International Journal of Engineering Inventions
ISSN: 2278-7461, ISBN: 2319-6491, www.ijeijournal.com
Volume 1, Issue 10 (November2012) PP: 29-34

An Approach towards Developing an Efficient Software Cost Estimation System Using Fuzzy and Analogy Methods

[1]Mr S A RAMESH KUMAR, [2]Dr.S.SivaSubramanian,
[1]*Asst Professor, Karpaga Vinayaga College of Engineering, Chennai*
[2]*Professor, Dhanalakshmi College of Engineering, Chennai*

Abstract:—Software development cost estimation is important for effective project management. Many models have been introduced to predict software development cost. In this paper, a novel emotional COnstructive Cost MOdel II (COCOMO II) has been proposed for software cost estimation. In COCOMO II only the project characteristics are considered, whereas the characteristics of team members are also important factors. Hence our proposed method is effectively estimated the software effort and the cost by employing analogy with fuzzy based methods. In order to overcome all these difficulties of existing systems, in the software cost estimation process, we proposed an efficient software cost estimation system based on fuzzy analogy. In this, we have developed and validated a set of candidate measures for software projects similarity. These measures are based on fuzzy sets, fuzzy reasoning and linguistic quantifiers. A new approach to estimate effort by analogy when software projects are described either by numerical or categorical data. The normal procedure for analogy is not applicable for categorical data. So a new approach which can be seen as a Fuzzification of the classical approach of estimation by analogy. The proposed method will applicable to Function Point Metric also. It easy to generate rules in fuzzy analogy method. The fuzzy analogy based cost estimation will overcome the multi co-linearity problem. The effort can be estimated based on the emotional characteristics of the employees in the projects. The neuroticism characters highly affect the performance of the cost estimation system. Thus our proposed method will give better performance when compared to these recent papers. Our method will overcome all the drawbacks of these previous methods.

I. INTRODUCTION

Software cost estimation is the process of predicting the effort required to develop software system.The key factor in selecting a cost estimation model is the accuracy of its estimates. It involves in the determination of some estimates are effort (usually in person-months), project duration (in calendar time) and cost (in dollars). The goal was to obtain an in-depth understanding of estimation practice and to examine factors that affect the accuracy of effort estimation. The basis for the measurement of estimation accuracy was a comparison of the actual use of effort with the estimated most likely effort provided in the planning stage of the project, i.e. the amount of effort the contractor believes that the project will require, regardless of the price to the customer or the budget [5]. As software grew in size and importance, it also grew in its complexity, making it very difficult to accurately predict the cost of the software development. The better accuracy in estimation can be achieved by developing the domain based useful models that constructively explain the development life cycle and accurately predict the effort of developing the software [16]. Effort Estimation with good accuracy helps in managing overall budgeting and planning. The accuracy of these estimates is very less and most difficult to obtain, because no or very little detail about the project is known at the beginning [12].Accurate effort estimates help software consultancies to make appropriate bids when quoting for tenders – a lower estimate than the actual will lead to a loss and an unreasonably high estimate will loose the bid. Such estimation models are developed using a set of measures that describe the software development process, product and resources such as developer experience, system size and complexity and the characteristics of the development environment, respectively [11]. This technique evolves the consultation of one expert in order to derive an estimate for the project based on his experience and available information about the project under development. This technique has been used extensively. However, the estimates are produced in intuitive and non-explicit way [4]. It means that the component should provide the functions and services as per the requirement when used under the specified condition. Pre existing components with or minimum changes will allow low cost, faster delivery of end product. If the functionality is high then the efforts invested are also high [12]. Software development effort estimation is the process of predicting the most realistic use of effort required for developing software based on some parameters. It has always characterized one of the biggest challenges in Computer Science for the last decades. Because time and cost estimate at the early stages of the software development are the most difficult to obtain and they are often the least accurate [6]. The precision and reliability of the effort estimation is very important for software industry because both overestimates and underestimates of the software effort are harmful to software companies. Nevertheless, accurate estimation of software development effort in reality has major implications for the management of software development [2]. Some major aspects of estimations are Nueral Network, project management, regression modeling and so on. Project development has increased importance in business due to stiff competition and a fast changing business environment. Its cycle time is critical as it decides the success of a business. Regardless of scope or complexity, a series of stages are Initiation, in which the outputs and critical success factors are defined, followed by a Planning phase, characterized by breaking down the project into smaller parts/tasks, an Execution phase, in which the project plan is executed, and lastly a Closure Project activities must be grouped into phases or Exit phase, that marks the completion of the project [15]. Evidence-based Software Engineering is to provide the means by which current best evidence from research can be integrated with practical experience and human values in the decision making process regarding the development and maintenance of software [3]. Some Systematic rules are, existing evidence regarding a treatment of technology, to review existing empirical evidence of the benefits and

