International Research Journal of Computer Science (IRJCS) ISSN: 2393-9842
Issue 12, Volume 3 (December 2016) www.irjcs.com

A Survey Report on Software Effort Estimation models And techniques

M Pramod Kumar
Research Scholar
Krishna University
Asst.Prof in SVEC

Dr. M. Babu Reddy
Asst.Prof of Computer Science
Krishna University
Machilipatnam , Tadepalligudem

A Lakshmi Lavanya
Asst.Prof
Sri Vasavi Engg College
Tadepalligudem

Abstract-- Software Effort Estimation is a crucial factor in any software industry. There has been a lot of work done in the area of software effort estimation. As software grew in size and complexity, it is very difficult to precisely figure-out the cost of software development. The greatest risk of software industry was the fast changing nature of software development. Which leads difficult to develop parametric models that yield high efficiency for software development in all domains. This paper synopsizes several classes of Software Effort Estimation models and techniques. No single technique is best for all situations, and that a attentive comparison of the results of several approaches is most likely to produce pragmatic estimates. The use of personnel is measure as effort and defined as total time taken by development team associates to perform a given task. It is usually expressed in units such as man-day, man-month, and man-year. This value is important as it serves as basis for estimating other values relevant for software projects, like cost or total time required producing a software product. These paper overviews a research study comparing the different estimation techniques.

Keywords: Effort estimation, Software Size, fuzzy logic, ANN

1. INTRODUCTION

For any successful completion of project, Software Effort Estimation (SEE) is very important. It is to maintain and control if SEE is done in authentic way. Project Management is the process of planning and controlling the development as a system within a specified time frame at a minimum cost with the right functionality. Much software breakdown due to faulty project management practices. Therefore, it is important to learn different aspects of software project management. Key features of Project Management-

- ✓ *Project Scheduling*
- ✓ *Staffing*
- ✓ *Monitoring and control*
- ✓ *Project Estimation*
- ✓ *Risk Management*
- ✓ *Report generation*

All these Projects, Estimation is the most challenging task. Project estimation involves estimation of size, effort, cost, time, staffing. First, we calculate the size, from size estimation, we determine the required effort. From effort estimation, we can calculate product duration and cost. Software size estimation is important to determine the project effort. However, according to the last research reported by the Brazilian Ministry of Science and Technology-MCT, in 2001, only 29% of the companies accomplished size estimation and 45.7% accomplished software effort estimate. So that effort estimation has motivated considerable research during recent years. Effort Estimation: It is the process of predicting the effort required to developer maintain software product in person months. Many ways are available for categorizing estimation approaches. Most efficient categories are as follows1. Expert estimation: The quantification step, on the basis of judgmental process estimation is done.2. Formal estimation: the quantification step is based on mechanical processes, e.g., the use as a formula derived from historical data.3. Combination-Based estimation: This estimation approach deals with a judgmental or mechanical combination of estimates from different sources.

2. EFFORT ESTIMATION MODELS

There are two challenges in software development namely, unmanaged risks and inauthentic estimations of resources for a project. Kocaguneli et al. [1] has discussed about whether complex methods are needed for Software Effort Estimation(SEE). They characterize the essential content of SEE data that includes the least number of features and instances required to capture the information within SEE data. In case of less essential content, the contained information must be very brief and the value added of complex learning schemes must be minimal. The proposed QUICK methodcomputes the Euclidean distance between the instances and features of the SEE data and prunes the similar features and outliers. It assesses the reduced data by comparing predictions from a simple learner using the reduced data and CART using all data.

Rastogi et al. [2] has given a review of general techniques and models regarding effort estimation. The merits and risks of every technique are discussed. A single technique is not available and hence to produce realistic estimates, a hybrid of approaches is desirable. Pytel et al. [3] has designed two ad-hoc models for small and medium-sized enterprises to assess the feasibility, and to estimate the resources including time. Both models should be applied at the beginning of the project.

2.1 Analogy Based Effort Estimation Models:

Analogy-based Software development Effort Estimation (ASEE) techniques are drawing importance. The review studies on predicting software development effort have not examined the issues of ASEE techniques.
Wolverton [4] has dealt with the estimation by analogy and have described the similarities and differences of the existing software cost estimating techniques. Mukhopadhyay et al. [5] has also used analogy for SEE by retrieving the most similar cases. It is seen that the analogy based approach is more authentic and persistent than the function point and COCOMO models.

An analogy based approach for effort estimation is proposed by Shepperd and Schofield [6]. The projects are characterized in terms of features. The most similar featured projects are compared with the developed project. Similarity is the Euclidean distance in n-dimensional space where ‘n’ is the number of project features. The known effort values of the nearest neighbors to the currently developed project are used as the basis for the prediction. The process is automated using ANaloGy Estimation tool (ANGEL) and the performance is analyzed. The analogy based schemes exceedsalgorithmic models based on stepwise regression.

ANGEL proposed by Shepperd and Schofield [6] is an analogy based methodology. It is a non-proprietary tool, a form of non-parametric regression. Similarly, Bootstrap based Analogy Cost Estimation (BRACE) by Stamelos et al. [7] is an analogy-based tool that applies analogy based technique, and re-sampling methodology. It acts as a non-parametric bootstrap for calibration and aids in evaluating the model’s accuracy. As stated by various authors, Estimation by Analogy (EA) models offer better accuracy [6 - 11].Myrtveit and Stensrud [8] and Briand et al. [9] has given findings that contradict Shepperd's findings. They have shown that both EA and regression techniques improve the estimation accuracy, but EA does not outperform regression. Idri et al. [12] has proposed Fuzzy logic based EA model. The analogy estimation is adjusted based on fuzzy similarity between two software projects described only by ordinal data in the COCOMO dataset. This approach may not suit datasets that are structurally dissimilar to COCOMO dataset.

