International Journal of Research and Development in Applied Science and Engineering (IJRDASE)
ISSN: 2454-6844

# A Review on Software Effort Estimation by Simulated Annealing

**Kausar Husain Siddiqui**
Department of Computer Science Engg.
L.I.T., Lucknow (U.P.), India
erkausar@gmail.com

**Vipin Jaiswal**
Department of Computer Science Engg.
L.I.T., Lucknow (U.P.), India
dir@litlucknow.ac.in

**Abstract: Software effort estimation is the task of approximating the amount of work required to develop a software project. The reasons for effort estimation include project approval, project management, understanding by the development team members and defining the project tasks. Software effort estimation is not a precise science. Accurately estimating software effort has become a great issue for software engineers. Estimates done at the proposal stage has high degree of inaccuracy, where requirements for the scope are not defined to the lowest details, but as the project progresses and requirements are elaborated, accuracy and confidence on estimate increases. Clearly there is a strong need for better software effort estimation methods. This thesis focuses on the use of the COCOMO model and variations of this model due to the public description of the algorithm, available data, as well as prior use and research by the research client, NASA's Jet Propulsion Laboratory. COCOMO is a popular model, developed by Barry Boehm of USC, which defines a linear relationship between effort and code size. The model includes several cost drivers, such as programmer capability and required reliability, which affect the estimate either as a linear scalar value, or as a slight exponential change.**

**Keywords: COCOMO, Effort Estimation, Simulated Aniline, Software Engineering.**

## 1. Introduction:

In software engineering effort is used to represent measure of use of workforce and is defined as aggregate time that takes members of a development team to accomplish a given task. It is usually uttered in units such as man-day, man-month, man-year. It should be mentioned that these categories do not exclude one another. Furthermore, estimation methods exist which, in respect of some of their features, belong to different groups. The objective of this chapter is to present the most relevant methods and models for effort estimation used by software engineers in the past. Software effort estimation spawned some of the first attempts at meticulous software measurement, so it is the oldest, most mature aspect of software metrics. Considerable research had been carried out in the past, to come up with a variety of effort prediction models using empirical parametric (algorithmic), expertise-based and learning-based (non-algorithmic techniques). This chapter discusses the evolution of these techniques eventually.

## 2. Related Work:

Adanma C. Eberendu, (2014), gave an overview of software cost estimation and answered the following questions: (1) what have people already done in area of cost estimation; (2) what are the types of software cost estimation model available; (3) how is the contingency allowance accounted for in software projects? Since none of the existing software cost estimation model can work accurately on its own, they proposed a hybrid model to bridge the gap and arrive at accurate and acceptable estimates.

Shiyna Kumar, Vinay Chopra, (2013), presented an overview of the different techniques currently available for software effort estimation in the software industry. Software effort estimation is a incredibly essential task in the software engineering field because the future of the project depends on the estimation report. The techniques discussed about algorithmic model, non algorithmic model and some soft computing technique. Finally it was concluded that it is not only the metrics which can be responsible for accurate estimation, but also it is how and when they are being used. So, one cannot say a specific technique is best fit for all the situations to give an accurate estimation.

Prabhakar et. al., (2013), they used Artificial Neural Network (ANN), and Support Vector Machine (SVM) learning techniques to analyze the results using China dataset for predicting software development effort. A similar study can be carried out to predict software effort using prediction models based on other machine learning algorithms such as Genetic Algorithms (GA) and Random Forest (RF) techniques. Cost benefit analysis of models may be carried out to determine whether a given effort prediction model would be economically viable.

Srinivasa Rao T, et. al. (2013), proposed software cost estimation based on PSO technique. Neural network technique was used to train the network for classification of values during testing phase. The porposed model could be applied to large datasets. It was also seen that the model could be very effective if it contained similar genres. However it was seen that PSO is a probabilistic model which could not generate exact values. But based on enough historical datasets the accuracy could be increased.

Syed Ali Abbas, et. al. (2012), dealt with a detailed discussion about the models that were presented earlier in various studies. It contains many parametric and non parametric modeling techniques. Thus it clearly gives a detailed analysis of the pros and cons, similarities and differences amongst the models. From the analysis it was concluded that any approach based on rigid mathematical formula can not alone solve the issue of effort estimation. However, expert based techniques be of great help in this case and can be validated by the application of some parametric models and soft computing techniques.

