IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. XII (Mar-Apr. 2014), PP 29-40
www.iosrjournals.org

Organizational Strategies and Social Interaction Influence in Software Development Effort Estimation

B V A N S S Prabhakar Rao[1] & P Seetharamaiah[2]
[1]*Research Scholar, Department of Computer Science Engineering, JNTUK, Kakinada, India*
[2]*Professor Emeritus, Department of CS & SE, College of Engineering (A), Andhra University, India*

Abstract: *In software development cost estimation, effort allocation is an important and usually challenging task for project management. This paper observes the use of concepts in software effort estimation by analyzing group work with organizational strategies as outgoing practice. The purpose is to improve our understanding of how software professionals raise different types of information when talking, way of thinking and reaching a decision on a software effort estimate. Software effort estimation is a core task regarding planning, budgeting and controlling software development projects. However, providing exact effort estimates is not easy. Estimation work is increasingly group based, and to support it, there is a need to reveal how work practices are carried out as mutual efforts. This paper contributes to an understanding of the role of concepts in group work and of software effort estimation as a specific work practice. In this paper we investigate the estimation practice through a detailed data analytical technique with machine learning approach.*

Keywords: *Effort Estimation, Group work, Interaction, Qualitative Analysis, and Organizational Strategies.*

I. Introduction

Software development effort estimation is carried out more than once in the life of software project. The project acquisition stage is the one where the detail available to estimator is minimal. The confusion of estimation arises in this stage. All other stages, progressively, increase the detail and hence reduce uncertainty in estimation. Accurate estimation of software development effort is critical because the estimated figure drives the budget to be allocated, and if this is beyond allowable limits, the project itself might not be approved. Underestimates lead to time pressures to the developers that may negotiation full functional development and painstaking testing. In contrast, overestimates can result in noncompetitive contract bids and/or over allocation of development resources and personnel [1-5].

Without the proper data and experience, software development teams usually generate inaccurate estimates of the effort required for the product to be developed. As a result, the teams are required to renegotiate with the clients to ensure that the product to be developed is within the scope achievable by the development team. Without the necessary data, it is nearly impossible for teams to make proper predictions with respect to project scope, complexity, and resources required. Typically, projects progress through their life cycles based on these inaccurate estimates. This means that regardless of how well or poorly the projects progress, the estimates remain constant. When projects begin with the initial overestimation of resources or effort required, the teams must negotiate with the clients to reduce the size of the projects. This often results in clients needing to throw away some of the critical core capabilities of the product, thus losing some of the expected benefits they had hoped for from the completed project. The reality is that the team may actually have enough resources to deliver all of the initial requirements prior to the re-scoping of the project [6, 7].

On the other hand, when projects underestimate the resources, the teams tend to over promise the goals that the project can achieve. As the project progresses towards the end of its life cycle, the team may start to realize that the remainder of the project is more than they can manage to complete. When this happens, one scenario is that they try to satisfy the client by attempting to complete the project as quickly as possible, while the quality of the project may suffer greatly from this attempt and result in higher long-term maintenance costs. Another scenario is that they end up delivering a project that is not complete, thus leaving the clients with unusable or unsustainable products [8-11].

Most systems can be seen from two viewpoints, the user's view and the developer's view. The perspective of the user relates to what the system can do for the user and function size measure support the user's perspective. The developer sees in terms of the internals of what needs to be built and is able to relate better to technical size measures [10].

When software development teams lack the proper data and experience, they can't accurately assess project size and team capabilities. These unknowns and uncertainties can typically be reduced with proper assessments as the project progresses. Unfortunately, team assessments are often overlooked, even though personnel uncertainties often have significant influence on the cone of uncertainty. For most projects, estimates

during this phase are expected to be very rough estimates. Budget estimated figures could vary between +30 % to -30% of actual numbers.

The main motivation behind this research is derived from the well-known software cone of uncertainty and calibrated to completed projects [Barry Boehm]. Good team performance and accurate software cost and schedule estimations are essential in determining the quality and timely delivery of the final product. According to Standish report few projects were delivered with full capabilities within budget and schedule. Many projects were cancelled, and were either over budget, over schedule, or under-delivered. These numbers are also consistent. This figure shows that nearly half the projects were unsuccessful due to issues related to cost and schedule estimations, and that software development projects have been consistent in replicating these shortfalls.