As stated by Mendes et al. [10, 11] and Shepperd and Schofield [6], EA performs better in contrast to the linear and stepwise regression models. Jorgensen et al. [13] has used regression towards the mean method to regulate EA. This method is appropriate for extreme analogues and inauthentic estimation models. The adjusted estimation is authentic than EA without adjustment. To improve EA, Mittas et al. [14] has used iterative re-sampling method. According to them, EA is closely related to formal nearest neighbor non-parametric regression. EA needs more number of sensed similarity methods [15]. The effort obtained by these similarity methods is not reusable without processing. The similarity methods are to be adjusted to make the retrieved effort more reasonable. GA was used to find the project distance and to adjust retrieved effort. From the results, it is evident that the adjusted similarity mechanism yields better accuracy than the traditional similarity distance. Analogy-based SEE based on similarity distances between every pair of projects is done. Adjusting effort based on the analogy-based software effort estimations yields better results as it uses three distance metrics. The proposed method is compatible with the widely used estimation models of ANN, CART and OLS.

Azzeh et al. [16] has developed a Fuzzy set theory and GRA based similarity measure for analogy-based estimation. The measure has the capability to deal with numerical and categorical attributes and two levels of similarity measures are defined namely, local and global measures. The performance of the measure is far better when compared to CBR, stepwise regression and ANN. The work by Idri et al. [17] classifies the ASEE studies and proposes a new modified technique based on five criteria namely, research approach, contribution type, techniques used in combination with ASEE methods, ASEE steps, and identifying publication channels and trends.

Further, the performance is analyzed in terms of estimation accuracy, accuracy comparison, estimation context, impact of the techniques used in combination with ASEE methods and ASEE tools. ASEE methods outperform the eight techniques and yield acceptable results when combined with Fuzzy Logic (FL) or Genetic Algorithms (GA).

3.2 COMPONENT BASED SIZE ESTIMATION MODELS:

Verner and Tate [1] have proposed a technique for estimating the number of LOC early in the software life-cycle. This method called Component Based Method (CBM) determines the sizes of the individual components or modules first and then adds the component sizes to get the overall system size. This approach generalizes the division in components by function point analysis. They have determined the type of every component by examining the characteristics of each type and looking for its predictors of size. Regression methods are applied to the independent predictor variables and the LoC to obtain estimation equations. This method finds the size of the components and selects the predictor variables that are available at the corresponding life-cycle phase for estimation. Verner and Tate have applied the method for two types of systems namely, business systems and systems programming applications.

In the literature, the Science Applications International Corporation (SAIC) model is the first model developed to estimate the effort of Component Based Software Development (CBSD) [2]. It takes into consideration the estimated cost, component licensing cost, number of licenses required, component training cost and glue code development cost. It ignores the effort involved in searching and selecting components. Further, it does not provide the details of determining the effort involved in glue code development .Another effort model that focuses on the volatility cost of components is proposed by Stutzke [4]. Component volatility is the frequency of releases of new versions of components. This model finds an estimate of the additional cost involved in using a given component with a significant volatility based on the estimated additional cost of using a component, component's volatility over system's life, architectural coupling of the component, interface size of the component, cost of screening the component along with the component with which it interfaces, and the cost of making changes to the components that have impact. Component volatility is the only factor that needs to be considered when predicting the effort of CBSD.

Ellis in 1943 [5] has proposed about 17 cost drivers for developing an effort model that predicts the effort involved in component integration. It takes into consideration the following factors namely, productivity, labor months, work units and a function to find the relationship between the size of glue code and ratings of cost drivers to work units. Function Point (FP) analysis was used by Albrecht and Gaffney [6] to estimate the glue code size. It is an application with a Graphical User Interface (GUI). In 1944, Aoyama [7] brought out four main differences between conventional software development and CBSD process models based on four factors namely, newly introduced component acquisition, compositional design, component integration processes and unit testing process. He proposed an economic model for CBSD, where unit process and unit product costs of processes for conventional software development and CBSD process models were taken intoaccount. Based on the results, it is evident that the CBSD approach is capable of reducing the total development cost by 1 -18% [8]. Nevertheless, testing a CBS requires more time and effort than a system developed using custom development.

3.3 NEURAL NETWORK (NN) BASED EFFORT ESTIMATION MODELS:

Many researchers like Jorgerson [18], Srinivasan and Fisher [19], Hughes [20], Wittig and Finnie [21], Samson et al. [22], Schofield [23], Seluca [24],Heiat [25] has applied Neural Networks (NNs) to estimate software development effort. The effort estimation is classified into four main categories namely, Expert judgment-based methods, Analogy based-methods, parametric model-based methods and Machine learning-based methods. An expert judgment-based method is based on the expert perception and experience gained [13], whereas Analogy based-methods identify one or more developed projects similar to the project currently being developed and compute the total estimated effort manually [26]. Parametric model-based methods rely mainly on historical data based equations. Effort is taken as function of parameters influencing effort [27]. Machine learning-based methods model the complex relationship between effort and effort drivers using Artificial Intelligence (AI) based techniques like Neural Networks (NN) and Fuzzy Logic [19]. It is found that though it shows outstanding performance in contrast to COCOMO and SLIM, the results are not that good than a statistical model derived from function points or a NN. The division of Kemmerer dataset for training and validation purposes is not clear. Further, it is found that the results are sensitive to the number of hidden units and layers. NN based effort estimation models learn from previous data, adapt to any organization and project context, can be updated over time and model complex relationships [28 - 30]. A SEEmethod is developed by Laqrichi [31] to provide realistic effort estimates based on the uncertainty in the effort estimation process. Neural Network based effort estimation model using bootstrap re-sampling technique is presented. The methodology generates a probability distribution depicting the effort estimates from which the prediction interval associated to a confidence level can be computed. The propounded technique offers better performance for International Software Benchmarking Standards Group dataset in contrast to the traditional effort estimation based on linear regression.

3.4 Fuzzy Based Effort Estimation Models:

Fuzzy logic offers a better mapping between input and output spaces [32]. The main properties of a fuzzy model are that it operates at a level of linguistic terms (fuzzy sets), represents and processes uncertainty [33]. Fuzzy set theory is a complete approach that deals with linguistic values like small, medium, average, or high [34]. Fuzzy model is best suited for software development effort estimation. Developing a precise mathematical model for the domain is challenging [35]. Metrics produce estimations of the real complexity. A set of natural rules describing the relation between software metrics and the effort estimation is vital.

