collection and understanding effectiveness. Therefore, reviewing literature is an essential step in the development of the research project. A literature review is a body of text intended to evaluate the key points of existing knowledge including empirical studies as well as theoretical and analytical approaches to a particular subject. Literature reports are secondary sources, which have no current or initial scientific research published. Over the final decade, various examinations have been conducted in arrange to recognize issues influencing program upkeep and taken a toll estimation. Diverse investigate think about has been conducted over the long time, in arrange to extricate prove, indeed on the Bequest frameworks.

Use case points

A use case point (UCP or UCPs) is a technique which is used for software estimation. This technique is used to estimate the size of software development projects. UCP- Use case points are used when Unified Modeling Language (UML) and Rational Unified Process (RUP) methodologies are used for the software design and development. It is a UCP software estimation technique used to calculate the actual size of software with use cases.

D. S. Senevirathne et al [1] conducting a study of comparisons questionnaire, which is having highly extracting knowledge regarding aspects of affecting the software maintenance. This paper shows the survey of software estimation which help to identify various resources that are being used for software development maintenance.

R. Sari Dewi et al [5] within the estimate estimation program, there are numerous strategies that have demonstrated their unwavering quality. One of them is Use Case Point (UCP). UCP incorporates a well-known advantage based on the utilities case circumstance which might be a transformation of the client story within the Software Requirement Specification (SRS) report. We used COSMIC and UCP method for getting high accuracy for cost estimation.

Shofiyah Al Idrus et al [8] software development project is a failure which can be caused by an unseemly and incapable administration approach in developing software projects. This system makes an application for cost estimation by using Use Case Point (UCP) but this application is not having a more feasibility.

Some Related papers with other methods

S. S. Ali et al [2] the knowledge representation of expert system is used where the terms of If-then rules. A rule is related with each and every weight for managing with dubious information. This framework includes 394 rules & 101 activities. The information base is able to learn & enhance the database.

M. Wang et al [3] the trustworthiness of software is a vital property. Software development increases the taken a toll of program with its change by dependability of program. The proposed algorithm of allocation can allocate improvement cost to each component to optimize software development reliability. Moreover, to arrange the designate advancement cost to every component automatically. A web-based computer program instrument for apportioning improvement costs to each component is created. They got 80% exactness with this demonstrate.

L. S. Nair et al [4] software Program support brings about a critical fetched amid the life cycle of any computer program item. This paper proposes a technique to diminish the testing exertion through progressive discharges by deciding a set of most critical functionalities by computing scores based on the utilization recurrence, code viability and alter powerlessness. This framework triggers the issues of code and item bolster to illuminate this issue we are utilizing diverse procedure.

I. C. Suherman et al [6] there are strategies that can be utilized to assess program advancement exertion; COCOMO II is one strategy that commonly utilized. Numerous analysts some time recently have been utilized calculation, such as Bat, Bee Colony, or MOPSO to extend COCOMO II estimation precision. We utilize Arbitrary Timberland Relapse as machine learning calculation to assess the effort with this framework able to diminish the less exertion for program advancement.

Marta Fernandez-Diego et al [7] The Effective effort of estimation models to facilitate the project planning of software development. This system shows a proper and systematic literature survey of different system which is developed by different authors.

[9] The proposed method is coordinates with a nonexclusive technique which makes a difference to assimilate the accessible past data and information. This will advance utilize it into spry estimation capability.

Proposed Work

The uncertainty about cost estimation is usually quite high, because of prediction of basic element size, cost drivers and other parameters. We used COCOMO model for cost estimation.

Prediction to the future: We aim to give general insights to identify some of the impacts that constrained environments have on the success of software development projects. Our aim is to encourage further research in cost estimation modelling and evaluations for such environments. The aim of the system is therefore to propose a statistical framework for comparative software cost estimation experiments concerning multiple prediction models. The statistical methodology is also based on an algorithmic procedure which is able to produce no overlapping clusters of prediction models, homogeneous with respect to their predictive performance. For the algorithmic approach we are using the Naïve Bayes algorithm to predict the profit.

