Survey on Software Effort Estimation Technique – A Review

1E.KARUNAKARAN, 2N.SREENATH
Department of Computer Science and Engineering
Pondicherry Engineering College, Puducherry,
INDIA
ekaruna@pec.edu, 2nsreenath@pec.edu

Abstract : This paper aims to provide improvements of software sizing, effort estimation and cost estimation research through a systematic review of the previous work carried out by many authors. The review used 150 research papers from different journals and conferences and analysed in three title namely software sizing models, effort estimation models, and cost estimation models. The review results combined with other knowledge provide support for recommendation for future software sizing, effort estimation and cost estimation.

Keywords : Software Sizing, Software Effort Estimation, Software Cost Estimation, Systematic Review, effort prediction, software cost prediction.

1. INTRODUCTION

This paper reviews journals and conference papers of high quality of software size, effort estimation and cost estimation by a systematic manner. This paper is supporting the future estimation research work. Our review is organized under three topics namely Software Sizing Model, Software Effort Estimation Model and Software Cost Estimation Model which are interdependent.

1.1 Software Sizing Models : Software sizing is carried out in either lines of codes (LoC) or using Full Function Point (FFP). Sizing is the first step to find Effort and Cost Estimation. Determining the size of software is an essential activity among the tasks of software management. Effective software project sizing process is one of the most challenging activities in the software development, since proper project planning, monitoring and controlling cannot be done efficiently without accurate sizing.

1.2 Software Effort Estimation Models : Software effort estimation is one of the significant steps in software project management process since the success or failure of the software project highly depends upon the accuracy of the effort estimation. The criticalities associated with software effort estimation are i) effort estimation process must be done in earlier

phase of software planning and development, and ii) although number of methods and metrics are available for effort estimation, when the size of the software grows exponentially, all those existing methods fail to produce accurate effort estimation.

1.3 Software Cost Estimation Models : Accurate prediction of Software Cost is a challenging research issue. A number of methodologies are available in the literature. The techniques applied for improving the existing methodologies offer different measures of accuracy. Choosing an appropriate method for a dataset is an uphill task. There arises a need for well-established statistical frameworks and automated tools to reinforce and perform comprehensive experimentation.

2. REVIEW PROCESS

2.1 Software Sizing Models :

The activities are planned and the effort involved is estimated based on the software size. Estimating software development is an uphill task for software engineers, predominantly when there is uncertainty and subjectivity associated with the factors that influence effort. It is better if the estimates are done as the project is developed so as to improve the management of ensuing tasks.

2.1.1 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 [3].

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 1995 [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, labour 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 1996, 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 into

account. Based on the results, it is evident that the CBSD approach is capable of reducing the total development cost by 50 -70% [8]. Nevertheless, testing a CBS requires more time and effort than a system developed using custom development.

Hakuta et al. [9] has proposed a general approach similar to CBM general that is independent of the domain of application. This is applicable at the program level and considers the processing units, processing complexity and environmental factors. In this bottom up approach, the estimator computes the final size of a program based on the products generated at different stages of development. This model uses subjective variables like complexity explicitly for prediction whereas CBM uses them for explanation.

Yakimovich et al. [10] has presented a different approach for estimating the effort of component integration. The increase in effort for writing wrappers or adapters by means of glue code is taken into consideration. The assumptions about interactions of individual components and system architecture are presented as interaction vectors with variables that represent the inter-component interaction assumptions for packaging, control, information flow, synchronisation and binding. The component integration effort is estimated by comparing the interaction vectors for the system's architecture and component.

Apart from Lines of Code (LoC), a frequently used measure, there are several methods available for prior estimation of the final LoC of a software system. In [11], the results of validation of the component-based method for software sizing are shown. 46 projects involving more than 100,000 LoC of a fourth-generation language were analysed. CBM shows reasonable performance though not as Mark II function points as the performance is based on the type of the component.

A complete approach for estimating the effort of CBSD is the constructive COmmerCial Off-The-Shelf (COCOTS) integration cost model as mentioned in [12-14] developed as an extension of the COCOMO II model [15]. It is based on two characteristics that define components namely, the unavailability of source codes of components with the application developer and the future evolutions of components that are beyond the control of the application developer. It involves three sub-models that estimate the efforts of component assessment, tailoring and integration activities. Another economic goal-question-metrics approach for CBSD

is proposed by Dagnino et al. [16]. Two goals are formulated to evaluate the benefits of CBSD. The goals are to reduce the cost and effort by using CBSD. This method includes several questions that have to be answered to satisfy the goals. The answers are obtained by asking several sub-questions.

In [17], Wijayasiriwardhane has conducted a detailed survey on the recent researches done on predicting the effort of CBSD. The modelling technique used, the type of data required, the type of estimation provided, life cycle activities covered and the level of acceptability with regard to any validation are taken for discussion.

2.1.2 Proxy Based Size Estimation Model

Humphrey [18] has suggested a proxy-based estimating method in which objects are used as proxies. The objects are then characterized so as to estimate size for each category. This approach is similar to the CBM as it typifies components based on the environment. In this method, the size of each component is determined based on analogy by comparing with a database of previous developments.