Gray and MacDonell [36] have compared FLM with Linear Regression Models (LRMs) and Neural Networks (NNs). FLM is based on triangular membership functions. FLM yields better performance when compared to LRM and NN for the dataset from a Canadian thesis. A FLM based on trapezoidal membership functions is proposed by Idri et al. [37], wherein fuzzy logic is applied to the fifteen cost factors of COCOMO 29. The randomly generated dataset is compared with actual data of COCOMO 29. From the results, it is evident that the results of the FLM are mostly similar to those of COCOMO [29]. Idri et al. [34] has propounded an approach based on fuzzy logic named Fuzzy Analogy for COCOMO 29 dataset. Based on the accuracy and competence to deal with linguistic values, four techniques are ranked in the given order - Fuzzy Logic, Fuzzy intermediate COCOMO '29, Classical intermediate COCOMO'29 and Classical Analogy. Musflek et al. [38] has proposed f-COCOMO, a fuzzy model for COCOMO 29 to bring out the relationship between size fuzzy sets and effort fuzzy sets using triangular membership functions. They have concluded that fuzzy sets aid in enunciating the estimates by exploiting fuzzy numbers described by asymmetric membership functions.

A model combining Fuzzy Logic and NNs is proposed by Huang et al. [39] for the COCOMO dataset. FLM yields better performance when compared to NN. The FLM based on triangular membership function offers better interpretability by using the fuzzy rules. It combines the fuzzy rules, data and the traditional algorithmic model into one general framework. Ahmed et al. [40] has presented a FLM based on triangular membership functions. Randomly generated dataset and the one used for COCOMO 29 are used for validating the FLM. FLM shows slightly better performance in contrast to COCOMO equations. There are chances for improvement when more knowledge is added to the dataset. Fuzzy regression techniques based on fuzzification of input values are explored by Crespo et al. [41] for COCOMO-29 database. Fuzzy regression model is better than the existing basic estimation models. In 2004, Reformat et al. [42] has designed an estimation model based on fuzzy neural network to compute the development effort in a medical information system. The dataset is divided in three subsets, wherein one is used for validating the model. For linguistic data, has Xu and Khoshgoftaar [43] propounded a fuzzy identification cost estimation modelling technique that generates fuzzy membership functions and rules. It is an advanced fuzzy logic technique that integrates fuzzy clustering, space projection, fuzzy inference and defuzzification. The proposed system is applied for all three COCOMO 29 models - basic, intermediate and detailed. From the results, it is evident that the fuzzy identification model is better in terms of cost than the existing COCOMO models.

As the attributes are measured based on human judgment, the measurements are vague and imprecise. Hence, the uncertainty in software attribute measurement has significant impact on estimation accuracy. To overcome this challenge, a formal EA model based on the integration of Fuzzy set theory with Grey Relational Analysis (GRA) is proposed by Azzeh et al. [16]. Fuzzy logic is employed to reduce the uncertainty, whereas GRA is used to assess the similarity between two tuples. Since all the features need not be continuous and may have nominal and ordinal scale type, aggregating the different forms of similarity measures will lead to increase in the uncertainty in the similarity degree. GRA is employed to reduce the uncertainty in the distance measures for both continuous and categorical features. These techniques are suitable for complex relationships between effort and other effort drivers. The performance of the proposed system is better when compared to Case Based Reasoning (CBR), Multiple Linear Regression (MLR) and Artificial Neural Network (ANN) methods. Knowledge-based Mamdani max-min fuzzy expert system is applied for estimating the pressure between the contact area and contact is described by Taghavifar and Mardani [44]. Two paramount tire parameters namely, wheel load and tire inflation pressure are the input variables for the proposed model with five membership functions each. A set of fuzzy if-then rules are used in accordance with fuzzy logic principles and an intelligent predicting model based on centriode method is developed at defuzzification stage. The results show that FES offers better performance in terms of diverse statistical criteria

3.5 Grey Relational Analysis (GRA) Based Effort Estimation Models:

Idri et al. [12] has shown that replacing the categorical features including nominal or ordinal values by numerical values increase the uncertainty in estimation. Fuzzy set theory and GRA are employed to decrease the imprecision in the distance between two projects containing continuous and categorical values. Song et al. [45] has proposed a SEE method based on Grey Relational Analysis (GRA) called GRACE. GRA is used to select an optimal feature set based on the similarity degree between dependent variable and other variables. The variables which are very much similar form the optimal feature set. Continuous variables are preferred than categorical. GRA derives new estimate by finding the case closest to the current case on all effort drivers. This model yields better performance when compared to other prediction models like NNS, decision tree and stepwise regression.

Huang et al. [46] has integrated GRA with GAs to improve software effort estimation. GA is used to adjust the weight factor associated with weighted GRA. GA necessitates many parameters and assumptions to be setup before finding appropriate weights. Yet, the performance of the proposed system for well-established datasets has shown that the weighted GRA with GAs improves the accuracy of software effort estimation. Hsu and Huang [47] have designed diverse weighted GRA models for SEElike distance-based weight, linear weight, non-linear weight, maximal weight and correlative weight. According to them, weighted GRA yields better results when compared to the non-weighted GRA. Linearly weighted GRA outperforms other weighted GRA.

3.6 Phase-Level Based Effort Estimation Models:

SEEduring early stages of software development is a crucial task as the data collected during the early stages of a software development lifecycle is imprecise and uncertain. Authentic estimates can't be obtained. Analogy-based estimation is hardly used during the early stage of a project due to the uncertainty in attribute measurement and data availability. Kulkarni et al. [48] has described phase-based size and effort prediction for ADA systems, wherein the measures of the outputs of one phase are provided as the predictive inputs to the next. This system relies on object measures rather than recorded effort values.