Magne Jørgensen, et. al. (2012), explored the effects, i.e., "first impression" or "recency," that is likely to be the stronger on model development. Here four experiments were carried out. It contained the following steps. The first step included the consideration of the low or high reference effort values, thus creating a baised impression through comparative study of the work to be estimated. The second step included the reference effort values in the opposite direction, i.e. to say that comparison of the task was carried out in reverse to what was done in step one, i.e. changinh the low reference value to high and vice versa. The third step included the use of most likely effort. The results showed that the first impression was stronger than the next one but there was clearly seen effect from the more recent comparisons. Thus the crux of the matter is that for software developers it is essential that their first impression for the development of effort should be based on some standard reference values because first impression are easily seen but are hard to be forgotten. This would ultimately lead to a stronger impact rather than wrong first impression.

S.K. MOHANTY & A.K. BISOI, (2012), dealt with a generalized software estimation modeling techniques. It showed that models can be categorized as Size-Based, Function-Based, Learning-Based and Expertise-Based. The first two are known as parametric models, using a function or a formula for effort estimation. Each one has its own merits and demerits. However no single technique is suited to all the situations. Hence there is need to consider all the techniques for a better solution.

Tirimula Rao Benala, (2012), discussed a few important computational intelligence techniques that have been used for software cost estimation. CI techniques are generally considered as promising techniques because a large number of generalization factors are connected with software projects. Spurious attributes or missing attributes can lead to erroneous conclusions for CI techniques. Therefore researcher should clearly specify the testing and validation methodology followed for particular data sets, while claiming their method as superior to contemporary ones. It was found that many CI techniques are still to be investigated for the effectiveness in the field of Software Cost estimation.

Sanjay Kumar Dubey, et.al. (2012), , provided a mechanism for comparing all the object oriented software metrics which define all the methods, attributes are used in software engineering environment. The increase is software development means the measurement was also so high. The increasing significance being placed software measurement which has to lead and increase amount of research on developing the new software measures. In this paper, all of the software metrics for object oriented development has been presented. They provided a basis for measuring all of the characteristics like size, complexity, performance and quality. In rely of some notions the quality may be increased by added some features like abstraction, polymorphism and inheritance which are inherent in object orientation. This paper provides some help for researchers and practitioners for better understanding and selection of software metrics for their purposes.

Mohammad Saber Iraji, et.al. (2012), Used case size point (USP) method to approximate object oriented software development effort in the premature phase of software project and applied in many software organizations. USP is calculated by calculating the number of actors, prerequisites, post conditions, scenarios built-in in use case models. They have shown Adaptive fuzzy Neural Network model to approximate the effort of object oriented software using Use Case size Point technique. Here adaptive neural network fuzzy used case size point has minimum error and system worked more accurately and appropriately than prior methods.

Jovan Zivadinovic, et. al. (2011), presented the most relevant methods and models for effort estimation used by software engineers in the past four decades. Classification of the methods has been also suggested as well as brief description of the estimation methods presented. In the past four decades a great number of different models and effort estimation methods have been developed. This clearly indicates the awareness among the researchers of the need to improve effort estimation in software engineering. Unfortunately, the fact remains that even though, all the effort invested by the researchers yielded no result as they wished for and, even today, effort estimation still remains rather unreliable. Whenever estimations are made one actually looks to the future, and naturally, accepts certain level of uncertainty and risk. Risks happen as result of insufficient information, which one cannot know in advance. This is the point where good historic background becomes revealed. Risk is measured by uncertainty level in quantitative estimates of resources, costs and distribution. One should not expect effort estimation to be ever an exact science. Many factors have impact on the software development process. These factors can be human, technical but also political and their impact can never be fully predicted. This should no way be understood as a call to give up estimating because even insufficiently accurate estimates are far better than none.

Joao Carlos Cunha, et. al. (2011), presented the estimation procedures implemented at Critical Software SA (CSW), a medium-sized company that is currently recognized as CMMI Level 5. Starting from an analysis of the company needs, including several interviews with key people, two estimation approaches were implemented: bottom-up estimation, using 3-point estimates, and Wideband Delphi. Found that these techniques should be applied during tendering, at the beginning of the project, after requirements analysis, and whenever a major change occurs. The use of bottom-up estimation is mandatory, while Wideband Delphi should be used to complement estimates in projects for which there is much uncertainty. During implementation, special attention was put on the dissemination and training activities, with the goal of fostering acceptance and usage. Preliminary results showed that software effort estimation can be successfully applied in medium-sized companies like CSW, at very low cost. This allows reducing projects' uncertainty and increasing the probability of success during bidding and execution.