2.2 Cost Estimation Models :

Mittas et al. [19] has presented a framework based on an automated tool that facilitates strategies intelligent decision-making. This framework provides visualization and statistical comparison of the errors of the existing cost estimation methods. StatREC, a Graphical User Interface (GUI) statistical toolkit is used in each step. StatREC takes a simple data matrix of predictions by different models as input. It provides a variety of graphical tools and statistical hypothesis tests that assist the users in answering the questions and choosing the appropriate model themselves.

2.2.1 Software Life-cycle Model (SLIM) :

Software Life-cycle Model (SLIM) is designed to estimate effort, schedule and defect rate [20]. It is one of the earliest algorithmic cost models developed. It is generally known as a macro estimation model and is based on the Norden/Rayleigh function and SLIM involves the following functions namely, Calibration, Building an information model and Software sizing.

Calibration involves fine tuning the model to represent the local software development environment by interpreting a historical database of past projects. The information model of the software system is built by collecting software characteristics, personal attributes, and computer attributes etc. To determine the size of software, SLIM uses an automated version of the Lines of Code (LoC) costing technique. It considers the total life cycle effort in working years, development and the technology constant, and combines the effect of using tools, languages, methodology and Quality Assurance (QA). The value of technology constant varies from 610 to 57314. For experienced projects, the technology constant is high. It uses linear programming to consider development constraints on both cost and effort. It involves less number of parameters to generate an estimate when compared to COCOMO 81 and COCOMO II. Nevertheless, the estimates are extremely sensitive to the technology factors and not suitable for small projects.

Panlilio-Yap [21] has discussed in detail about SLIM, a metrics-based estimation tool, taking a Toronto project into consideration. The tool has a rich set of what-if capabilities that explores the possible alternatives that satisfy project constraints. The impacts on resource requirements, project duration, and product quality are assessed.

2.2.2 SEER-SEM for Software :

Software Evaluation and Estimation of Resource – Software Estimating Model (SEER-SEM), an algorithmic project management software application estimates, plans and monitors the effort and resources required for any type of software development and/or maintenance project. It is a tool based on the original Jensen model [22] developed by Galorath [23] which aids in project planning, cost management and tracking throughout the software development life cycle. It has the ability to make predictions relying on the parametric algorithms, knowledge bases, simulation-based probability, and historical precedents. It accurately estimates a project's cost schedule, risk and effort before the commencement of the project. SEER for Software Version 7.3, an improvement over the original implementation, shows that any version of SEER could be integrated to support all phases of a project's lifecycle. The original SEER-SEM is branched into SEER for Information Technology, SEER for Hardware, Electronics, and Systems and SEER for Manufacturing. SEER for Information Technology (SEER-IT) assists the IT

professionals to predetermine the design, build, and maintenance of information technology infrastructures and service management projects. SEER for Hardware, Electronics, and Systems (SEER-H) helps in the life-cycle cost estimation of any type of hardware, electronics or system. SEER for Manufacturing (SEER-MFG) is designed to aid in estimating the detailed production costs of manufacture, covering a wide range of state-of-practice and state-of-the-art manufacturing process knowledge.

2.2.3 PRICE-S :

Programming Review of Information Costing and Evaluation-Software (PRICE) [44] is the earliest developer of parametric cost estimation software. It is used for estimating US DoD, NASA and other government software projects. It relates the basic costs of Engineering and production to parameters that includes a specification profile of units to be built, amount of work to be performed, allowed schedule and resources available. It uses parametric relationships obtained by curve-fitting procedures performed on a historical repository of significant cost data.

2.2.4 Software Productivity Research (SPR) :

Jones [25] has proposed Software Productivity Research (SPR) knowledge plan. It is a knowledge-based estimation tool. It includes mechanisms to size projects and to estimate the effort, resources, schedule and defects at four levels of granularity namely, project, phase, activity and task.

2.2.5 COnstructive COst MOdel (COCOMO) :

Boehm [26] developed the COCOMO, typically called COCOMO 81. It is an algorithmic software cost estimation model. It uses a basic regression formula with parameters derived from historical project data, and characteristics of current and future project. COCOMO consists of a hierarchy of increasingly detailed and accurate forms. Basic COCOMO at the first level with limited accuracy is good for quick, early, rough order of magnitude estimates of software costs. It lacks factors that account for difference in project attributes (Cost Drivers).

The Intermediate COCOMO takes the Cost Drivers into account, while the Detailed COCOMO also accounts for the influence of individual project phases.

Ryder [27] has applied fuzzy modelling techniques to most widely used models for effort prediction namely, COCOMO and the Function- Points models.

In 1995, COCOMO II was developed and finally published in 2000 [15,28]. COCOMO II, the successor of COCOMO 81 is suited for estimating modern software development projects. As desktop development, code reusability and the use of off-the-shelf software components became predominant instead of mainframe and overnight batch processing, COCOMO II was developed to assist in software development.

An emotional COCOMO II model is proposed for software cost estimation in [29]. In COCOMO II, the characteristics of team members are ignored and only the project characteristics are considered. Fuzzy Emotional COCOMO II Software Cost Estimation (FECSCE) model is proposed, wherein besides project characteristics, it considers the communication skills, personality, mood and capabilities of team members. Multi-Agent System (MAS) is used to simulate team communications.