limitations of a specific Web development method, identify gaps in the existing research that will lead to topics for further investigation, provide context/framework so as to properly place new research activities [1].

Prediction of software development effort using Artificial Neural Networks has focused mostly on the accuracy comparison of algorithmic models rather than on the suitability of the approach for building software effort prediction systems.The case study considering real time software to predict the testability of each module from source code static measures. They consider Artificial Neural Networks as promising techniques to build predictive models [7]. Artificial neural networks method has a potential to be the most appropriate technique for early stage estimation. This is because neural networks, unlike linear aggression, are able to model interdependencies between input data which will inevitably occur when considering the significant variables on the construction such as number of storeys, floor area and number of lifts [14]. The challenge then lies in accurately modeling and predicting the software development effort, and then create project development schedule. This work employs a neural network (NN) approach and a multiple regression modeling approach to model and predict the software development effort based on an available real life dataset [13] Future software cost estimation research increase the breadth of the search for relevant studies, Search manually for relevant papers within a carefully selected set of journals when completeness is essential, Conduct more research on basic software cost estimation topics, Conduct more studies of software cost estimation in real-life settings, Conduct more studies on estimation methods commonly used by the software industry and Conduct fewer studies that evaluate methods based on arbitrarily chosen data sets [8] most of the available data cost estimation and found that above preposition develops the best way of finding cost estimation at the requirement phase itself. This will be appreciated if judgment is drawn on some worked out project and their cost estimation [10]. This gives rise to an interesting dilemma in software cost estimation: the quality of inputs starts becoming better only when the outputs have started losing value. Addressing this dilemma, which represents a major problem in the field of software cost estimation, is tantamount to addressing the problem of poor estimation accuracy during the early phases of software development. The quality of inputs during these early phases is difficult to improve due to the limited amount of information available at that time [9].

II. ANALOGY-BASED ESTIMATION

Estimation by Analogy presented the best prediction accuracy when using a dataset of Web hypermedia applications. Estimation by Analogy is an intuitive method and there is evidence that experts apply analogic reasoning when making estimates. Estimation by Analogy is simple and flexible, compared to algorithmic models. Estimation by Analogy can be used on qualitative and quantitative data, reflecting closer types of datasets found in real life [26]. Analogy-based estimation is able to deal with poorly understood domains that are difficult to model, since it relies on historical data of projects similar to the target project rather than on a formal model, or rules in the case of rule-based systems. It can be applied in the very early phase of a software project when detailed information about the project is not yet available, and can be later improved when more detailed information is accessible. Analogy-based estimation has the potential to mitigate the effect of outliers in a historical data set, since estimation by analogy does not rely on calibrating a single model to suit all projects [22]. Both Ant colony optimization (ACO) and Genetic algorithms (GA) use a population of agents or individuals to represent solutions, and the information collected by the population influences the next generation of the search.