Following figure 2 shows the classic model for Software cost estimation:

Figure 1: Flow diagram for Agile software development

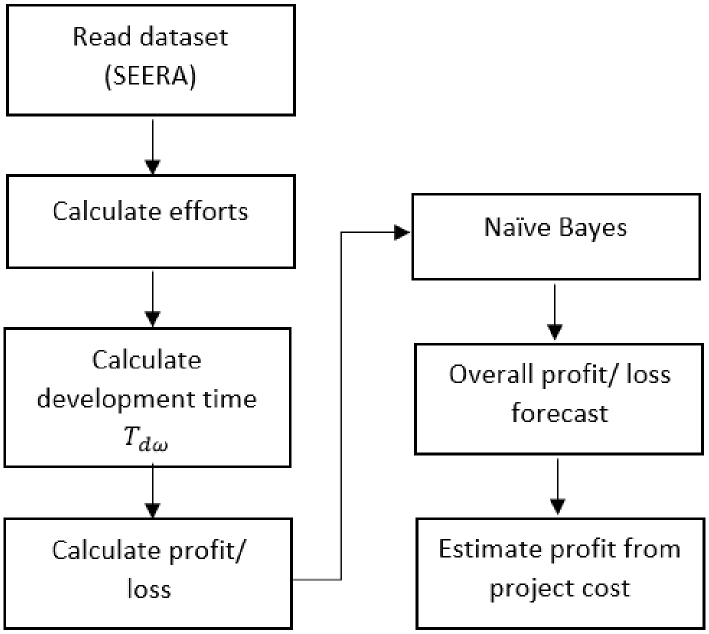

Read dataset (SEERA)
Calculate efforts
Calculate development time $T_{d\omega}$
Calculate profit/ loss
Naïve Bayes
Overall profit/ loss forecast
Estimate profit from project cost

Figure 1 represents overall flow of the agile software development. SEERA dataset will be given as an input in the proposed system. The system will calculate efforts in terms of actual efforts and estimated effort in the COCOMO model using formulas. Software cost estimation methods utilize historical datasets of previous projects to estimate project costs and effort. Now development time of the previous project is calculated to learn how much time a particular project requires to be complete. Based on the previous project cost, the system will calculate profit or loss. All these calculation is done in the COCOMO model, now these values are transferred to the Naïve Bayes algorithm for the prediction of new project's profit or loss.

Basic software cost estimation model:

For any new software project, it is necessary to know how much it will cost to develop and how much development time will it take. These estimates are needed before development is initiated, but how is this done? Several estimation procedures have been developed and are having the following attributes in common.

1. Project scope must be established in advanced.
2. Software metrics are used as a support from which evaluation is made.
3. The project is broken into small PCs which are estimated individually.

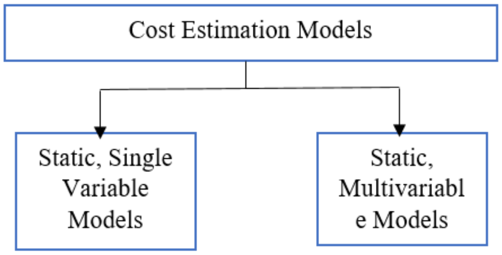
To achieve true cost & schedule estimate, several option arise.

4. Delay estimation
5. Used symbol decomposition techniques to generate project cost and schedule estimates.
6. Acquire one or more automated estimation tools.

Cost Estimation Models:

A model may be static or dynamic. In a static model, a single variable is taken as a key element for calculating cost and time. In a dynamic model, all variables are interdependent, and there is no basic variable.

Figure 2: Basic model of cost estimation



Static, Single Variable Models: When a model makes use of single variables to calculate desired values such as cost, time, efforts, etc. is said to be a single variable model. Effort is calculated using,

1. Estimated effort:

$$[Estimated\ duration\ \times (Dediacted\ Team\ Members + (Team\ size - Dediacted\ Team\ Members) \times 50\%)] \times (Daily\ Working\ Hours \times 22) \qquad (1)$$

2. Actual effort:

$$[Actual\ duration\ \times (Dediacted\ Team\ Members + (Team\ size - Dediacted\ Team\ Members) \times 50\%)] \times (Daily\ Working\ Hours \times 22) \qquad (2)$$

Profit or loss is calculated using the following formula,

$$\frac{(Contract\ price - Actual\ incurred\ costs)}{Contact\ price} \times 100 \qquad (3)$$

Static, Multivariable Models: These models are based on method (1), they depend on several variables describing various aspects of the software development environment. In some model, several variables are needed to describe the software development process, and selected equation combined these variables to give the estimate of time & cost. These models are called multivariable models. Duration of development is given by

$$D=4.1L^{0.36} \qquad (4)$$

COCOMO Model:

COCOMO as a collection of three variants, they are Basic model, Intermediate model, and Detailed model. Boehm described three development modes and Organic is for relatively simple projects, Semidetached is for relatively intermediate projects, embedded for a project developed under tight constraints. Constructive Cost Model (COCOMO) is widely used algorithmic software cost model. It was proposed by Boehm. 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).