Mittas and Angelis [30] has discussed about the limitations of past studies and have proposed a method for ranking several prediction models and clustering them into non-overlapping groups. Ranking and clustering is augmented with additional mechanisms available in StatREC that helps in the in-depth exploration of the properties and capabilities of prediction models. The evolution of the COCOMO cost estimation models from 1981 to 2005 is presented. Mainly, COCOMO 81, ADA COCOMO, and COCOMO II are dealt here. COCOMO keeps introducing and illustrating software engineering methods and techniques.

2.2.6 Calibrated COCOMO :

A methodology that calibrates the Constructive Cost Estimation Model (COCOMO) is presented in [51]. This approach aids COCOMO in completely and automatically calibrating to the development environments beyond which it was constructed. It not only considers the nominal coefficients, but also all cost-driver multipliers simultaneously. A reformulated structure

which possesses additional statistical properties that are unavailable within COCOMO is also presented. It can be used as a framework for developing generalized and potentially dynamic models.

An extended set of tool rating scales based on the completeness of tool coverage, the degree of tool integration, and tool maturity/user support is provided in [32]. These scales refine the way in which CASE tools are efficiently evaluated within COCOMO II. To find the best fit of weighting values for the extended set of tool rating scales and to increase prediction accuracy, a Bayesian approach is adopted to combine two sources of information. The model is validated by using the cross-validation methodologies, data splitting and bootstrapping. It disaggregates the parameters that have significant impacts on software development productivity and calibrates the best-fit weight values based on data-determined and expert-judged distributions.

Building cost estimation models is a search problem in which an optimal solution satisfying an objective function should be returned. Certain constraints like coefficients of COCOMO models must be non-negative, should also be satisfied. In [33], Nguyen et al. has proposed a constrained regression technique that involves objective functions and constraints to estimate the coefficients of the COCOMO models. Cross-validation is done to evaluate and compare the prediction accuracy with approaches like least squares, stepwise, Lasso, and Ridge regression. The regression model with minimum sum of relative errors and non-negative coefficients is a technique that is suitable for calibrating the COCOMO model parameters.

2.2.7 Simulation Based Cost Estimation Models :

Some of the existing models in the literature that have used simulation environments for cost estimation are :

Gray [34] has presented diverse predictive model-building techniques such as robust statistical procedures, several forms of neural network models, fuzzy logic, CBR and regression trees. A simulation-based study is also made on the performance of these empirical modelling

techniques using size and effort software metric dataset. It is observed that M-estimation regression performs better than other parametric and non-parametric techniques.

In [35, 36], a Multi-Agent System simulation tool that provides information about the behaviour of a team is designed. This tool enables the project managers to integrate a team in terms of cost and time.

Naunchan and Sutivong [37] has designed an adjustable cost model for estimating the effort and duration of the component integration. It is an integration of three existing approaches namely, effort multipliers of the COCOTS model that identifies and determines the productivity factors, system dynamics that simulates the software process and communication overhead assumptions that adjust the productivity of the development team. It takes into consideration diverse factors like the estimated development time, percentage of rework due to upgrades of components during the development, workforce, and communication overhead rate, size of the system in terms of LoC for glue code development and system integration, and function size for the tailoring process. The model gives a relationship between workforce and development time by plotting them. The area under the curve gives the total estimated effort. It gives the estimate of the required workforce for each time period and the minimal effort needed.

Choi and Bae [38] has propounded a simulation method for determining the project performance dynamically. The changes that occur in the user requirements or project personnel are used to estimate effort, schedule, and defect density. COCOMO II is integrated with system dynamics.

2.2.8 Multi-Agent Based Cost Estimation Models :

Ping et al. [39] has built architecture of Multi-Agent Systems for cost estimation, wherein fuzzy classification is employed to classify the user's request to fulfil the task by expert agents. Each agent represents a kind of cost estimation method.

A CBR approach integrated with Multi-Agent technology is proposed by Al-Sakran [40] to retrieve related projects from a comparable domain in multi-organizational distributed

datasets. Wang et al. [41] has proposed an ontology-based fuzzy agent for Capability Maturity Model Integration (CMMI) project planning to estimate the total project cost. Lee et al. [42] has propounded an Ontology-based Intelligent Decision Support Agent (OIDSA) for project monitoring and control of CMMI.

Further, Lee et al. [43] has presented an ontology based intelligent estimation agent for total project cost estimation that includes a CMMI-based project planning ontology. It contains information that is predefined by domain experts and a fuzzy cost estimation mechanism to deduce the total project cost. Lee and Wang [44] has designed an ontology-based computational intelligent Multi-Agent System for CMMI assessment.

2.2.9 Neural Networks and Fuzzy Logic Based Cost Estimation Models :

Predicting the cost and quality of software product under development is a vital but challenging task. There is no universal model that accurately and effectively predicts the software development cost.

In [45], Wavelet Neural Network (WNN) is used to forecast the software development effort. Two types of WNN are used with the Morlet function and Gaussian function is used as a transfer function. Threshold acceptance Training Algorithm is proposed for Wavelet Neural Network (TAWNN). The performance of WNN variants is compared with other techniques such as Multilayer Perceptron (MLP), Radial Basis Function Network (RBFN), Multiple Linear Regression (MLR), Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS) and Support Vector Machine (SVM) in terms of Mean Magnitude Relative Error (MMRE) obtained on Canadian Financial (CF) dataset and IBM Data Processing Services (IBMDPS) dataset. It is evident that the WNN-Morlet for CF dataset and WNN-Gaussian for IBMDPS performs better when compared to all the other techniques.