III. REGRESSION-BASED SOFTWARE COST ESTIMATION MODEL

An important difference of ACO over GAs is that the information maintained in the artificial pheromone trails represents the memory of the entire colony from all generations, whereas the information on the performance of the search is contained only in the current generation of a GA.The main advantage of ACO over GAs is that in every new cycle of the search, solutions are developed from the collective information maintained in the pheromone trails [25]. The COCOMO model is a regression-based software cost estimation model. It was developed by Bohem (1995; 2000) in 1981 and thought to be the most cited, best known and the most plausible (Fei and Liu, 1992) of all traditional cost prediction models. COCOMO model can be used to calculate the amount of effort and the time schedule for software projects. COCOMO 81 was a stable model on that time. One of the problems with using COCOMO 81 today is that it does not match the development environment of the late 1990's [24]. COCOMO represents an approach that could be regarded as "off the shelf." Here the estimator hopes that the equations contained in the cost model adequately represent their development environment and that any variations can be satisfactorily accounted for in terms of cost drivers or parameters built into the model.

For instance COCOMO has 15 such drivers. Unfortunately, there is considerable evidence that this "off the shelf" approach is not always very successful. Kemerer reports average errors (in terms of the difference between predicted and actual project effort) of over 600 percent in his independent study of COCOMO. Analogy-based systems can also handle failed cases (i.e., those cases for which an accurate prediction was not made). This is useful as it enables users to identify potentially high-risk situations. Analogy is able to deal with poorly understood domains (such as software projects) since solutions are based upon what has actually happened as opposed to chains of rules in the case of rule based systems. Users may be more willing to accept solutions from analogy based systems since they are derived from a form of reasoning more akin to human problem solving, as opposed to the somewhat arcane chains of rules or neural nets. This final advantage is particularly important if systems are to be not only deployed but also have reliance placed upon them [23].

3.1 RBFN NETWORK STRUCTURE

A handful of researches have been presented in the literature for the software cost detection using software development estimation Recently, utilizing artificial intelligence techniques like Neural Network, Fuzzy and web development in software cost estimation have received a great deal of attention among researchers. A brief review of some recent researches is presented here Ali Idri *et al.* [4] has suggested that the several software effort estimation models

developed over the last 30 years, providing accurate estimates of the software project under development was still unachievable goal. Therefore, many researchers were working on the development of new models and the improvement of the existing ones using artificial intelligence techniques such as: case-based reasoning, decision trees, genetic algorithms and neural networks. This paper was devoted to the design of Radial Basis Function Networks for software cost estimation. It shows the impact of the RBFN network structure, especially the number of neurons in the hidden layer and the widths of the basis function, on the accuracy of the produced estimates measured by means of MMRE and Pred indicators.

3.2 STRUCTURE REVIEW

Stein Grimstad *et al*. [5] has documented that the software industry suffers from frequent cost overruns. A contributing factor was the imprecise estimation terminologies have been used. A lack of clarity and precision in the use of estimation terms reduces the interpretability of estimation accuracy results, makes the communication of estimates difficult, and lowers the learning possibilities. This paper reports on a structured review of typical software effort estimation terminology in software engineering textbooks and software estimation research papers. The review provides evidence that the term 'effort estimate' was frequently used without sufficient clarification of its meaning, and that estimation accuracy have often evaluated without ensuring that the estimated and the actual effort was comparable. Guidelines were suggested on how to reduce that lack of clarity and precision in terminology. B. Tirimula Rao *et al*. [7] proposed that the software cost estimation was critical for software project management. Many approaches have been proposed to estimate the cost with current project by referring to the data collected from past projects. We discuss an approach for the validation of the dataset for training the neural network for the software cost estimation. By combining the mathematical approach, it gives an base model for our validation procedure, we get much more accurate results when compared to the primitive ones .By tracking the results with the standard ones we calculate the error percentile which was proved to be very efficient than artificial neural networks and which simplifies the operations of artificial neural networks