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.
Model 3 (Detailed COCOMO Model): The detailed 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.

Naïve Bayes Algorithm:
Naive Bayes models are a group of extremely fast and simple classification algorithms that are often suitable for very high-dimensional datasets. Because they are so fast and have so few tunable parameters, they end up being very useful as a quick baseline for a classification problem. Naive Bayes classifiers are built on Bayesian classification methods. These rely on Bayles's theorem, which is an equation describing the relationship of conditional probabilities of statistical quantities. The Naive Bayes classifier A simple classifier, which in practice often performs surprisingly well, is the Naive Bayes classifier. This classifier basically learns the class-conditional probabilities $P(X_i = x_i|C = c_1)$ of each variable X_i given the class label cl. A new test case $(X_1 = x_1, \dots, X_n = x_n)$ is then classified by using Bayes' rule to compute the posterior probability of each class cl given the vector of observed variable values

$$P(C = c_1|X_1 = x_1, \dots, X_n) = \frac{P(C=C_1)P(X_1=x_1,\dots,X_n=x_n|C=c_1)}{P(X_1=x_1,\dots,X_n=x_n)} \qquad (5)$$

The simplifying assumption behind the Naive Bayes classifier then assumes that the variables are conditionally independent given the class label. Hence

$$P(X_1 = x_1, \dots, X_n = x_n|C = c_1) = \prod_{i=1}^{n} P(X_i = x_i|C = c_1) \qquad (6)$$

This assumption simplifies the estimation of the class-conditional probabilities from the training data. Notice that one does not estimate the denominator in Eq. (1) since it is independent of the class. Instead, one normalizes the nominator term $P(X_1 = x_1, \dots, X_n = x_n|C = c_1)$ to 1 over all classes. Naive Bayes classifiers are easy to construct since the structure is given apriori and no structure learning for the discrete variables and a normal or kernel density based method for continuous variables. phase is required. The probabilities $P(X_i = x_i|C = c_1)$ are estimated by using the frequency counts Figure 1(a) provides a graphical representation of a Naive Bayes classifier.
Thus, using the above mentioned formulas will predict the forecast of the upcoming new projects in the company in terms of profit or loss, estimated efforts and the development time. So that it provides recent diverse data that will allow researchers to compare the applicability of international methods to constrained environments and develop new techniques that are more suitable for these environments.

Seera Dataset Collection
In this paper we introduce the **SEERA** (**S**oftware engin**EER**ing in Sud**A**n) cost estimation dataset, a dataset of 120 software development projects representing 42 organizations in Sudan. The SEERA dataset contains 76 attributes and, unlike current cost estimation datasets, is augmented with metadata and the original raw data. The SEERA dataset fills the gap in cost estimation datasets with traditional cost attributes. The datasets projects represent constrained technical and

economic software development environments, thus providing the international software engineering research community with a recent and diverse dataset to evaluate the generalization of previous and future costing models. The SEERA dataset overcomes the current limitations in dataset quality and transparency by augmenting the cost estimation dataset (e.g. a COCOMO- style dataset from the original SEERA data).

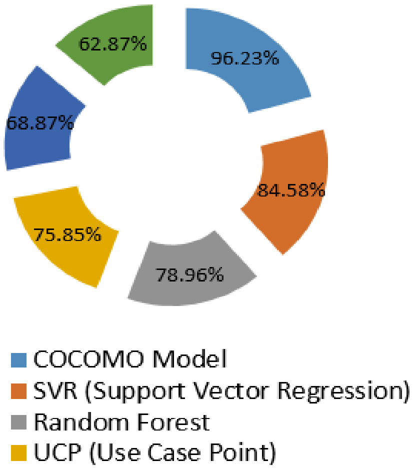
Results

Table 1: Software development accuracy with different methods.