Some of the existing algorithmic models were fuzzified so as to enable them to handle uncertainties and imprecision problems. Fei and Liu [49] dealt with the fuzziness of several aspects of COCOMO model. They observed that an authentic estimate of delivered source instruction could not be made before commencing the project. Case Point and Function Point models are widely used in the early stage estimation. As they are environment dependent models, they require calibration and are affected by the uncertainty and incompleteness of the dataset used, they face some challenges. These models depend on the input size, thus demanding reliable measurement [32, 50, 51].

The effort data recorded for completed project tasks are used to predict the effort needed for subsequent activities in [52]. Data collected from 16 projects undertaken by a single organization over a period of 18 months was taken into consideration. The proportions of effort for each development activity cannot be predicted. Simple linear regression combined with the managers' estimates provided better estimation and increased the predictive accuracy. Data of previous phase efforts could be used as a supplement to the estimation process and improve the management of subsequent tasks.

As the available data is often imprecise and vague, uncertainty at the early stage is a universal problem in estimation of software effort. Experienced software estimators are essential to translate the set of requirements into use cases, actors and scenarios [53]. Machine learning based estimation techniques such as analogy-based estimation and NNs are hardly used at the initial stages of software development due to uncertainty in determining the values of attributes.

The algorithmic effort prediction models are not able to deal with the uncertainties and imprecision present in software projects in the early stages of the development life cycle. An adaptive fuzzy logic framework for software effort prediction is presented by Ahmed et al. [54]. The training and adaptation algorithms in the proposed framework bears fuzziness, describes prediction rationale by rules, incorporates expert knowledge, offers transparency in the prediction system, and adapts to new environments as new data becomes available. The system was validated for artificial datasets as well as the COCOMO public database. In [55], analogy-based estimation is combined with Fuzzy numbers to improve the performance of software project effort estimation during the early stages of a software development lifecycle. Software project similarity measure and an adaptation technique based on Fuzzy numbers are proposed. Empirical evaluations with Jack-knifing procedure is carried out using five benchmark data sets of software projects, namely, ISBSG, Desharnais, Kemerer, Albrecht and COCOMO, and the performance is analyzed. The results are compared to the methods involving CBR and stepwise regression. In all datasets, from the empirical evaluations, it is evident that the proposed similarity measure and adaptation techniques method significantly improves the performance of analogy-based estimation during the early stages of software development. The proposed method performs better in contrast to CBR and stepwise regression.

3.7 Case Based Reasoning Effort Estimation Models:

On the other hand, Mendes et al. [56, 57] has examined the use of CBR and adaptation rules on the data collected from web hypermedia projects. From the results, it is evident that the adaptation rules are not significant as they do not contribute to better estimation.

3.8 Empirical Effort Estimation Models:

The empirical work done by Ohlsson and Wohlin [58] is similar to the one used by Kulkarni et al. [48]. They have used phase-based data to perform predictions for the subsequent phase. They have used artefact measures as predictive model inputs. These measures did not correlate particularly with effort, yet they provided a pictorial view of a project's progress and gave an idea for re-plan.

Another empirical work done by Rainer and Shepperd [59] has provided a longitudinal case study of planning and effort expenditure at IBM. The need for the organisation to continually re-plan is the fact that the initial schedule was so unrealistic. Re-planning aids in the success of projects. Jørgensen and Sjøberg [60] has performed an empirical analysis on the impact of estimates on the effort expended. It is found that the estimates made early in the software process has a significance, even if they are found to be incorrect as the in the ensuing processes.

3.9 REGRESSION BASED EFFORT ESTIMATION MODELS:

Regression analysis generates equations to predict effort for software development using methods like fuzzy logic. Several algorithmic models are available in the literature. General form of linear regression equation is proposed by Kok et al. [61], while a group of non-linear regression equations are presented by Boehm [26] in COCOMO 29 and COCOMO II [62]. An Albus multilayer perceptron is used to predict software effort in [22] for Boehm's COCOMO dataset. Linear regression is compared with NN based approach for the COCOMO dataset. Both the approaches do not provide better results. In Briand and Wieczorek[63], a relationship between effort and one or more characteristic of a project is presented. The software size is taken as the cost determinant.

To improve the accuracy of effort estimation in the single regression model, several data partitioning based studies on deriving multiple regression models are developed by Cuadrado-Gallego et al. [64], Cuadrado-Gallego et al. [65] and Aroba et al. [67]. These models overcome the common shortcomings like poor model fitting and low accuracy of effort estimation in datasets of heterogeneous projects. An approach for generating multiple regression models by clustering using Expectation-Maximization (EM) algorithm is proposed by Cuadrado-Gallego et al. [64, 65]. Based on the experimental results validated with the ISBSG (Release 8) dataset, the accuracy of effort estimation by the multiple regression models is better when compared to the single model. Parametric software cost estimation models based on the historical software projects databases involve mathematical relations and are useful in estimating the effort and time required to develop a software product. Heterogeneous projects are considered and a single parametric model for a range of diverging project sizes and characteristics is not available. Segmented models are used in which several models are combined into one which gives the estimates depending on the concrete characteristic of the inputs. A given project can belong to several segments with different degrees of fuzziness.

An approach that generates multi-standard LSR models based on fuzzy clustering is proposed by Aroba et al. [67]. The above mentioned problems are addressed using a segmented model based on fuzzy clusters of the project space. Fuzzy clustering aids in obtaining different mathematical models for each cluster and also allow the items of a project to contribute to more than one cluster, while preserving constant time execution of the estimation process. Fuzzy clustering generates different LSR models for each cluster. The data points are contained in more than one cluster with different degrees of fuzziness. The final effort estimate is derived from the membership values of each data point used as a weight for each model. The proposed approach is validated for the ISBSG (Release 8) dataset, and the results are found to be better than the single model. The number of clusters is increased to find better estimation results

López-Martín [67] has compared Fuzzy Logic Models (FLM) with Linear Regression Model (LRM). The evaluation criterion is based on the Magnitude of Error Relative to the estimate (MER) and Mean of MER (MMER). From the programs developed, three FLMs were generated to estimate the effort. FLM and LRM offer similar performance. Least Squares Regression (LSR) is the most commonly used SEEmethod. LSR model is affected by the data distribution. For scattered dataset, the model usually shows poor performance. Data partitioning-based approaches are considered to be better when compared to the clustering-based approaches. Seo et al. [68], a new data partitioning-based approach is proposed to achieve more authentic and stable effort estimation using Least Squares Regression (LSR). This approach provides an effort prediction interval that is useful in determining the uncertainty of the estimates. The proposed approach is compared with the basic LSR approach and clustering-based approaches based on industrial datasets.