In software cost estimation, effort allocation is an important and usually challenging task for project management. Due to the Cone of Uncertainty effect on overall effort estimation and lack of representative effort distribution data, project managers often find it difficult to plan for staffing and other team resources. This often leads to risky decisions to assign too few or too many people to complete software lifecycle activities. As a result, projects with inaccurate resource allocation will generally experience serious schedule delay or cost overrun. [12-16]

The Standish Group estimates that in 2013 the worldwide yearly spending for software projects was $750 billion. The United States accounted for about 40% of this number or about $300 billion. Europe spent about 25% or $200 billion. Asia accounted for $100 billion. The rest of the world spent the remaining $150 billion. Canceled or failed projects were 16% or $120 billion. The United States portion was a little higher and the European portion was slightly lower. Challenged projects, those that were late, cost more and off-target, were 48% or $360 billion. Overruns vary with many legitimate reasons, but The Standish Group estimates in 2013 the cost of unintended worldwide overruns is about $80 billion; leaving the cost of worldwide project software failure to be about $200 billion. The 2009 Standish Report reported that out of the 9000 projects surveyed, 32% were delivered with full capability within budget and schedule, 24% were cancelled, and 44% were either over budget, over schedule, or under-delivered. These numbers are also consistent with the two previous reports in 2004 and 2006 [Standish Group Report].

The Standish Report on Project Success			
Year (s)	***Failed (%)***	***Challenged (%)***	***Succeeded (%)***
2013	16	48	36
2009	24	44	32
2006	19	46	35
2004	15	51	34

Successful projects did not return value to the organization or the users and executive sponsor were unsatisfied. In addition to that many challenged projects bring great value to the organization.

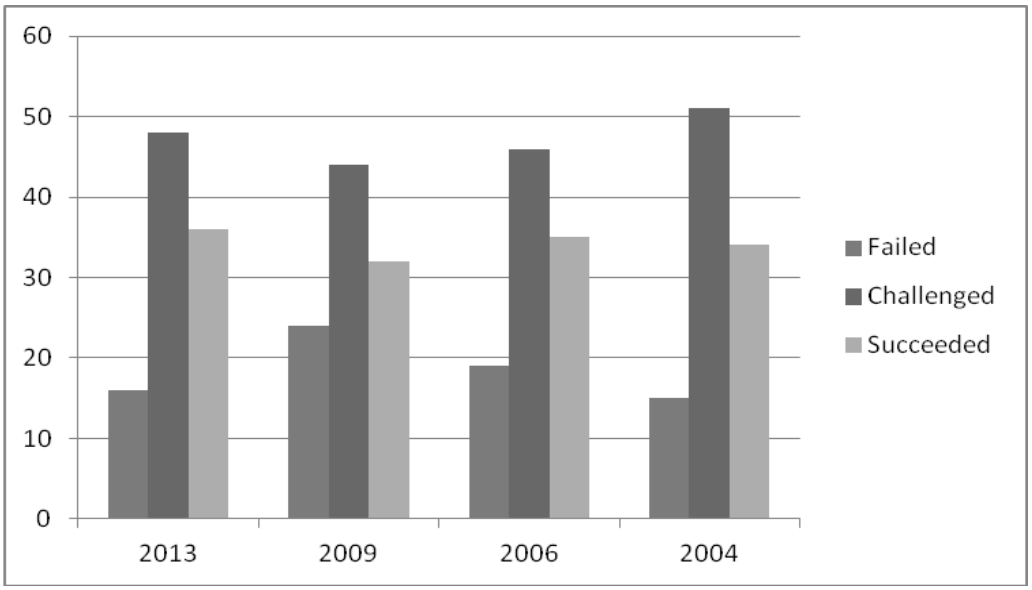

Figure1. Standish Chaos Report on Software Projects Success, Failure and Challenged Ratio