3.3 FUTURE SOFTWARE COST ESTIMATION

Magne Jargensen and Martin Shepperd [8] have proved that the basis for the improvement of software estimation research through a systematic review of previous work. The review identifies 304 software cost estimation papers in 76 journals and classifies the papers according to research topic, estimation approach, research approach, study context and data set. Based on the review, we provide recommendations for future software cost estimation research: 1) Increase the breadth of the search for relevant studies, 2) Search manually for relevant papers within a carefully selected set of journals when completeness is essential, 3) Conduct more research on basic software cost estimation topics, 4) Conduct more studies of software cost estimation in real-life settings, 5) Conduct more studies on estimation methods commonly used by the software industry, and, 6) Conduct fewer studies that evaluate methods based on arbitrarily chosen data sets.Kirti Seth *et al*. [12] has pointed that the Effort Estimation with good accuracy helps in managing overall budgeting and planning. The accuracy of those estimates was very less and most difficult to obtain, because no or very little detail about the project were known at the beginning. Due to architectural difference in CBS (Component Based Systems), the estimation becomes more crucial. Component-based development mainly involves the reuse of already developed components. The selection of best quality component is of prime concern for developing an overall quality product.CBS mainly involves two types of efforts: selection and integration. Present paper presents a fuzzy rule based model for estimating the efforts in selecting those Components for developing an application using CBSE approach.This simulator will be an asset to affordably keep track of time during the process of development and thus to satisfy the client in those era of competitive market of software.Maya Daneva and Roel Wieringa [18] have concluded that many methods for estimating size, effort, schedule and other cost aspects of IS projects, but only one specifically developed for Enterprise Resource Planning(ERP) and none for simultaneous, interdependent ERP projects in a cross-organizational context. The objective of those papers was to sketch the problem domain of cross-organizational ERP cost estimation, to survey available solutions, and to propose a research program to improve those solutions. In it, we explained why knowledge in the cost estimation of cross-organizational ERP was fragmented, assess the need to integrate research perspectives and propose research directions that an integrated view of estimating cross-organizational ERP project cost should include.Jaswinder Kaur *et al*. [19] estimated that software development effort is an important task in the management of large software projects. The task was challenging and it has been receiving the attentions of researchers ever since software was developed for commercial purpose. A number of estimation models exist for effort prediction. However, there was a need for novel model to obtain more accurate estimations. The primary purpose of those study, propose a precise method of estimation by selecting the most popular models in order to improve accuracy. Here, we explore the use of Soft Computing techniques to build a suitable model structure to utilize improved estimation of software effort for NASA software projects. A comparison between Artificial-Neural-Network Based Model (ANN) and Halstead, Walston-Felix, Bailey-Basili and Doty models was provided. The evaluation criteria were based upon MRE and MMRE. Consequently, the final results very precise and reliable when they are applied to a real dataset in a software project. .The results show that Artificial Neural Network were effective in effort estimation. Bingchiang jeng *et al* [20] has guided that the software estimation provides an important tool for project planning; whose quality and accuracy greatly affect the success of a project. Despite a plethora of estimation models, practitioners experience difficulties in applying them because most models attempt to include as many influential factors as possible in estimating software size and/or effort. This research suggests a different approach that simplifies and tailors a generic function point analysis model to increase ease of use. The proposed approach redefines the function type categories in the FPA model, on the basis of the target application's characteristics and system architecture. This method makes the function types more suitable for the particular application domain. It also enables function point counting by the programmers themselves instead of by an expert. An empirical study using historical data establishes the regression model and demonstrates that its prediction accuracy was comparable to that of a FPA model.