3.10 CLASS POINT AND USE CASE POINT (UCP) BASED EFFORT ESTIMATION MODEL:

Satapathy et al. [69] has computed the effort taken in software development using class point approach. To obtain better accuracy, the effort parameters are optimized using adaptive regression based multi-layer perceptron technique of Artificial NN (ANN). The software effort estimations using multi-layer perceptron and Radial Basis Function Network (RBFN) are compared. Use Case Point (UCP) method is proposed to estimate software development effort in the early stages of software development. UCP is the count of the number of actors and transactions involved in use case models. Several tools are developed to assist in calculating UCP. The actors and use cases are extracted and the complexity classification is performed manually. Kusumoto et al. [70] has developed an automatic use case measurement tool, called U-EST. It automatically classifies the complexity of actors and use cases from use case model. U-EST is applied to actual use case models and the difference between the values produced by the tool and the specialist are examined. UCPs offer similar values as the ones produced by the specialists.

Mohagheghi et al. [71] has propounded an effort estimation method based on use cases, the Use Case Points (UCPs) method. The original method is based on incremental development and evaluated on a large industrial system with modification of software from the previous release.

The elements of the original method including the complexity assessment of actors and use cases, and handling of non-functional requirements and team factors that affects effort are modified. Two elements that include counting both the modified actors and transactions of use cases, and effort estimation for secondary changes of software not reflected in use cases were added to the incremental method. The proposed scheme was extended to cover all development effort in a very large project. It was calibrated using data from one release. The estimate produced for the successive release was only 17% lower than the actual effort. This study identified factors affecting effort on large projects with incremental development and showed how these factors can be calibrated for a specific context to produce relatively authentic estimates. There is a growing interest in SEEbased on use cases. In [72], Anda et al. has proposed the use case points method inspired by function points analysis. This work takes the functionalities and processes of four companies. They developed equivalent functionality, but their development processes varied, ranging from a light, code-and-fix process with limited emphasis on code quality, to a heavy process with considerable emphasis on analysis, design and code quality. The effort estimate of the proposed model based on the use case points method was close to the actual effort of the one with the lightest development process. From the results, it is evident that the use case points method needs modification to better handle effort related to the development process and the quality of the code.

Costagliola et al. [73] has presented a Function Point (FP)-like approach, named class point to estimate the size of object-oriented products. Two measures are proposed, theoretically validated to see that the renowned properties for estimating size measures are satisfied. An empirical validation is also performed initially to assess the usefulness and effectiveness of the proposed measures and to predict the development effort of object-oriented systems. The performance is compared with several other size measures. Zivkovicet et al. [74] has proposed a unified mapping of Use case Modeling Language (UML) models into function points. The mapping is formally described to enable the automation of the counting procedure. Three estimation levels that correspond to the different abstraction levels of the system and influence the estimate's accuracy is defined. The model considers a small dataset and it is seen that the accuracy increases with each subsequent abstraction level. Changes proposed to the FPA complexity tables for transactional functions are also proposed so as to measure the characteristics of object-oriented software. In the object-oriented framework, traditional methods and metrics were extended to help managers in this activity. Use Case Points (UCP) considers functional aspects of the Use Case (UC) model, widely used in most organizations in the early phases of the development. Nevertheless, UCP presents some limitations related to the granularity of the UC. To overcome these limitations, Braz and Vergilio [75] has introduced two metrics based on UCs namely; Use case Size Points (USPs) and Fuzzy Use case Size Points (FUSPs). USP considers the internal structures of the UC and captures the functionality, while FUSP considers concepts of the fuzzy set theory to create gradual classifications that deals with uncertainty.

Class points are recognized to estimate the size of object Oriented (OO) products and to directly predict the effort, cost and duration of the software projects. Many estimation models in the literature are based on regression techniques. NNs are used to estimate the development effort of OO systems using class points by Kanmani et al. [76]. Class points are used as the independent variables and development effort is taken as the dependent variable. From the results, it is evident that the estimation accuracy is higher in NNs in contrast to the regression model. FL aids in mapping the input space to the output space. In another paper, Kanmani et al. [77] has used Fuzzy Subtractive Clustering and ANNs to estimate the development effort of OO systems using class points. As in the former work, the proposed model also uses class points as an independent variable and development effort as the dependent variable. The estimation accuracy is higher in FL when compared to the model based on NNs. Ochodek et al. [78] has investigated the construction of Use Case Points (UCP) to find possible ways of simplifying it. A cross-validation procedure has been used to compare the accuracy of the different variants of UCP. Further, factor analysis has been performed to investigate the possibility of reducing the number of adjustment factors. Two variants of UCP – with and without Unadjusted Actor Weights (UAW) were developed. They provided comparable prediction accuracy and only an negligible impact of the adjustment factors on the accuracy of UCP was observed. The variants of UCP calculated based on steps were slightly more authentic than the variants calculated based on transactions. To conclude, the UCP method could be simplified by ignoring UAW. UCP can be calculated based on steps instead of transactions or by counting the total number of steps in use cases. SEE in the early stages of the software life cycle is done to derive the required cost and schedule for a project. In the requirements phase, if software estimation is conducted, the available information is generally imprecise or incomplete. Nassif et al. [79] has proposed a regression model for SEEbased on UCP model. A Sugeno Fuzzy Inference System (FIS) approach is applied on this model to improve the estimation.

CONCLUSION

In this paper we have surveyed about 80 research papers from literature for software effort estimation. Authentic estimation of software development effort is a very difficult job. Both under estimation as well as over estimation can lead to serious consequences. After several years research, there exist many software cost estimation methods like, estimation byanalogy, algorithmic methods, expert judgment method, bottom-up method and top-down method. As weaknesses and strengths of these methods are often complimentary, we cannot say a method is better or worse than the other. Before estimation of projects, understanding the strengths and weaknesses of every method is important. So it's very important to find a technique which can yield authentic results for software effort estimation.