II. Software Product Development Effort Estimation is a Wicked Problem

Software Effort Estimation is a wicked problem as one that could be clearly defined only by solving it, or by solving part of it. This method assists software industries to estimate the required effort to be spent on various activities during requirements, architectural design, development, testing, deployment, operation and maintenance. We do not quote too less, programmers work for overnight that leads to lose the project or end doing social service, or loss. Do not quote too high that lose the project. So, be fair to ourselves and our customers. Hence, there is need to use of a repeatable, clearly defined and well understood software development process that has to be the most effective method. This paradox implies, essentially, that you have to solve the problem once in order to clearly define it and then solve it again to create a solution that works. There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies and the other is to make it so complicated that there are no obvious deficiencies. The phrase software design means the conception, invention, or contrivance of a scheme for turning a specification for a computer program into an operational program. Design is the activity that links requirements to coding and debugging. A good top-level design provides a structure that can safely contain multiple lower level designs. Good design is useful on small projects and indispensable on large projects. [17-30]

- It's easy to estimate what you know.
- It's hard to estimate what you know you don't know.
- It's very hard to estimate things that you don't know you don't know.

Figure 2. Similarities between Iceberg & Software Product

The Software Development Effort Estimation approach is just like the above one. Since it's development is intangible unlike other product. The iceberg image to the tip represents the software size or its functionality. The real issue is not the tip, but what is under the surface of the water and cannot be seen. The same is true when you design a software application.

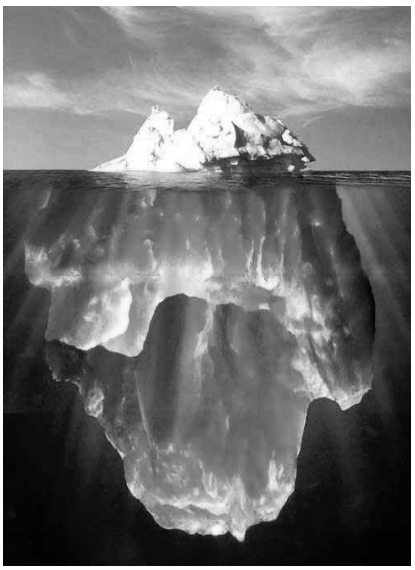

Figure 3. The Role Estimator: four images in one: sky, background, top iceberg, and underwater iceberg

The water was calm and the sun was almost directly overhead so that the diver was able to get into the water and take the picture. But how could anyone take such a picture? You could never see the underside of an iceberg that size in one shot - and where does all the light come from at that depth?

In fact, the picture is not real. Ralph Clevenger digitally composed a nature and underwater photographer who find the stories circulating about his "impossible" picture amusing. Four separate images were used; the sky, the background, the top iceberg, and the underwater iceberg. The picture does, however, accurately represent the amount of an iceberg that is hidden underwater. It was designed to illustrate the concept of "what you see is not necessarily what you get". The project tracking mechanism allows for development teams to constantly monitor the resources required for the teams to complete their project. These estimated resources can be updated as necessary depending on the team's productivity and capability. Software Project planning encompasses five major activities – estimation, scheduling, risk analysis, quality management planning, and change management planning. Estimation includes your attempt to determine how much money, effort, resources, and time it will take to build a specific software-based product. Software project managers using information solicited from project stakeholders and software metrics data collected from past projects. [31-45]

III. Organizational Strategies & Social Interaction in Decision Making

Before taking the final decision data analytical techniques have been played a vital role. Since beginning no organization started their business only with service motto. No one will do the business with loss. So they need profit. Decision-making is choosing between alternatives while having incomplete / unreliable information about the scenario at hand and with uncertain and unpredictable outcomes of the available alternatives, mainly for the sake of expediency. Decisions are made in organizations to tide over the present situation / difficulty. Therefore, sometimes decisions may render injustice. One misunderstanding that is prevalent is that, the decisions are judgments – it is far from true. We can classify decision in to the following classes for our better understanding. [46-51]