Vahid Khatibi B.(2011), et. al. applied a hybrid system based on linking C-Means clustering, neural network and analogy method. Since, there are elaborate and nonlinear relations among software project features, the suggested method can be beneficial to interpret such relations and to present extra accurate estimations. The obtained results exposed that fuzzy clustering could decrease the negative effect of inappropriate projects on accuracy of estimations. In addition, evaluation of suggested hybrid method showed the significant perfection of accuracy as related to the neural network the analogy method and statistical methods.

Qinbao Song, (2011), recommended a novel methodology of utilizing Gray Relational Analysis of Gray System Theory to address outlier detection, feature subset selection and software effort prediction at an early stage of a software development process. Using five publicly available data sets, compared the suggested method with prediction models based on stepwise regression and Analogy, and when available, also with NW GRA, LW GRA, ANN, CART, GRA, and GP methods. The results showed that the proposed method outperformed (1) stepwise regression and Analogy in terms of MMRE, MdMRE, and PRED for the majority (4 out of 5) of data sets used; (2) other methods in terms of MMRE and PRED excepting for a 4.2% PRED smaller than GRA with one data set. The results also showed that the proposed outlier detection method can not only improve the performance of the proposed effort prediction method but also the performance of SWR and Analogy. This is very encouraging and indicates that the method has considerable potential.
.
Carolyn Mair, (2011), reviewed the uptake of past software project prediction research and identify relevant cognitive psychology research on expert behaviour. In particular explored potential applications of recent metacognition research. It was found that the human aspect is largely ignored, despite the availability of many important results from cognitive psychology.
In order to increase the actual use of our metrics research e.g. effort prediction systems one needs to have a more integrated view of how such research might be used and who might be using it. This leads to belief that future research must be more holistic and inter-disciplinary.

MOHAMMAD AZZEH, et. al., (2011) proposed a new evolutionary algorithm model based on the integration of Fuzzy set theory with Grey Relational Analysis (GRA). This was done to cater to the uncertainty in effort estimation calculation, resulting in low accuracy, mainly due to the reason that these attributes are based on human judgments, which are more or less imprecise and vague. Fuzzy set theory reduces uncertainty in measuring the distance between two tuples. GRA, on the other hand, for project for both continuous and categorical features, reduces uncertainty in the distance measure between two software projects. Both these methods are suited to conditions when the relation between effort and effort drivers are complex. Comparing the results of the above two techniques with the case based, regression and artificial neural network techniques, it was found that the estimates so calculated were very good and accurate.

Ekrem Kocaguneli,, et.al. (2010), developed an effort estimation model for a multinational bank. They tried to predict the effort before the start of the project development cycle. For this product, process and resource metrics from earlier projects were collected, along with the effort values spread amongst various software project life cycles. They used Support Vector Regression (SVR) to predict the effort. Also clustering approach was used for the formation of consistent project groups. The whole approach was validated using real life problems.

Prasad Reddy P.V.G.D,(2010), used artificial intelligence technique like artificial neural network for the development of effort estimation model. For this COCOMO model was used. The model was developed using Radial Basis and Generalized Regression functions. Comparative study was carried out using COCOMO81 and COCOMO intermediate data model. For this MMRE, MARE, VARE, Mean BRE and Prediction were used as performance evaluation criteria. Overall it was seen that radial basis neural network gave the best performance.

Saleem Basha, et. al. (2010), stated that the existing models were highly credible; however, this survey found this not to be so based upon the research performed. All the models could not predict the actual against either the calibration data or validation data to any level of accuracy or consistency. Surprisingly, SEER and machine learning techniques were reliable good at predicting the effort. But however they are not accurate because all the model lies in the term prediction, prediction never comes true is proved in this estimation models. In all the models, the two key factors that influenced the estimate were project size either in terms of LOC or FP and the capabilities of the development team personnel. It was also found that no model is so sensitive to the abilities of the development team can be applied across the board to any software development effort. Finally they concluded that the no model is best for all situations and environment.