2.2.10 Parametric Cost Estimation Models :

Parametric cost estimation models demand continuous calibration and improvement to ensure more accurate software estimates to reflect the changes in the software development contexts. Local calibration is frequent as a subset of model parameters is often tuned so as to increase model usability and accuracy.

In [46], a quantitative analysis of effectively handling local bias related to historical cross-company data is presented. It improves the usability of cross-company datasets for calibrating and maintaining parametric estimation models. A method is defined for measuring the local bias associated with individual organization data subset in the latest COCOMO II calibration dataset. The impacts of local bias on the performance of an estimation model are analysed. A weighted sampling approach to handle local bias is proposed. Local bias has a negative impact on the performance of parametric model. The local bias based weighted sampling technique reduces negative impacts of local bias on model performance.

2.2.11 Linear-Least-Squares Regression Cost Estimation Models :

The most commonly used models and tools in software cost estimation that predict software development effort are based on linear-least-squares regression such as COCOMO [47, 48].

Shepperd and MacDonell [49] have proposed a framework for formal validation of models based on the following concepts - comparison with a reference model, significance testing and the evaluation of effect size. The framework is appropriate for situations, where the objective is the comparison of only two prediction systems. When the number of models is more, for multiple hypotheses have to be tested simultaneously, the problem of error inflation is taken into account.

2.3 Effort Estimation Models

There are two challenges in software development namely, unmanaged risks and inaccurate estimations of resources for a project.

Kocaguneli et al. [50] 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 method

computes 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. [51] has given a review of general techniques and models regarding effort estimation. The merits and pitfalls 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. [52] 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.

2.3.1 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 [53] has dealt with the estimation by analogy and have described the similarities and differences of the existing software cost estimating techniques. Mukhopadhyay et al. [54] has also used analogy for software effort estimation by retrieving the most similar cases. It is seen that the analogy based approach is more accurate and consistent than the function point and COCOMO models.

An analogy based approach for effort estimation is proposed by Shepperd and Schofield [55]. The projects are characterized in terms of features. The most similar projects to the one for which a prediction is required is compared with the developed ones. Similarity is the Euclidean distance in n-dimensional space where ‘n’ is the number of project features. The known effort values of the nearest neighbours 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 analysed. The analogy based schemes outperform algorithmic models based on stepwise regression.

ANGEL proposed by Shepperd and Schofield [55] 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. [56] 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 [55 - 63].

Myrtveit and Stensrud [57] and Briand et al. [58] 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. [64] 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. [62, 63] and Shepperd and Schofield [55], EA performs better in contrast to the linear and stepwise regression models. Jorgensen et al. [65] has used regression towards the mean method to regulate EA. This method is appropriate for extreme analogues and inaccurate estimation models. The adjusted estimation is accurate than EA without adjustment.

To improve EA, Mittas et al. [66] has used iterative re-sampling method. According to them, EA is closely related to formal nearest neighbour non-parametric regression.

EA needs more number of sensed similarity methods [67]. 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 software effort estimation 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. [68] 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. [69] classifies the ASEE studies and proposes a new modified ASEE 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 analysed 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).

2.3.2 Neural Network (NN) Based Effort Estimation Models :

Many researchers like Jorgerson [70], Srinivasan and Fisher [71], Hughes [72], Wittig and Finnie [73], Samson et al. [74], Schofield [75], Seluca [76], Heiat [77] 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 [65], whereas Analogy based-methods identify one or more developed projects similar to the project currently being developed and compute the total estimated effort manually [78]. Parametric model-based methods rely mainly on historical data based equations. Effort is taken as function of parameters influencing effort [79]. 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 [71]. 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 [80 - 82].

A software effort estimation method is developed by Laqrichi [83] 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.

2.3.3 Fuzzy Based Effort Estimation Models :

Fuzzy logic offers a better mapping between input and output spaces [84]. The main properties of a fuzzy model are that it operates at a level of linguistic terms (fuzzy sets), represents and processes uncertainty [85]. Fuzzy set theory is a complete approach that deals with linguistic values like small, medium, average, or high [86]. Fuzzy model is best suited for software development effort estimation. Developing a precise mathematical model for the domain is challenging [87]. 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 [88] 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. [89], wherein fuzzy logic is applied to the fifteen cost factors of COCOMO 81. The randomly generated dataset is compared with actual data of COCOMO 81. From the results, it is evident that the results of the FLM are mostly similar to those of COCOMO [81].

Idri et al. [86] has propounded an approach based on fuzzy logic named Fuzzy Analogy for COCOMO 81 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 '81, Classical intermediate COCOMO'81 and Classical Analogy. Musflek et al. [90] has proposed f-COCOMO, a fuzzy model for COCOMO 81 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. [91] 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. [92] has presented a FLM based on triangular membership functions. Randomly generated dataset and the one used for COCOMO 81 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. [93] for COCOMO-81 database. Fuzzy regression model is better than the existing basic estimation models.