IV. CURRENT PROPOSED METHODOLOGY DURING THE TENURE OF THE RESEARCH WOR

ACTION	DESCRIPTION	RESPONSIBILITY	SUMMARY
Gather and Analyze Software Functional & Programmatic Requirements	Analyze and refine software requirements, software architecture, and programmatic constraints.	Software manager, system engineers, and cognizant engineers	• Identified constraints • Methods used to refine requirements • Resulting requirements • Resulting architecture hierarchy • Refined software architecture • Refined software functional requirements
Define the Work Elements and Procurements	Define software work elements and procurements for specific project.	Software manager, system engineers, and cognizant engineers	• Project-Specific product-based software WBS • Procurements • Risk List
Estimate Software Size	Estimate size of software in logical Source Lines of Code (SLOC).	Software manager, cognizant engineers	• Methods used for size estimation • Lower level and total software size estimates in logical SLOC
Estimate Software Effort	Convert software size estimate in SLOC to software development effort. Use software development effort to derive effort for all work elements.	Software manager, cognizant engineers, and software estimators	• Methods used to estimate effort for all work elements • Lower level and Total Software Development Effort in work-months (WM) • Total Software Effort for all work elements of the project WBS in work months • Major assumptions used in effort estimates
Schedule the effort	Determine length of time needed to complete the software effort. Establish time periods of work elements of the software project WBS and milestones	Software manager, cognizant engineers, and software estimators	• Schedule for all work elements of project's software WBS • Milestones and review dates • Revised estimates and assumptions made
Calculate the Cost	Estimate the total cost of the software project	Software manager, cognizant engineers, and software estimators	• Methods used to estimate the cost • Cost of procurements • Itemization of cost elements in dollars across all work elements • Total cost estimate in dollars

Software cost estimation is the process of predicting the effort to be required to develop a software system. The Software cost estimation process included a number of iterative steps which can be summarized in Table I.
Totally there are six steps needed for the software cost estimation process. They are,

1. Gather and Analyze Software Functional and Programmatic Requirements
2. Define the Work Elements and Procurements
3. Estimate Software Size
4. Estimate Software Effort
5. Schedule the Effort
6. Calculate the Cost

The limitations of algorithmic models led to the exploration of the non-algorithmic techniques which are soft computing based. New paradigms offer alternatives to estimate the software development effort, in particular the Computational Intelligence (CI) that exploits mechanisms of interaction between humans and processes domain knowledge

with the intention of building intelligent systems (IS). Amongst IS, fuzzy logic may be a convenient tool for software development effort estimation. Fuzzy logic-based cost estimation models are more appropriate when vague and imprecise information is to be accounted for. Fuzzy systems try to behave just like the processes of the brain with a rule base. In our proposed method the effort and the cost can be estimated using the Fuzzy logic. In fuzzy logic, there are three steps, 1) Fuzzification: to produce trapezoidal numbers for the linguistic terms 2) to develop the complexity matrix by producing a new linguistic term 3) to determine the productivity rate and the attempt for the new linguistic terms 4) Defuzzification: to determine the effort required to complete a task. After the estimation of effort, the cost for the software development process can be estimated. The implementation of the proposed work will be done in JAVA.

V. CONCLUSION

The limitations of algorithmic models led to the exploration of the non-algorithmic techniques which are soft computing based. New paradigms offer alternatives to estimate the software development effort, in particular the Computational Intelligence (CI) that exploits mechanisms of interaction between humans and processes domain knowledge with the intention of building intelligent systems (IS). Amongst IS, fuzzy logic may be a convenient tool for software development effort estimation. Fuzzy logic-based cost estimation models are more appropriate when vague and imprecise information is to be accounted for. Fuzzy systems try to behave just like the processes of the brain with a rule base. In our proposed method the effort and the cost can be estimated using the Fuzzy logic. In fuzzy logic, there are three steps, 1) Fuzzification: to produce trapezoidal numbers for the linguistic terms 2) to develop the complexity matrix by producing a new linguistic term 3) to determine the productivity rate and the attempt for the new linguistic terms 4) Defuzzification: to determine the effort required to complete a task. After the estimation of effort, the cost for the software development process can be estimated. The implementation of the proposed work will be done in JAVA. In our future work, we aim to perform experiments on the other personality factors (agreeableness and neuroticism), other emotions of OCC, and other parameters of human modeling such as culture and motivational states. Fixed values for emotions were considered in this study and they initialized the mood only without any affect during the simulation. Implementing a computational emotional model on the basis of OCC to compute the intensity of emotions with respect to the events and TMA's actions during the simulation is another work for future. In this study, the role of team members did not affect the computation. Roles only affect the direction of communications not the quality of the communications. Hence, the effect of the role assignment (i.e. Belbin model [26]) will be assessed in computations. The lack of data is one of the hindrances in our present work. We will be testing the FECSCE with more data in future. In this study, project difficulty was considered without change during project execution time. When a project progresses, there are more developed CASE development tools which can be reused [27]; hence, the project difficulty may encounter change during the project progression. We want to consider this aspect also in future work.