Harsh Kumar Verma, et. al. (2010), presented a transparent, enhanced fuzzy logic based framework for software development effort prediction. The Gaussian MFs used in the framework have shown good results by handling the imprecision in inputs quite well and also their ability to adapt further make them a valid choice to represent fuzzy sets. The framework is and adaptable to the changing environments and handles the inherent imprecision and uncertainties present in the inputs quite well. The framework is augmented by the contribution of the experts in terms of modifiable fuzzy sets and rule base in accordance with the environments. The performance of the framework is demonstrated in terms of empirical validation carried on both artificial datasets and live project data of the COCOMO public database.

Ch.Satyananda Reddy and KVSVN Raju, (2010), used ANN to develop effort estimation model. The model so developed was built in such a way so as to cater to COCOMO model and lead to the improvement in the performance. Single layer feed forward neural network model was used. For training the network, back propagation learning algorithm and Resilient Back propagation algorithm (RPROP) was used. The results so obtained were compared with the actual effort. The neural network model improved the accuracy of the COCOMO model. Comparative analysis of the two models showed that models (SLANN with BP and SLANN with RPROP) works better than COCOMO. For small size projects SLANN with BP was found to perform better, but on the other hand SLANN with RPROP worked better for all types of projects.

Iman Attarzadeh, et. al. (2010), tried to develop effort estimation technique that could better perform than other existing techniques. Used neural network to handle imprecision and uncertainty. They were able to develop better estimation model by applying neural network technique on algorithmic and non-algorithmic models. Based on MMRE it gave better results as compared to COCOMO. Thus it showed that in order to cater to uncertainty and vagueness in effort drivers, it is a better technique.

Juan J. Cuadrado-Gallegoa,, et. al. (2010), presented an overview of a variety of software estimation techniques, focusing on Machine Learning and Expertise models, providing an overview of several popular estimation models currently available. From the foregoing analysis, with all the reviewed literature, it would seem that estimation by analogy offer some advantages over the other methods. It can succeed even where no statistical relationships can be found. It does not require calibration or for that matter recalibration whereas other research has shown calibration to be essential for the algorithmic methods.

Rudy Setiono, et. al. (2010), for the development of effort estimation model described a rule extraction technique, based on IF-THEN rules from a trained neural network. Using ISBSG R11 data set, they compared the performance of this techniques with linear regression and radial basis function network and CART. It was seen that better results were obtained from CART but it resulted in large number of rules, which limited its comprehensibility. Thus it was concluded that comprehensible models performed better than CART with less number of rules.

Iman Attarzadeh and Siew Hock Ow, (2009), suggested a new approach for estimating of software projects development time. The major difference between present work and previous works is that Two-sided Gaussian membership function in fuzzy technique is used for software development time estimation and then it's validated with gathered data. Here, the advantages of fuzzy logic and good generalization are obtained. The primary advantage of this model is its great interpretability by utilizing the fuzzy rules and another great advantage of this research is that it can formulate expert knowledge (fuzzy rules) project data into one general framework that may have a wide range of applicability in software estimation. The results showed that Two-sided Gaussian membership function is much better than other mentioned models. In order to achieve more accurate estimation, voting the estimated values of several techniques and combine their results maybe be useful.

Ye Yang, et. al. ,(2008), used CSBSG) database having 75 industry projects. Worked on an empirical study to provide results on effort distribution. Presented the phase effort distribution pattern. The analysis of the results showed that software size and team size on code has effect on phase effort estimation. Also there was some consistency in the deviations in requirements, design, and transition phases, compared with recommendations in the COCOMO model. Discussed some guidelined in direct effort allocation. Emperical findings from the present study are useful for conducting debate and discussion on how to improve cost estimation.

Annabella Loconsole, et. al. (2007), For finding the subjective and objective measures of requirements volatility, carried out a case study dealing with the industry. A medium sized software project with all use cases was utilised for the collection of the data. From the study it was seen that structural measure performed better than the expert judgment in estimating the total number of changes to use case based requirements. The overall study showed that mangers of any company should not rely solely on expert judgment for effort estimation and decision making.

Harish Mittal, et. al. (2007), Cost estimation process includes a progression of efficient steps that providing estimate with satisfactory risk. Two new models, based on fuzzy logic sizing, are presented here. Rather than using a single number, the software size is regarded as a triangular fuzzy number. They optimized the estimated effort for any application by varying arbitrary constants for these models. The created models were verified on 10 NASA software projects, on the basis of four criterions for assessment of software cost estimation models. Comparison of the considerable number of models was done and it is found that the created models give better estimation.