In 2004, Reformat et al. [94] 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, Xu and Khoshgoftaar [95] has 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 de-fuzzification. The proposed system is applied for all three COCOMO 81 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. [68]. 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 [96]. 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 Centroid method is developed at de-fuzzification stage. The results show that FES offers better performance in terms of diverse statistical criteria.

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

Idri et al. [64] 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. [97] has proposed a software effort estimation 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. [98] 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 [99] has designed diverse weighted GRA models for software effort estimation like 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.

2.3.5 Phase-Level Based Effort Estimation Models :

Software effort estimation during 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. Accurate 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. [100] 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 [101] dealt with the fuzziness of several aspects of COCOMO model. They observed that an accurate 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 [84, 102, 103].

The effort data recorded for completed project tasks are used to predict the effort needed for subsequent activities in [104]. 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 [105]. 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. [106]. 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 [107], 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 analysed. 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.

2.3.6 Case Based Reasoning Effort Estimation Model :

On the other hand, Mendes et al. [108, 109] 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.

2.3.7 Empirical Effort Estimation Models :

The empirical work done by Ohlsson and Wohlin [110] is similar to the one used by Kulkarni et al. [100]. 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 [111] 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 [112] 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.

2.3.8 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. [113], while a group of non-linear regression equations are presented by Boehm [26] in COCOMO 81 and COCOMO II [114]. An Albus multilayer perceptron is used to predict software effort in [74] 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

[115], 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. [116], Cuadrado-Gallego et al. [117] and Aroba et al. [118]. 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. [116, 117]. 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. [118]. 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 [119] 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 software effort estimation method. 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. [120], a new data partitioning-based approach is proposed to achieve more accurate 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.

2.3.9 Class Point and Use Case Point (UCP) based Effort Estimation Model :

Satapathy et al. [121] 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. [122] 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. [123] 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 accurate estimates.

There is a growing interest in software effort estimation based on use cases. In [124], 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. [125] 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. [126] has proposed a unified mapping of Use case Modelling 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 [127] 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. [128]. 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. [129] 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. [130] 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 accurate 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.

Software effort estimation 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. [131] has proposed a regression model for software effort estimation based on UCP model. A Sugeno Fuzzy Inference System (FIS) approach is applied on this model to improve the estimation.

2.4 Extract of the Literature

From the literature review conducted, the following short comings are identified.

i) Generally, it is seen that most of the researches conducted focus on the datasets and the initial phase of the estimation, but fail to concentrate on the development phase of the effort,

ii) Most of the methods for finding size, effort and cost are not transparent to the client,

iii) Most of the methods use Fuzzy Logic (FL) to handle imprecision in the datasets but do not focus on performance factors,

iv) Regression based software estimation techniques were not much explored in the literature, and

v) Deming regression based estimation does not provide more accuracy with minimum software multipliers used for software effort estimation.

REFERENCE

1. Verner, J. and Tate. G, "A Software Size Model", IEEE Transaction on Software Eng., pp. 265-278, vol. 18, no. 4, ISSN : 0098-5589, 2003.

2. Karpowich, M. F., Sanders, T. R., and Verge, R. E, "An Economic Analysis Model for Determining the Custom versus Commercial Software Trade-off'. Analytical Methods in Software Engineering Economics, pp. 237-252, ISSN: 978-3-642-77797-4, 1993.

3. Abts, C. M., and Boehm, B., COCOTS (Constructive COTS), "Software Integration Cost Model: An Overview". USC Centre for Software Engineering, University of Southern California, briefing, ISSN: 3540455884, 1998.

4. Stutzke, R. "Costs impact of COTS volatility", Proceedings of Workshop on COCOMO, Vol. 2, 1995.

5. Ellis, T. COTS, "Integration in Software Solutions-A Cost Model, 1995", Proceedings of the NCOSE International Symposium Systems Engineering in the Global Marketplace, pp. 174-182, Vol.5 (1), 2014.

6. Albrecht, A. J., and Gaffney Jr, J. E., "Software function, source lines of code, and development effort prediction" A software science validation Software Engineering, IEEE Transactions on, pp. 639-648, Vol.9 (6), ISSN: 0098-5589, 1983.

7. Aoyama, M., "A component-based software development methodology", Proceeding of IPSJ SIG Notes, pp. 33–40, Vol.96 (84), 1993.

8. Aoyama, M, "Process and economic model of component-based software development:" A study from software CALS next generation software engineering program, Proceedings Fifth International Symposium on Assessment of Software Tools and Technologies, pp.100-103, ISBN: 0-8186-7940-9, 1997.

9. Hakuta, M., Tone, F., and Ohminami, M. "A Software Size Estimation Model and Its Evaluation" J. Systems and Software, pp. 253-263, vol. 37(3), doi:10.1016/S0164-1212(96)00020-9, 1997.

10. Yakimovich, D., Bieman, J. M., and Basili, V. R "Software architecture classification for estimating the cost of COTS integration. In Software Engineering", Proceedings of the 1999

International Conference on Software engineering, pp. 296-302, ISBN:1-58113-074-0, 1999.

11. Dolado, J. J. "A validation of the component-based method for software size estimation. Software Engineering", IEEE Transactions on Software Engineering, pp. 1006-1021, Vol. 26(10), ISSN: 0098-5589, 2000.