REFERENCES

1. Barbara A. Kitchenham, Emilia Mendes and Guilherme H. Travassos," Cross versus Within-Company Cost Estimation Studies: A Systematic Review," IEEE Transactions on Software Engineering, Vol. 33, No. 5, pp. 316-329, "May 2007.
2. Ch. Satyananda Reddy and KVSVN Raju," Improving the Accuracy of Effort Estimation through Fuzzy Set Representation of Size," Journal of Computer Science, Vol. 5, No.6, pp. 451-455, 2009
3. Barbara Kitchenham, O. Pearl Brereton , David Budgen, Mark Turner, John Bailey and Stephen Linkman," Systematic literature reviews in software engineering – A systematic literature review," Information and Software Technology, Vol. 51, pp.7–15, 2009.
4. Ali Idri, Abdelali Zakrani and Azeddine Zahi," Design of Radial Basis Function Neural Networks for Software Effort Estimation," IJCSI International Journal of Computer Science Issues, Vol. 7, No. 3, pp. 11-18, July 2010.
5. Stein Grimstad, Magne Jorgensen and Kjetil Molokken-stvold," Software effort estimation terminology: The tower of Babel," Information and Software Technology, Vol. 48, pp. 302–310, 2006
6. Iman Attarzadeh and Siew Hock Ow," A Novel Algorithmic Cost Estimation Model Based on Soft Computing Technique," Journal of Computer Science, Vol. 6, No. 2, pp 117-125, 2010.
7. B. Tirimula Rao, B. Sameet, G. Kiran Swathi, K. Vikram Gupta, Ch. Ravi Teja and S.Sumana," A Novel Neural Network Approach For Software Cost Estimation Using Functional Link Artificial Neural Network (FLANN)," IJCSNS International Journal of Compute Science and Network Security, Vol.9, No.6, pp. 126-131, June 2009.
8. Mange Jorgensen and Martin Shepperd," A Systematic Review of Software Development Cost Estimation Studies," IEEE Transactions on Software Engineering," Vol. 33, No.1, pp. 33 - 53, Jan. 2007.
9. Ali Afzal Malik and Barry Boehm," Quantifying requirements elaboration to improve early software cost estimation," Information Sciences, 2009.
10. Surya Prakash Tripathi, Kavita Agarwal and Shyam Kishore Bajpai," A note on cost estimation based on prime numbers," International Journal of Information Technology and Knowledge Management, Vol. 2, No. 2, pp. 241-245 Dec. 2010.
11. K. Vinay Kumar, V. Raves, Mahil Carr and N. Raj Kiran," Software development cost estimation using wavelet neural networks," The Journal of Systems and Software, 2008.
12. Kirti Seth, Arun Sharma and Ashish Seth," Component Selection Efforts Estimation– a Fuzzy Logic Based Approach," International Journal of Computer Science and Security, (IJCSS) Vol. 3, No.3, pp. 210-215, 2009.
13. Roheet Bhatnagar, Vandana Bhattacharjee and Mrinal Kanti Ghose," Software Development Effort Estimation–Neural Network vs. Regression Modeling Approach," International Journal of Engineering Science and Technology, Vol. 2, No.7, pp.2950-2956, 2010.