Mehwish Nasir, et. al. (2006), described in detail about the various effort estimation techniques existing in the industries and various literature. They also discussed about their strengths and weaknesses. The main emphasis was on the gap analysis of the organizations with respecdt to CMMI level 3. From the analysis of the data collected it showed that industries mainly used heuristic approaches along with Delphi technique for software effort estimation. They also suggested that formal methods of effort estimation should also be catered to. It was also shown that many techniques are used for effort estimation but none of them is fully fit for effort estimation in different categories of projects. But making proper use of all of them can give better results.

Khaled Hamdan, et. al. (2006), Here effort has been made to identify the ontology based cost estimation process framework, so as to be able to define the semantics of the project development. They showed that culture factor also affected the effort estimation and Ontology system in a group of organizations. This system provided a set of common project parameters with their understanding and their semantics. The system so developed helped the project managers to elicit software project features that are compatible with new project requirements. The system so developed used Java and RDBMS.

## 3. Conclusion

This paper presents a brief description about the various works carried out by different authors in the field of software effort estimation. The various papers published are arranged year wise, giving an overview of the different techniques currently available for software effort estimation in the software industry. Software effort estimation is a incredibly essential task in the software engineering field because the future of the project depends on the estimation report. The techniques discussed about algorithmic model, non algorithmic model and some soft computing technique. Though many researchers contributed to the literature on effort estimation, still the difficulty of effort estimation is an open challenge. Many effort estimation techniques exist in the literature, but their utilization is very particular to the development environment. Finally it may be concluded that it is not only the metrics which can be responsible for accurate estimation, but also it is how and when they are being used. So, we cannot say a specific technique is best fit for all the situations to give an accurate estimation.

## Reference:

[1] Adanma C. Eberendu, (2014), " Software Project Cost Estimation: Issues, Problems and Possible Solutions', IJESI, Volume 3 Issue 6, PP.38-43.
[2] Shiyna Kumar, Vinay Chopra,(2013), "A Review of Surveys on Estimating Software Development Effort", IJAIR Vol. 2 Issue 5, pp. 448-451.
[3] Srinivasa Rao T, et. al. (2013), "Predictive and Stochastic Approach for Software Effort Estimation", Int. J. of Software Engineering, IJSE Vol. 6 No. 1, pp. 97-115.
[4] Prabhakar et. al. , (2013), " Prediction of Software Effort Using Artificial Neural Network and Support Vector Machine", IJARSSE, Volume 3, Issue 3, pp. 40-46.
[5] Magne Jørgensen, et. al. (2012), "First Impressions in Software Development Effort Estimation:Easy to Create and Difficult to Neutralize_", Proceedings of the EASE 2012 - Published by the IET, pp. 216-222.
[6] Syed Ali Abbas, et. al. (2012), " Cost Estimation: A Survey of Well-known Historic Cost Estimation Techniques", VOL. 3, NO. 4, April 2012 ISSN 2079-8407 Journal of Emerging Trends in Computing and Information Sciences, pp. 612-636.
[7] S.K. MOHANTY & A.K. BISOI, (2012), "SOFTWARE EFFORT ESTIMATION APPROACHES – A REVIEW", International Journal of Internet Computing ISSN No: 2231 – 6965, VOL- 1, ISS- 3 2012, pp. 82-88.
[8] Sanjay Kumar Dubey, et.al., (2012), "Comparison Study and Review on Object- Oriented Metrics", Global Journal of Computer Science and Technology Volume 12 Issue 7 Version 1.0.
[9] Mohammad Saber Iraji, et.al., (2012), " Object Oriented Software Effort Estimate with Adaptive Neuro Fuzzy use Case Size Point (ANFUSP)", I.J.Intelligent Systems and Applications, 6, 14-24.
[10] Syed Ali Abbas, et. al. (2012), " Cost Estimation: A Survey of Well-known Historic Cost Estimation Techniques", Journal of Emerging Trends in Computing and Information Sciences, VOL. 3, NO. 4,, pp. 612-636.
[11] Tirimula Rao Benala, (2012), "Computational Intelligence in Software Cost Estimation: An Emerging Paradigm", ACM SIGSOFT Software Engineering Notes Page 1, Volume 37, Number 3, pp. 1-7.
[12] Carolyn Mair, (2011), "Human Judgement and Software Metrics: Vision for the Future", ICSE '11, ACM, pp. 81-84.
[13] Jovan Zivadinovic, et. al. (2011), "METHODS OF EFFORT ESTIMATION IN SOFTWARE ENGINEERING", International Symposium Engineering Management And Competitiveness 2011 (EMC2011), pp.417-422.
[14] Joao Carlos Cunha, et. al. (2011), "Implementing Software Effort Estimation in a Medium-sized Company", 34th IEEE Software Engineering Workshop, IEEE, pp. 92-96.
[15] Mohammad Azzeh, et. al., (2011) " Fuzzy Grey Relational Analysis for Software Effort Estimation", AI Research Centre, Department of Computing, University of Bradford.
[16] Qinbao Song, (2011), "Predicting software project effort: A grey relational analysis based method", Expert Systems with Applications 38 (2011) 7302–7316.
[17] Vahid Khatibi B., et. al. (2011), "A New Fuzzy Clustering Based Method to Increase the Accuracy of