Classification of Decisions

1. Strategic & Periodic Decisions

a. Selection Decisions
- i. Products / Services
- ii. Process
- iii. Locations
- iv. Layout
- v. Equipment
- vi. Workforce

b. Design Decisions
- i. Product design
- ii. Service Design
- iii. Job Design
- iv. Process Design
- v. Control System Design
- vi. Capacity Design

2. Recurring Decisions
- a. Target Setting
- b. Scheduling
- c. Sequencing
- d. Inventory Control
- e. Cost Control
- f. Maintenance

3. Planning Decisions
- a. Planning the system
- b. Planning the usage of the system

4. Organizing Decisions
- a. Organization Structure
- b. Organizing the jobs
- c. Staffing
- d. Work and Workstation Design
- e. Standards of Performance
- f. Compensation Systems

5. Controlling Decisions

a. Quality
b. Quantity
c. Schedule
d. Inventories
e. Costs
f. Maintenance

It is not necessary that all decision makers make all the above-mentioned decisions. All of us make some of those decisions. It is perhaps, very few people – especially entrepreneurs – make all the above-mentioned decisions. Besides the following things are to be considered for social interaction:

Relationship with Customers
- ✓ Satisfaction
- ✓ On time Delivery
- ✓ Accountability
- ✓ Top Priority
- ✓ Maintenance

Benefits from other Sectors
- ✓ Govt. / R & D support
- ✓ Financial
- ✓ Tax
- ✓ Business Expansion
- ✓ Recognition

Relationship with Employees
- ✓ Training / Job Satisfaction
- ✓ Partnership / Organization Hierarchy
- ✓ Policies / Timings / Importance
- ✓ Job Recognition / Security

Therefore before finalize the price of the product they need consider the above in different perceptive. At the same time they need to satisfy the stakeholder in order to promote the product. [52-62]

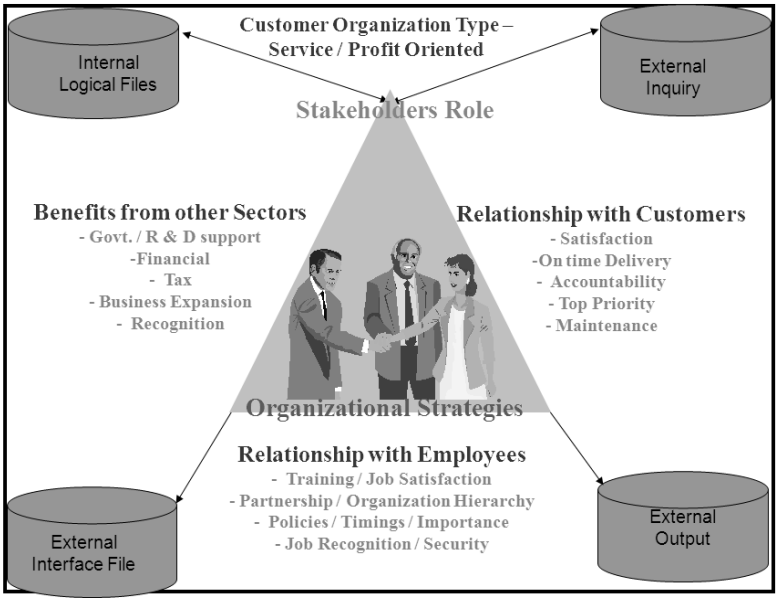

Figure 4. Organizational Strategies Role in Estimation

IV. Methodology For Estimation With Machine Learning

This article describes two methods of machine learning, which we use to build estimators of software development effort from historical data with Neuro-Fuzzy Expert System. In order to optimize the results data analytical techniques have been used [63-71]. First we need to find the size of the product then it is easy to find the required effort. Also capabilities of the team members in terms of project complexity and risk factors to be measured. After that we can focus on estimating the total cost of the product and as a result we can schedule the product [2, 4-7].

The process encompasses certain key ingredients that include

- ➢ Developing a good project execution plan

- Converting the goal into a well-structured requirement specification
- Dividing the project into well-defined phases
- Allocating the right resources to the right job
- Working at the cost of project execution
- Delivering the project within the overall timeframe

Formal estimation models not tailored to a particular organization's own context, may be very inaccurate. Use of own historical data is consequently crucial if one cannot be sure that the estimation model's core relationships are based on similar project contexts [72-88].