REFERENCES

[1]. Kocaguneli, E., Menzies, T., Keung, J., Cok, D., and Madachy, R, "Active learning and effort estimation: Finding the essential content of SEEdata", IEEE Transactions on Software Engineering, pp. 1040-1053, Vol. 39(8), ISSN: 0098-5589, 2013.

[2]. Rastogi, H., Dhankhar, S., and Kakkar, M., "A survey on SEEtechniques", The Next Generation Information Technology Summit (Confluence) 2014 5th International Conference, pp. 826-830, ISSN: 978-1-4799-4237-4, 2014, September.

[3]. Pytel, P., Hossian, A., Britos, P., and García-Martínez, R. "Feasibility and effort estimation models for medium and small size information mining projects", Information Systems, pp. 1-14, Vol. 47. 2015..

[4]. Wolverton, R. W. , "The cost of developing large-scale software", IEEE Transactions on Computers, pp. 615-636, Vol.100(6) , ISSN: 0018-9340 , 1974.

[5]. Mukhopadhyay, T., Vicinanza, S. S., &Prietula, M. J. "Examining the feasibility of a case-based reasoning model for software effort estimation", MIS quarterly, pp. 155-171, Vol.16(2), 1992.

[6]. Shepperd, M., and Schofield, C. "Estimating software project effort using analogies", IEEE Transactions on Software Engineering, pp. 736-743, Vol.23(11), ISSN: 0098-5589, 1997.

[7]. Stamelos, I., Angelis, L., and Sakellaris, E. "BRACE: BootstRap based Analogy Cost Estimation: Automated support for an enhanced effort prediction method", Proceedings 12th European Software Control and Metrics Conference (ESCOM'2001), pp. 17-23, 2001.

[8]. Myrtveit, I., and Stensrud, E., "A controlled experiment to assess the benefits of estimating with analogy and regression models", IEEE Transactions on Software Engineering, pp. 510-525, Vol.25(4), ISSN: 0098-5589, 1999.

[9]. Briand, L. C., Langley, T., and Wieczorek, I , "A replicated assessment and comparison of common software cost Modeling techniques", Proceedings of the 22nd international conference on Software engineering, pp. 377-386, ISBN:1-58113-206-9, 2000.

[10]. Mendes, E., Mosley, N., and Counsell, S, "A replicated assessment of the use of adaptation rules to improve Web cost estimation", International Symposium on Empirical Software Engineering ISESE, pp. 100-109, ISSN: 0-7695-2002-2, 2003.

[11]. Mendes, E., Watson, I., Triggs, C., Mosley, N., &Counsell, S. "A comparative study of cost estimation models for web hypermedia applications", Empirical Software Engineering, pp.163-196, Vol. 8(2), ISSN: 1382-3256, 2003.

[12]. Idri, A., Abran, A., and Khoshgoftaar, T. M, "Fuzzy analogy: A new approach for software cost estimation", In International Workshop on Software Measurement, pp. 28-29, 2001, August.

[13]. Jorgensen, M. "A review of studies on expert estimation of software development effort", Journal of Systems and Software, pp. 37-60, Vol.70(1), 2004.

[14]. Mittas, N., Athanasiades, M., and Angelis, L., "Improving analogy-based software cost estimation by a resampling method" Information and Software Technology, pp. 221-230, Vol. 50(3), 2008.

[15]. Chiu, N. H., and Huang, S. J., "The adjusted analogy-based SEEbased on similarity distances", Journal of Systems and Software, pp. 628-640, Vol. 80(4), 2007.

[16]. Azzeh, M., Neagu, D., and Cowling, P. I, "Fuzzy grey relational analysis for software effort estimation", Empirical Software Engineering, pp.60-90, Vol.15(1), ISSN: 1382-3256, 2010.

[17]. Idri, A., AzzahraAmazal, F., and Abran, A, "Analogy-based software development effort estimation: A systematic mapping and review", Information and Software Technology, pp. 206-230, Vol.58, 2015.

[18]. Jorgensen, M. "Experience with the accuracy of software maintenance task effort prediction models", IEEE Transactions on Software Engineering, pp. 674-681, Vol. 21(8), ISSN: 0098-5589, 1995.

[19]. Rastogi, H., Dhankhar, S., and Kakkar, M., "A survey on SEEtechniques", The Next Generation Information Technology Summit (Confluence) 2014 5th International Conference, pp. 826-830, ISSN: 978-1-4799-4237-4, 2014, September.

[20]. Pytel, P., Hossian, A., Britos, P., and García-Martínez, R. "Feasibility and effort estimation models for medium and small size information mining projects", Information Systems, pp. 1-14, Vol. 47. 2015.

[21]. Wolverton, R. W. , "The cost of developing large-scale software", IEEE Transactions on Computers, pp. 615-636, Vol.100(6) , ISSN: 0018-9340 , 1974.

[22]. Mukhopadhyay, T., Vicinanza, S. S., &Prietula, M. J. "Examining the feasibility of a case-based reasoning model for software effort estimation", MIS quarterly, pp. 155-171, Vol.16(2), 1992.

[23]. Shepperd, M., and Schofield, C. "Estimating software project effort using analogies", IEEE Transactions on Software Engineering, pp. 736-743, Vol.23(11), ISSN: 0098-5589, 1997.

[24]. Stamelos, I., Angelis, L., and Sakellaris, E. "BRACE: BootstRap based Analogy Cost Estimation: Automated support for an enhanced effort prediction method", Proceedings 12th European Software Control and Metrics Conference (ESCOM'2001), pp. 17-23, 2001.

[25]. Myrtveit, I., and Stensrud, E., "A controlled experiment to assess the benefits of estimating with analogy and regression models", IEEE Transactions on Software Engineering, pp. 510-525, Vol.25(4), ISSN: 0098-5589, 1999.