12. Abts, C., Boehm, B. W., and Clark, E. B. COCOTS, "A COTS software integration lifecycle cost model-model overview and preliminary data collection findings", In the proceeding of the ESCOM-SCOPE Conference, pp. 18-20, 2000.

13. Abts, C., Boehm, B., and Clark, E. B "Empirical observations on COTS software integration effort based on the initial COCOTS calibration database" ICSE COTS Workshop, 2000.

14. Abts, C. M. "Extending the COCOMO II software cost model to estimate effort and schedule for software systems using commercial-off-the-shelf (cots) software components: the COCOTS model", ACM Digital Library", 2004.

15. Boehm, B., Clark, B., Horowitz, E., Westland, C., Madachy, R., and Selby, R., "Cost models for future software life cycle processes: COCOMO 2.0", Annals of software engineering, pp. 57-94, Vol.1(1), ISSN: 1022-7091, 1995.

16. Dagnino, A., Srikanth, H., Naedele, M., and Brantly, D, "A model to evaluate the economic benefits of software components development", IEEE International Conference on Systems Man and Cybernetics, pp. 3792 - 3797, Vol. 4, ISSN: 1062-922X , 2003.

17. Wijayasiriwardhane, T., Lai, R., and Kang, K. C. (2011), "Effort estimation of component-based software development–a survey", IET software, pp. 216-228, Vol. 5(2), ISSN : 1751-8806, 2011.

18. Humphrey, W. (1995), "A Discipline for Software Eng", Addison-Wesley, ISBN:0201546108, 1995.

19. Mittas, N., Mamalikidis, I., and Angelis, L, "A framework for comparing multiple cost estimation methods using an automated visualization toolkit" Information and Software Technology, pp.310-328, Vol.57, D, 2015.

20. Putnam, L. H., " A general empirical solution to the macro software sizing and estimating problem", IEEE transactions on Software Engineering, pp. 345-361, Vol4 (4), ISSN : 0098-5589, 2006.

21. Panlilio-Yap, "N. Software estimation using the SLIM tool", Proceedings of the 1992 conference of the Centre for Advanced Studies on Collaborative research-Volume 1, IBM Press, pp. 439-475, 1992.

22. Jensen, R. "An improved macrolevel software development resource estimation model", In the 5th ISPA Conference, pp. 88-92, 1983.

23. Galorath, D. "Why SEER Got Started?" 2008.

24. PRICE Systems L.L.C. <http://www.pricesystems.com>.

25. Jones, C. (1996). "Applied Software Measurement: Assuring Productivity and Quality", McGraw Hill., ISBN- 10: 0070328269 , 1996.

26. Boehm, B., "Software Engineering Economics", Prentice Hall, ISBN:0138221227, 1981.

27. Ryder, J. "Fuzzy modelling of software effort prediction", Proceedings of Information Technology Conference IEEE, pp. 53-56, ISBN: 0-7803-9914-5, 1998, September.

28. Boehm, B., Abts, C., Clark, B., and Devnani-Chulani, S, "COCOMO II model definition manual". The University of Southern California, 1997.

29. Kazemifard, M., Zaeri, A., Ghasem-Aghaee, N., Nematbakhsh, M. A., and Mardukhi, F. "Fuzzy emotional COCOMO II software cost estimation (FECSCE) using multi-agent systems", Applied Soft Computing, pp.2260-2270, Vol.11(2), 2011.

30. Mittas, N., and Angelis, L. "Ranking and clustering software cost estimation models through a multiple comparisons algorithm", IEEE Transactions on Software Engineering, pp. 537-551, Vol.39 (4), ISSN: 0098-5589, 2013.

31. Gulezian, R., "Reformulating and calibrating COCOMO", Journal of Systems and Software, pp. 235-242, Vol.16(3), 1991.

32. Baik, J., Boehm, B., and Steece, B. M, "Disaggregating and calibrating the CASE tool variable in COCOMO II", IEEE Transactions on Software Engineering, pp. 1009-1022, Vol. 28(11), ISSN: 0098-5589, 2002.

33. Nguyen, V., Steece, B., and Boehm, B. "A constrained regression technique for COCOMO calibration", Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, pp. 213-222, ISBN: 978-1-59593-971-5, 2008.

34. Gray, A. K., "A simulation-based comparison of empirical modelling techniques for software metric models of development effort", Neural Information Processing, 1999. Proceedings of ICONIP'99 6th International Conference, pp. 526-531, Vol. 2, ISSN: 0-7803-5871-6, 1999.

35. Martínez-Miranda, J., Aldea, A., and Bañares-Alcántara, R., "Simulation of work teams using a multi-agent system", Proceedings of the second international joint conference on Autonomous agents and multiagent systems, pp. 1064-1065, ISBN:1-58113-683-8, 2003.

36. Martınez-Miranda, J., and Aldea, "A. Emotions in human and artificial intelligence", Computers in Human Behaviour, pp. 323-341, Vol. 21(2), 2005.

37. Naunchan, P., and Sutivong, D. "Adjustable cost estimation model for COTS-based development", Software Engineering Conference ASWEC 2007. Australia, pp. 341-348, ISSN: 1530-0803, 2007.

38. Choi, K., and Bae, D. H., " Dynamic project performance estimation by combining static estimation models with system dynamics", Information and software technology, pp. 162-172,Vol. 51(1), 2009.