Formulating the problem and fitting into Machine learning. Domain knowledge is more required in identifying the features in the form of requirements that can represent the data set. All the problems may not be easily expressed. Without proper understanding of distribution of the data, formulation of the problem, preprocessing, assumptions and presumptions of machine learning algorithms implementation is not that easy task. [89-101]

- ✓ In order to solve the estimation problem the following key points to be addressed:
- ✓ Unavailability of data in a suitable format
- ✓ Incomplete, noisy, and inconsistent data
- ✓ Assessing the expected error of a learning algorithm on a problem
- ✓ Data normalization and standardization
- ✓ Relationships and correlations can be hidden within large amounts of data
- ✓ Human expertise does not exist for all kinds of problems
- ✓ Human designers often produce machines that do not work as desired in the environments in which they are not used.
- ✓ The amount of knowledge available about certain tasks might be too large for explicit encoding by humans
- ✓ Environments change over time.
- ✓ New knowledge about tasks is constantly being discovered by humans. It may be difficult to continuously re-design systems "by hand".
- ✓ What algorithms are available for learning a concept? How well do they perform?
- ✓ How much training data is sufficient to learn a concept with high confidence?
- ✓ When is it useful to use prior knowledge?
- ✓ What are best tasks for a system to learn?
- ✓ What is the best way for a system to represent its knowledge?
- ✓ How can we optimize the accuracy on future data points?
- ✓ How can we formulate application problems as machine learning paradigms?

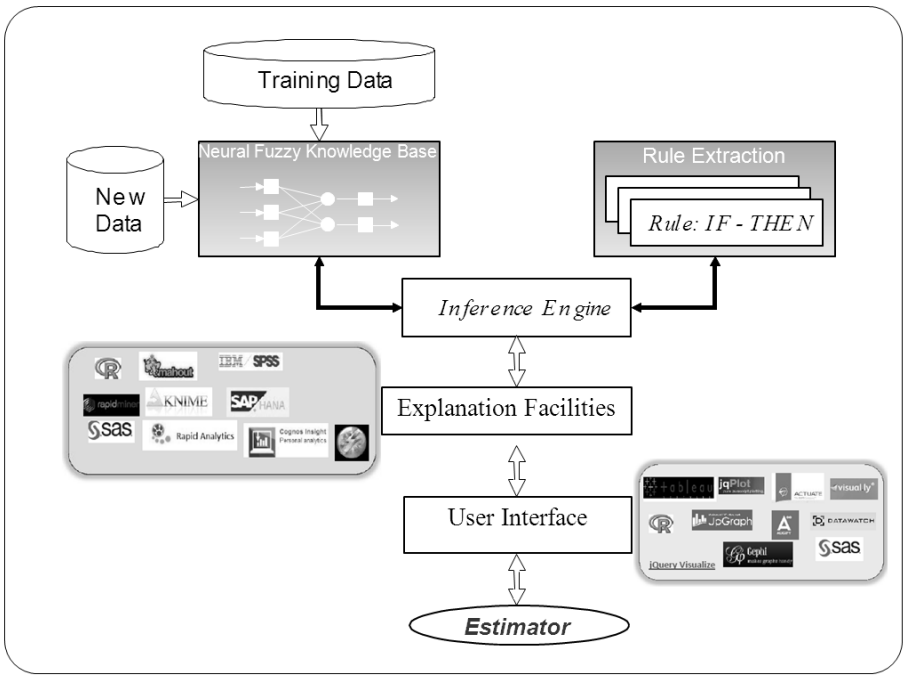

Figure 5. Model of Neuro-Fuzzy Expert Estimation System with Data Analytics tools

In Software Industry size estimation is a precursor to effort, schedule and cost estimation, the accuracy of these estimates depend on the accuracy of the size estimates.