[26]. Briand, L. C., Langley, T., and Wieczorek, I , "A replicated assessment and comparison of common software cost Modeling techniques", Proceedings of the 22nd international conference on Software engineering, pp. 377-386, ISBN:1-58113-206-9, 2000.
[27]. Kirsopp, C., and Shepperd, M., "Case and feature subset selection in case-based software project effort prediction", Research and Development in Intelligent Systems XIX , pp. 61-74, ISBN:978-1-85233-674-5, 2003.
[28]. Auer, M., and Biffl, S. "Increasing the accuracy and reliability of analogy-based cost estimation with extensive project feature dimension weighting", Proceedings on International Symposium on Empirical Software Engineering ISESE'04, pp. 147-155, ISSN: 0-7695-2165-7, 2004.
[29]. Keung, J., and Kitchenham, B, " Experiments with analogy-x for software cost estimation", Software Engineering, 2008. ASWEC 2008. 19th Australian Conference, pp. 229-238, ISBN: 978-0-7695-3100-7, 2008 March.
[30]. Mendes, E., Mosley, N., and Counsell, S, "A replicated assessment of the use of adaptation rules to improve Web cost estimation", International Symposium on Empirical Software Engineering ISESE, pp. 100-109, ISSN: 0-7695-2002-2, 2003.
[31]. Mendes, E., Watson, I., Triggs, C., Mosley, N., &Counsell, S. "A comparative study of cost estimation models for web hypermedia applications", Empirical Software Engineering, pp.163-196, Vol. 8(2), ISSN: 1382-3256, 2003.
[32]. Idri, A., Abran, A., and Khoshgoftaar, T. M, "Fuzzy analogy: A new approach for software cost estimation", In International Workshop on Software Measurement, pp. 28-29, 2001, August.
[33]. Jorgensen, M. "A review of studies on expert estimation of software development effort", Journal of Systems and Software, pp. 37-60, Vol.70(1), 2004
[34]. Mittas, N., Athanasiades, M., and Angelis, L., "Improving analogy-based software cost estimation by a resampling method" Information and Software Technology, pp. 221-230, Vol. 50(3), 2008.
[35]. Chiu, N. H., and Huang, S. J., "The adjusted analogy-based SEEbased on similarity distances", Journal of Systems and Software, pp. 628-640, Vol. 80(4), 2007.
[36]. Azzeh, M., Neagu, D., and Cowling, P. I, "Fuzzy grey relational analysis for software effort estimation", Empirical Software Engineering, pp.60-90, Vol.15(1), ISSN: 1382-3256, 2010.
[37]. Idri, A., AzzahraAmazal, F., and Abran, A, "Analogy-based software development effort estimation: A systematic mapping and review", Information and Software Technology, pp. 206-230, Vol.58, 2015.
[38]. Jorgensen, M. "Experience with the accuracy of software maintenance task effort prediction models", IEEE Transactions on Software Engineering, pp. 674-681, Vol. 21(8), ISSN: 0098-5589, 1995.
[39]. Srinivasan, K., and Fisher, D , "Machine learning approaches to estimating software development effort", IEEE Transactions on Software Engineering, , pp. 126-137, Vol. 21(2), ISSN: 0098-5589, 1995.
[40]. Hughes, R. T. "An evaluation of machine learning techniques for software effort estimation" University of Brighton, pp. 1-15, ISBN: 978-3-642-32340-9, 1996.
[41]. Wittig, G., and Finnie, G, "Estimating software development effort with connectionist models", Information and Software Technology", pp.469-476, Vol.39(7), 1997.
[42]. Samson, B., Ellison, D., and Dugard, P, " Software cost estimation using an Albus perceptron (CMAC)", Information and Software Technology, pp.55-60, Vol.39(1), 1997.
[43]. Schofield, C. "Non-algorithmic effort estimation techniques". ESERG, TR98-01, 1998.
[44]. Serluca, C, "An investigation into SEEusing a back propagation neural network", Bournemouth University, ISBN: 1605667676 1995.
[45]. Heiat, A, "Comparison of artificial neural network and regression models for estimating software development effort", Information and software Technology, pp.911-922, Vol.44(15), 2002.
[46]. Shepperd, M., Schofield, C., and Kitchenham, B, "Effort estimation using analogy", Proceedings of the18th International Conference on Software Engineering, pp. 170-178, ISBN:0-8186-7246-3, 1996.
[47]. Boehm B, "Software Cost Estimation with Cocomo II", Prentice-Hall, ISBN: 0130266922, 2000.
[48]. Park, H. and Baek, S, "An empirical validation of a neural network model for software effort estimation", Expert Systems with Applications, pp. 929-937, Vol. 35(3), 2008.
[49]. Idri, A., Zakrani, A., &Zahi, "A. Design of radial basis function neural networks for software effort estimation", IJCSI International Journal of Computer Science Issues, pp.11–17, Vol.7(4), ISSN: 1694-0814, 2010.
[50]. Setiono, R., Dejaeger, K.,Verbeke,W., Martens,D., and Baesens, B, "Software Effort Prediction Using Regression Rule Extraction from Neural Networks", pp.45-52, ISSN: 1082-3409, 2010.
[51]. Laqrichi, S., Marmier, F., Gourc, D., and Nevoux, J., "Integrating uncertainty in SEEusing Bootstrap based Neural Networks", IFAC-Papers online, pp.954-959, Vol.48(3), 2015.
[52]. Zadeh, L.A, "From computing with numbers to computing with words - from manipulation of measurements to manipulation of perceptions", IEEE Transactions on Circuits and Systems–I: Fundamental Theory and Applications, Vol.45 (1), pp.105-119, ISBN:978-3-662-00323-7 1999.
[53]. Pedrycz, W., Gomide, F.,"An Introduction to Fuzzy Sets", The MIT Press, ISBN: 8120346998, 1998.
[54]. Idri, A., Abran, A., Khoshgoftaar, " T. Estimating Software Project Effort by Analogy Based on Linguistic Values", Eight IEEE Symposium on Software Metrics, ISBN: 8132216024, 2002.