14. Mohammed arafa and Mamoun Alqedra," Early stage cost estimation of building construction projects using artifical neural networks," Journal of Artifical Intelligence, Vol.4, No.1, pp. 63-75, 2011.
15. P. K. Suri, Bharat Bhushan and Ashish Jolly," Time Estimation for Project Management Life Cycle: A Simulation Approach," IJCSNS International Journal of Computer Science and Network Security, Vol.9, No.5, pp. 211-216, May 2009.
16. Naveen Aggarwal, Nupur Prakash, and Sanjeev Sofat," Content Management System Effort Estimation Model based on Object Point Analysis," International Journal of Electrical and Electronics Engineering," Vol.2, No.8, pp.483-490, 2008
17. P. K. Suri and Bharat Bhushan," Simulator for Time Estimation of Software Development Process," IJCSNS International Journal of Computer Science and Network Security, Vol.7, No.7, pp 288-292, Jul 2007.
18. Maya Daneva and Roel Wieringa," Cost estimation for cross-organizational ERP projects: research perspectives," software quality journal, Vol. 16, No. 3, pp. 459-481, 2008.
19. Jaswinder Kaur, Satwinder Singh, Dr. Karanjeet Singh Kahlon and Pourush Bassi," Neural Network-A Novel Technique for Software Effort Estimation," International Journal of Computer Theory and Engineering, Vol. 2, No. 1 Feb. 2010.
20. Bingchiang jeng, dowming yeh, deron wang, shu-lan chu and chia-mei chen," A Specific Effort Estimation Method Using Function Point," Journal of information science and engineering, Vol. 27, pp.1363-1376, 2011.
21. Dheerendra Singh and K. S. Dhindsa, "Estimation By Analogy In The Perspective Educational Web Hypermedia Applications", International Journal of Information Technology and Knowledge Management, Vol. 4, No. 1, pp. 305-313,Jan 2011.
22. Jingzhou Li, Guenther Ruhe, Ahmed Al-Emran and Michael M. Richter, "A flexible method for software effort estimation by analogy", Journal Empirical Software Engineering, Vol. 12, No. 1, pp.65-106, Feb 2007.
23. Martin Shepperd and Chris Schofield, "Estimating Software Project Effort Using Analogies", IEEE Transactions On Software Engineering, Vol. 23, No. 12, pp. 736-743, Nov 1997.
24. Iman Attarzadeh and Siew Hock Ow, "A Novel Algorithmic Cost Estimation Model Based on Soft Computing Technique", Journal of Computer Science, Vol.6, No. 2, pp. 117-125, 2010.
25. Khaled Mahar and Ahmed El-Deraa, "Software Project Estimation Model Using Function Point Analysis With Cbr Support", International Conference Applied Computing, Alexandria, Egypt,pp. 508-512, 2006.
26. R. Belbin, Team Roles at Work, Butterworth- Heinemann, Oxford, 1983
27. R. Banker, R. Kauffman, R. Kumar, An empirical test of object-based outputmeasurement metrics in a computer aided software engineering (CASE) environment, Journal of Management Information Systems 8 (1994) 127–150.

AUTHOR PROFILE

1) Mr.S.A.Rameshkumar , M.Tech(IT)., pursuing Ph.d. as an Asst.professor from Department of CSE of Karpaga Vinayaga College of Engineering Bharath University, Chennai, Tamil Nadu. He has more than 9 years of teaching and his areas of specialization are mobile computing, Artificial Intelligence Operating System, software Engineering and Data Structure .

2) Dr.S.Sivasubramanain,M.Tech(CSE)., Ph.D(CSE) as anProfessor from Department of IT of Dhanalakshmi College Of Enginnering, Chennai,Tamil Nadu. He has more than 12 years of teaching and research experience and his areas of specialization are mobile computing,Database Management System, Computer Networks, Networks Security and Data Mining.