- **Unfamiliar project**
  For the rest of this report, the definition of unfamiliar projects, or teams, means they either have little knowledge or little experience with the type of project being developed.
  This includes the following:
  Inexperienced in general
  Experienced, but in a new domain
  Experienced, but using new technologies
- **Continuous assessment**
  Continuous assessment is a type of assessment methodology that takes place over a period of time. In the software development context, the process is done parallel to the development, instead of being done at major milestones or reviews.
- **Traditional development project**
  A type of project in which, the product must be developed from scratch.
  The development team must write the majority of the source code to implement the end user functionalities.
  A type of project that aims at integrating and/or tailoring either one, or a set of, non-developmental items or commercial off-the-shelf products.

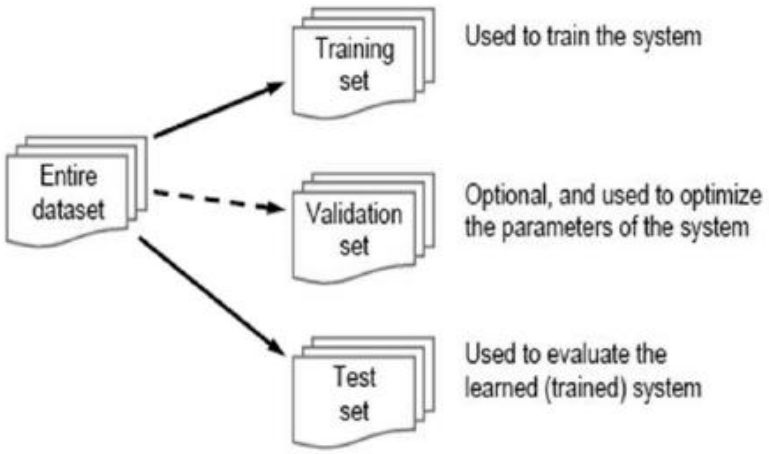

Figure 6. Optimized Estimation Method

To use this method, one needs to be aware of the techniques but need not be an expert. What is needed is the knowledge of the software project or which they are making an estimate. This method facilitates making a new estimate thru copying an existing one and modifying it. This also facilitates making a new estimate from an existing estimate - thus improving estimation productivity greatly. Thus is a time and effort saving mechanism resulting in decreasing pressure on the valuable time of Project Managers / Leaders and leads to better productivity in general. It is important to point out that though producing high quality software requires more effort; this additional effort is more than recovered by lower maintenance effort and trouble [101].

The estimation process, model and technique that we decide to use should take into account all the below parameters appropriately:

- ✓ Availability and stability of the developer environment
- ✓ Team capability, skill and experience
- ✓ Team stability / manpower turnover
- ✓ Maturity of the processes
- ✓ Reusable software available to the project
- ✓ Reusable software to be build by the project
- ✓ Extent of communication possible with the user / customers
- ✓ Extent of automated tools used for software development and maintenance
- ✓ Extent and degree of detail of the required user documentation
- ✓ Cohesion of stakeholders and teams

.