[55]. Lewis, J.P, “Large limits to software estimation”, ACM Software Engineering Notes, Vol.26 (4), pp.54–59, 2001.
[56]. Gray, A.R., MacDonell, S.G, “Applications of Fuzzy Logic to Software Metric Models for Development Effort Estimation”, Proceedings of NAFIPS, pp. 394 – 399, ISBN: 0-7803-4078-7, 1997.
[57]. Idri, A., Abran, A., Kjiri, L.,“ COCOMO Cost Model Using Fuzzy Logic”, 7th International Conference on Fuzzy Theory and Technology, Vol.5 ISBN: 978-81-920575,2000.
[58]. Musflek, P., Pedrycz, W., Succi, G., Reformat, M, “Software cost estimation with fuzzy models”, Applied Computing Review, pp.24-29, vol. 8 (2), 2000.
[59]. Huang, X., Capretz. L.F., Ren, J., Ho, D.A, “Neuro-Fuzzy Model for Software Cost Estimation”, Proceedings of the Third International Conference on Quality Software, pp. 126-133, ISSN: 0-7695-2015-4, 2003.
[60]. Ahmed, M.A., Saliu, M.O., AlGhamdi, J., “Adaptive Fuzzy Logic-Based Framework for Software Development Effort Prediction”, Information and Software Technology. Elsevier, pp. 31-48, Vol. 47(1), 2004.
[61]. Crespo, F.J., Sicicila, M.A., Cuadrado, J.J., “On the use of fuzzy regression in parametric software estimation models: integrating imprecision in COCOMO cost drivers”, WSEAS Transactions on Systems, PP. 129-137, Vol.41(2), ISBN: 978-3-540-32780-6, 2004.
[62]. Reformat, M., Pedrycz, W., Pizzi, N., “Building a Software Experience Factory Using Granular-Based Models. Fuzzy Sets and Systems” Elsevier. 2004.
[63]. Xu, Z., Khoshgoftaar, T.M., “Identification of fuzzy models of software cost estimation”, Elsevier Fuzzy Sets and Systems, Vol.145 (1), pp.141- 163, 2004.
[64]. Taghavifar, H., and Mardani, “A. Fuzzy logic system based prediction effort: A case study on the effects of tire parameters on contact area and contact pressure”, Applied Soft Computing, pp.390-396, Vol. 14, 2014.
[65]. Song, Q., Shepperd, M., and Mair, C. , “Using grey relational analysis to predict software effort with small data sets”, Software Metrics, 11th IEEE International Symposium, pp. 10-35, ISSN: 1530-1435, September 2005.
[66]. Huang, S. J., Chiu, N. H., and Chen, L. W., “ Integration of the grey relational analysis with genetic algorithm for software effort estimation”, European Journal of Operational Research, pp. 898-909, Vol.188(3), 2008.
[67]. Hsu, C. J., and Huang, C. Y., “ Improving effort estimation accuracy by weighted grey relational analysis during software development” , Software Engineering Conference, 2007. APSEC 2007. 14th Asia-Pacific, pp. 534-541, ISSN: 1530-1362, 2007, December.
[68]. Kulkarni, A., Greenspan, J. B., Kriegman, D., Logan, J. J., and Roth, T. D. , “A generic technique for developing a software sizing and effort estimation model”, In Computer Software and Applications Conference, 1988. COMPSAC 88 Proceedings, Twelfth International, pp. 155-161, ISBN: 0-8186-0873-0, 1988 October.
[69]. Fei, Z., and Liu, X. “f-COCOMO: fuzzy constructive cost model in software engineering”, In IEEE International Conference on Fuzzy Systems, pp. 331-337,ISSN: 0-7803-0236-2, 1992 March.
[70]. Azzeh, M., Neagu, D., and Cowling, P, “ Fuzzy Feature subset Selection for Software Effort Estimation”, International workshop on software predictors PROMISE'08 , pp. 71-78, 2008.
[71]. Azzeh, M., Neagu, D., and Cowling, P, “Software project similarity measurement based on fuzzy C-means - Making Globally Distributed Software Development a Success Story” Springer Berlin Heidelberg, pp. Pp. 123-134, ISBN:3-540-79587-1 978-3-540-79587-2, 2008.
[72]. MacDonell, S. G., and Shepperd, M. J., “Using prior-phase effort records for re-estimation during software projects”, Software Metrics Symposium, Proceedings of Ninth International, pp. 73-86, ISSN: 1530-1435 , September 2003.
[73]. Pfleeger, S. L., Wu, F., and Lewis, R., “ Software cost estimation and sizing methods: issues, and guidelines”, Rand Corporation, Vol. 269, 2005.
[74]. Ahmed, M. A., Saliu, M. O., and AlGhamdi, J., “ Adaptive fuzzy logic-based framework for software development effort prediction”, Information and Software Technology, pp. 31-48, Vol. 47(1), 2005.
[75]. Azzeh, M., Neagu, D., and Cowling, P. I., “ Analogy-based SEEusing Fuzzy numbers”, Journal of Systems and Software, pp.270-284, Vol. 84(2), 2011.
[76]. Mendes, E., Mosley, N., and Counsell, S., “ A replicated assessment of the use of adaptation rules to improve Web cost estimation. In Empirical Software Engineering”, Proceedings. 2003 International Symposium on ISESE 2003, pp. 100-109, ISBN: 0-7695-2002-2, 2003.
[77]. Mendes, E., Watson, I., Triggs, C., Mosley, N., and Counsell, S., “A comparative study of cost estimation models for web hypermedia applications”, Empirical Software Engineering, pp. 163-196, Vol. 8(2), ISSN: 1382-3256, 2003.
[78]. Ohlsson, M.C., and Wohlin, C, “ An empirical study of effort estimation during project execution”, Proc. 6th Intl Software Metrics Symposium. Boca Raton FL, pp. 91-98,ISSN: 0-7695-0403-5, 1999.
[79]. Rainer, A., and Shepperd, M, “Re-planning for a successful project schedule”, Proceedings. Sixth International In Software Metrics Symposium , pp. 72-81,ISBN: 0-7695-0403-5, 1999