Software Development Effort Estimation", World Applied Sciences Journal 1265-1275.
[18] Ekrem Kocaguneli,, et.al. (2010), "AI-Based Models for Software Effort Estimation", 36th EUROMICRO Conference on Software Engineering and Advanced Applications, IEEE, pp.323-326.
[19] Ch.Satyananda Reddy and KVSVN Raju, (2010), "An Optimal Neural Network Model for Software Effort Estimation", Int.J. of Software Engineering,IJSE Vol.3 No.1, pp. 63-78.
[20] Prasad Reddy P.V.G.D,(2010), "Software Effort Estimation using Radial Basis and Generalized Regression Neural Networks", Journal Of Computing, Volume 2, Issue 5, MAY 2010, ISSN 2151-9617 87.
[21] Saleem Basha, et. al. (2010), "Analysis of Empirical Software Effort Estimation Models", International Journal of Computer Science and Information Security, Vol. 7, No. 3,pp. 68-77.
[22] Harsh Kumar Verma, et. al. (2010), "Handling Imprecision in Inputs using Fuzzy Logic to Predict Effort in Software Development", IEEE.
[23] Iman Attarzadeh, et. al. (2010), "Proposing a New Software Cost Estimation Model Based on Artificial Neural Networks", IEEE, 2nd International Conference on Computer Engineering and Technology, PP. 487-491.
[24] Juan J. Cuadrado-Gallegoa,, et. al. (2010), ""Analogies and differences between Machine Learning and Expert based Software Project Effort Estimation,11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, IEEE, pp. 269-276.
[25] Rudy Setiono, et. al. (2010), "Software Effort Prediction using Regression Rule Extraction from Neural Networks", 22nd International Conference on Tools with Artificial Intelligence, IEEE, pop. 45-52.
[26] Iman Attarzadeh and Siew Hock Ow, (2009), "Software Development Effort Estimation Based on a New Fuzzy Logic Model", International Journal of Computer Theory and Engineering, Vol. 1, No. 4, October2009, pp.473-476.
[27] Iman Attarzadeh and Siew Hock Ow, (2009), "Software Development Effort Estimation Based on a New Fuzzy Logic Model", International Journal of Computer Theory and Engineering, Vol. 1, No. 4,, pp. 473-478.
[28] Ye Yang, et. al. ,(2008), "Phase Distribution of Software Development Effort" ESEM'08, ACM, pp. 61-69.
[29] Harish Mittal, et. al. (2007), "Optimization Criteria for Effort Estimation using Fuzzy Technique", Clei Electronic Journal, Volume 10, NUMBER 1, PAPER 2,pp.1-11.
[30] Annabella Loconsole, et. al. (2007), "Are size measures better than expert judgment?
[31] An industrial case study on requirements volatility",14th Asia-Pacific Software Engineering Conference, IEEE, PP.238-245.
[32] Mehwish Nasir, et. al. (2006), "An Empirical Study to Investigate Software Estimation Trend in Organizations Targeting CMMISM", Proceedings of the 5th IEEE/ACIS International Conference on Computer and Information Science and 1st IEEE/ACIS.
[33] Khaled Hamdan, et. al. (2006), "A Software Cost Ontology System for Assisting Estimation of Software Project Effort for Use with Case-Based Reasoning", IEEE.