39. Ping, L., Yongtong, H., and Bode, J. "Multi-agent system for cost estimation". Computers and industrial engineering, pp. 731-735, Vol.31(3), 1996.

40. Al-Sakran, H., " Software cost estimation model based on integration of multi-agent and case-based reasoning", Journal of Computer Science, pp. 276-282, Vol. 2(3), ISSN 1549-3636, 2006.

41. Wang, M. H., Lee, C. S., Yan, Z. R., Chuang, H. H., Lo, C. F., and Lin, Y. C, "A novel fuzzy CMMI ontology and its application to project estimation", Journal of Internet Technology, pp.317-325, Vol.9(4), 2008.

42. Lee, C. S., Wang, Chen, J.J, " Ontology-based intelligent decision support agent for CMMI project monitoring and control", International Journal of Approximate Reasoning, pp. 62-76, Vol.48, 2008.

43. Lee, C. S., Wang, M. H., Yan, Z. R., Lo, C. F., Chuang, H. H., and Lin, Y. C, "Intelligent estimation agent based on CMMI ontology for project planning", IEEE International Conference on In Systems, Man and Cybernetics, pp. 228-233, ISSN: 1062-922X, 2008.

44. Lee, C. S., and Wang, M. H., "Ontology-based computational intelligent multi-agent and its application to CMMI assessment", Applied Intelligence, pp. 203-219, Vol. 30(3), ISSN: 0924-669X, 2009.

45. Kumar, K. V., Ravi, V., Carr, M., and Kiran, N. R, "Software development cost estimation using wavelet neural networks", Journal of Systems and Software, pp. 1853-18678, Vol.1 (11), 2008.

46. Yang, Y., He, Z., Mao, K., Li, Q., Nguyen, V., Boehm, B., and Valerdi, R. , "Analyzing and handling local bias for calibrating parametric cost estimation models", Information and Software Technology, pp. 1496-1511, Vol. 55(8), 2013.

47. Fenton, N. E., and Pfleeger, S. L, "Software metrics: a rigorous and practical approach", PWS Publishing Co. Boston, MA, USA, ISBN 9781439838228, 1997.

48. Pressman, R.S. "Software Engineering: A Practitioner's Approach", fourth ed. McGraw-Hill, New York, NY, USA, ISBN: 0078022126, 1997.

49. Shepperd, M., and MacDonell, S., "Evaluating prediction systems in software project estimation", Information and Software Technology , pp. 820-827, Vol. 54(8), 2012.

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

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

52. 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..

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

54. 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.

55. 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.

56. 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.

57. 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.

58. 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.

59. 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.

60. 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.

61. 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.

62. 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.

63. 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.

64. 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.

65. 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.

66. 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.

67. Chiu, N. H., and Huang, S. J., "The adjusted analogy-based software effort estimation based on similarity distances", Journal of Systems and Software, pp. 628-640, Vol. 80(4), 2007.

68. 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.

69. Idri, A., Azzahra Amazal, 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.

70. 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.

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

72. 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.

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

74. 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.

75. 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.

76. 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.

77. 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.

78. 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.

79. 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.

80. 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.

81. 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.

82. 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.

83. 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.

84. 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.

85. 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

86. 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.

87. Chiu, N. H., and Huang, S. J., “The adjusted analogy-based software effort estimation based on similarity distances”, Journal of Systems and Software, pp. 628-640, Vol. 80(4), 2007.

88. 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.

89. Idri, A., Azzahra Amazal, 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.

90. 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.

91. 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.

92. 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.

93. Wittig, G., and Finnie, G, “Estimating software development effort with connectionist models”, Information and Software Technology”, pp.469-476, Vol.39(7), 1997.

94. 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.

95. Schofield, C. “Non-algorithmic effort estimation techniques”. ESERG, TR98-01, 1998.

96. Serluca, C, “An investigation into software effort estimation using a back propagation neural network”, Bournemouth University, ISBN: 1605667676 1995.

97. 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.

98. 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.

99. Boehm B, “Software Cost Estimation with Cocomo II”, Prentice-Hall, ISBN: 0130266922, 2000.

100. 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.

101. 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.

102. 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.

103. Laqrichi, S., Marmier, F., Gourc, D., and Nevoux, J., “Integrating uncertainty in software effort estimation using Bootstrap based Neural Networks”, IFAC-Papers online, pp.954-959, Vol.48(3), 2015.

104. 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.

105. Pedrycz, W., Gomide, F.,”An Introduction to Fuzzy Sets”, The MIT Press, ISBN: 8120346998, 1998.

106. 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.

107. Lewis, J.P, “Large limits to software estimation”, ACM Software Engineering Notes, Vol.26 (4), pp.54–59, 2001.

108. 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.

109. 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.

110. Musflek, P., Pedrycz, W., Succi, G., Reformat, M, “Software cost estimation with fuzzy models”, Applied Computing Review, pp.24-29, vol. 8 (2), 2000.

111. 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.

112. 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.

113. 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.

114. Reformat, M., Pedrycz, W., Pizzi, N., “Building a Software Experience Factory Using Granular-Based Models. Fuzzy Sets and Systems” Elsevier. 2004.

115. 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.

116. 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.

117. 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.

118. 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.

119. 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.

120. 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.

121. 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.

122. 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.

123. 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.

124. 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.