Methods	Estimation Accuracy
COCOMO Model	98.23%
SVR (Support Vector Regression)	92.54%
Random Forest	89.76%
UCP (Use Case Point)	81.02%

Table 1 represents different methods which are used for cost and effort estimation for software development. Above all method shows the accuracy result for the estimation of cost and effort. These all methods are used in different papers which we are studied in this project. Different authors are shown the results of these methods.

Graph 1: Cost estimation accuracy for proposed and existing techniques



Above graph shows the cost estimation results accuracy for different techniques. We used COCOMO model in our system which shows the highest accuracy of 96.23% for the cost estimation of software development more than other techniques. Remaining techniques are also used for cost estimation but they have got less accuracy as compare to our used techniques of COCOMO model.

Conclusion:

This paper presents a systematic literature survey on cost & effort techniques of estimation for ASD (Agile Software Development) by using different soft computing techniques. Many articles and research are studied on cost & effort estimation details & an analytical representation which is shown by authors. The authors have founded so much less work in the field of effort & cost estimation of ASD projects with the use of soft computing techniques. we estimate the cost methods software of various kinds provided a comprehensive overview and description of these methods has advantages and disadvantages of this paper relevant reasons presents some of the false assumptions that a meaningful and reliable estimates of production, we want software project properties and their causal relationships improve our understanding of, to develop effective ways of measuring software complexity and well-arranged and carefully assess the costs. There is a rapid increase the percentage of software projects which are created for agile methodologies, so it is important to explore the estimation methods for projects. We used COCOMO model for the cost, time and effort estimation of ASD (Agile software development) The advantages of calculating using COCOMO II with this application are simple data that must be prepared by the user, layout of calculations with minimum wages, and a comprehensive presentation of calculation results. As well as the future work is concerned the authors will study the related article of latest technology & will represent them as a detailed survey.

References

1. D. S. Senevirathne and T. K. Wijayasiriwardhane, "Extending use-case point-based software effort estimation for Open Source freelance software development," 2020 International Research Conference on Smart Computing and Systems Engineering (SCSE), Colombo, Sri Lanka, 2020, pp. 188-194.
2. S. S. Ali, M. Shoaib Zafar and M. T. Saeed, "Effort Estimation Problems in Software Maintenance – A Survey," 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 2020, pp. 1-9.
3. M. Wang, Y. Ma, G. Li, W. Zhou and L. Chen, "Multi-Value Models for Allocation of Software Component Development Costs Based on Trustworthiness," in IEEE Access, vol. 8, pp. 122673-122684, 2020.
4. L. S. Nair and J. Swaminathan, "Towards Reduction of Software Maintenance Cost through Assignment of Critical Functionality Scores," 2020 5th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2020, pp. 199-204.
5. R. Sari Dewi and R. Sarno, "Software Effort Estimation Using Early COSMIC to Substitute Use Case Weight," 2020 International Seminar on Application for Technology of Information and Communication (iSemantic), Semarang, Indonesia, 2020, pp. 214-219.
6. I. C. Suherman, R. Sarno and Sholiq, "Implementation of Random Forest Regression for COCOMO II Effort Estimation," 2020 International Seminar on Application for Technology of Information and Communication (iSemantic), Semarang, Indonesia, 2020, pp. 476-481, doi: 10.1109/iSemantic50169.2020.9234269.
7. M. Fernandez-Diego, E. R. Mendez, F. Gonzalez-Ladrón-De-Guevara, S. Abraham and E. Insfran, "An Update on Effort Estimation in Agile Software Development: A Systematic Literature Review," in IEEE Access, vol. 8, pp. 166768-166800, 2020.
8. S. A. Idrus, W. Nur Hidayat and A. Hamdan, "NAYA as a Tool of Software Cost Automatic Analysis," 2020 4th International Conference on Vocational Education and Training (ICOVET), Malang, Indonesia, 2020, pp. 258-263.
9. https://www.google.com/search?q=what+are+the+soft+computing+techniques+for+software+cost+estimation&oq=what+are+the+soft+computing+techniques+for+software+cost+estimation&aqs=chrome.69i57.19046j0j7&sourceid=chrome&ie=UTF-8