125. Pfleeger, S. L., Wu, F., and Lewis, R., " Software cost estimation and sizing methods: issues, and guidelines", Rand Corporation, Vol. 269, 2005.

126. 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.

127. Azzeh, M., Neagu, D., and Cowling, P. I., " Analogy-based software effort estimation using Fuzzy numbers", Journal of Systems and Software, pp.270-284, Vol. 84(2), 2011.

128. 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.

129. 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.

130. 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.

131. 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

132. Jørgensen, M., Indahl, U., and Sjøberg, D, " Software effort estimation by analogy and regression toward the mean", Journal of Systems and Software, pp. 253-262, Vol. 68(3), 2003

133. Kok, P., Kitchenhan, B.A., Kirakowski, J., "The MERMAID Approach to Software Cost Estimation", In the Proceedings of ESPRIT Technical week, ISSN: 978-94-010-6803-1, 1990.

134. Boehm, B., Abts, Ch, Brown, A.W., Chulani, S., Clarck, B.K., Horowitz, E., Madachy, R., Reifer, D., and Steece, B., COCOMO II, Prentice Hall, ISBN: 0137025769, 2000.

135. Briand, L.C., Wieczorek, I, "Software Resource Estimation" ,Encyclopedia of Software Engineering", John Wiley and Sons, New York, pp. 1160–1196, vol. 2. 2001.

136. Cuadrado-Gallego, J. J., Sicilia, M. Á., Garre, M., and Rodríguez, D, "An empirical study of process-related attributes in segmented software cost-estimation relationships", Journal of Systems and Software, Vol.79(3), pp.353-361, 2006.

137. Cuadrado-Gallego, J. J., Rodriguez, D., Sicilia, M.A., Rubio, M.G., Crespo, A.G, "Software project effort estimation based on multiple parametric models generated through data

clustering", Journal of Computer Science and Technology, Vol. 22 (3), pp.371–378, ISSN: 1000-9000, 2007.

138. Aroba, J., Cuadrado-Gallego, J. J., Sicilia, M. Á., Ramos, I., and García-Barriocanal, E, " Segmented software cost estimation models based on fuzzy clustering", Journal of Systems and Software, pp.1944-1950, Vol. 81(11), pp.1944-1950, 2008.

139. López-Martín, C., Yáñez-Márquez, C., and Gutiérrez-Tornés, A, "Predictive accuracy comparison of fuzzy models for software development effort of small programs", Journal of Systems and Software, Vol.81(6), pp. 949-960, 2008.

140. Seo, Y. S., Bae, D. H., and Jeffery, R., "AREION: Software effort estimation based on multiple regressions with adaptive recursive data partitioning", Information and Software Technology, Vol.55 (10), pp. 1710-1725, 2013.

141. Satapathy, S. M., Kumar, M., &Rath, S. K. , "Class point approach for software effort estimation using soft computing techniques", Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference, pp. 178-183, ISSN: 978-1-4799-2432-5, 2013.

142. Kusumoto, S., Matukawa, F., Inoue, K., Hanabusa, S., and Maegawa, Y. , "Estimating effort by use case points: method, tool and case study. In Software Metrics", Proceedings. 10th International Symposium , pp. 292-299, ISSN: 0-7695-2129-0, 2004, September.

143. Mohagheghi, P., Anda, B., and Conradi, R., " Effort estimation of use cases for incremental large-scale software development" Software Engineering, ICSE 2005. Proceedings. 27th International Conference pp. 303-311, ISBN: 1-59593-963-2, 2005.

144. Anda, B., Benestad, H. C., and Hove, S. E., "A multiple-case study of software effort estimation based on use case points" Empirical Software Engineering, International Symposium, pp. 10-pp, ISSN : 0-7803-9508-5, 2005.

145. Costagliola, G., Ferrucci, F., Tortora, G., and Vitiello, G, "Class point: an approach for the size estimation of object-oriented systems Software Engineering" , IEEE Transactions on, Vol.31(1), pp.52-74, ISSN: 0098-5589 , 2005.

146. Živkovič, A., Rozman, I., and Heričko, M. , "Automated software size estimation based on function points using UML models. Information and Software Technology", Vol. 47(13), pp.881-890, 2005

147. Braz, M. R., and Vergilio, S. R., “Software effort estimation based on use cases”, Computer Software and Applications Conference, COMPSAC'06. 30th Annual International, Vol. 1, pp. 221-228. 2006, ISSN : 0-7695-2655-1, September 2006.

148. Kanmani, S., Kathiravan, J., Kumar, S. S., and Shanmugam, M., “Neural network based effort estimation using class points for OO systems”, In Computing: Theory and Applications 2007. ICCTA'07. International Conference , pp. 261-266, ISBN: 0-7695-2770-1, 2007, March

149. Kanmani, S., Kathiravan, J., Kumar, S. S., and Shanmugam, M, “Class point based effort estimation of OO systems using fuzzy subtractive clustering and artificial neural networks”, Proceedings of the 1st India software engineering conference, pp. 141-142, ISBN: 978-1-59593-917-3, ACM. 2008, February.

150. Ochodek, M., Nawrocki, J., and Kwarciak, K., “Simplifying effort estimation based on Use Case Points”, Information and Software Technology, pp.200-213, Vol.53(3), 2011.