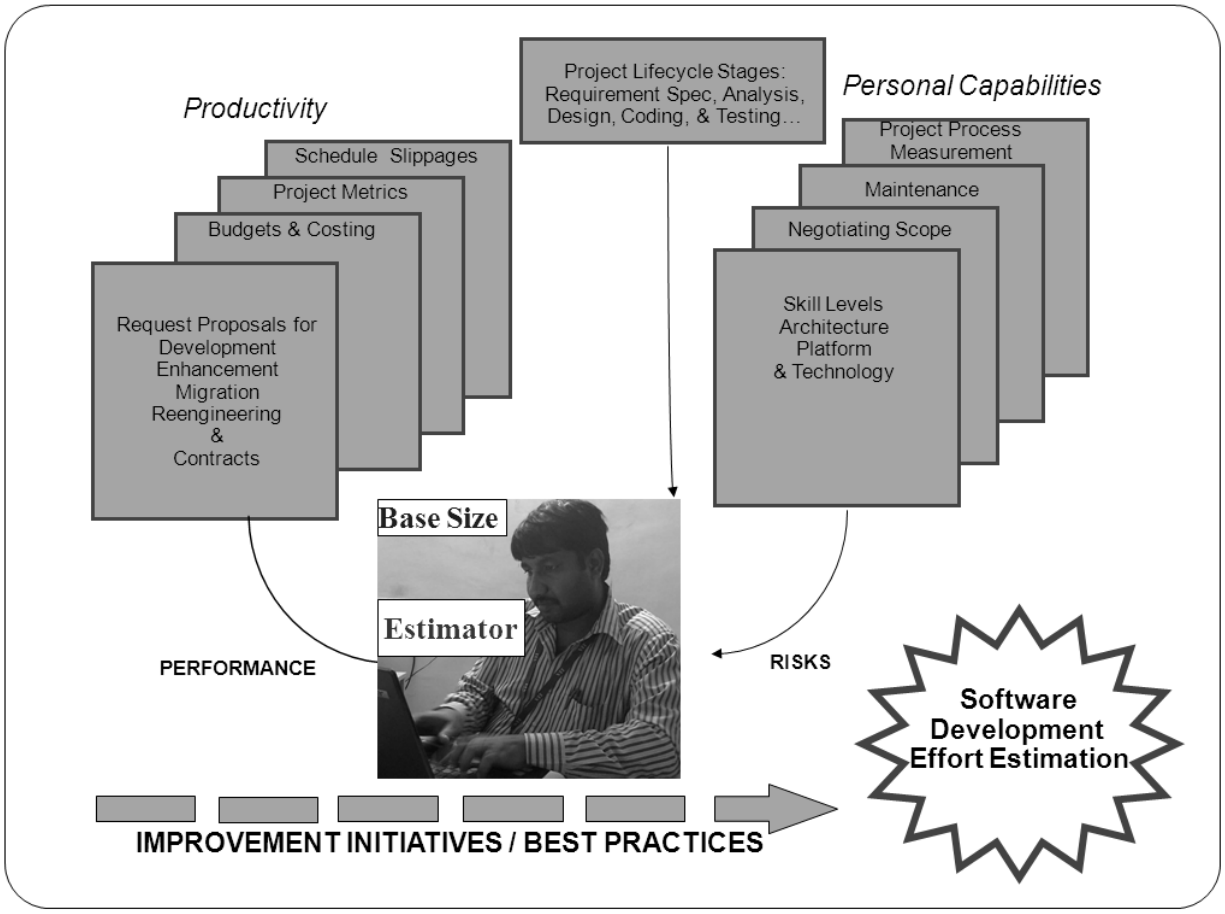

Figure7. The Role of Estimator in Software Industry

V. Expected Outcome With Data Analytics

Sometimes there is a dominant factor that influences the decision-making. For example – for a mining company there is no alternative but to open it near the mine. A maritime ship liner needs to be near the seacoast. Location of market is another dominant factor. Other cases may include emotional factors of the entrepreneur – like his native place when it comes to opening his company or the expertise of the entrepreneur when it comes to selecting the product and so on. In day-to-day affairs, customer preference becomes a dominant factor, around which we have to manage. In some cases like Y2K, the time becomes the dominant factor. In some cases, the statutory obligations become the dominant factor. When a dominant factor is present in a decision scenario – the decision is made for us. The software size is the most important factor that affects the software cost. Here the system will consider data analytical techniques to analyze the appropriate decision. The lines of code and function point are the most popular software size metrics used in practice. The cost to fix a defect rises dramatically as the time from when it's introduced. These remains true whether the project is highly sequential or highly iterative. The main reason is here doing around five to ten percent of requirements gathering and design based on that [2, 4, 5].

With high objective the organizations may establish the service level agreements for resolving issues. The targets pertain to factors that will measure the effectiveness of the system [6].

Analytical Decision Making – this style implies that a thorough analysis is carried out in which all possible alternatives are considered along with their costs and possible results are analyzed and the optimal decision is selected. This is used by knowledgeable people and somewhat less experienced in their field. The scenarios that come to mind where this style is appropriate are –

a) Strategic decisions which have long term impact – especially selection and design decisions
b) There is time available for making the decision

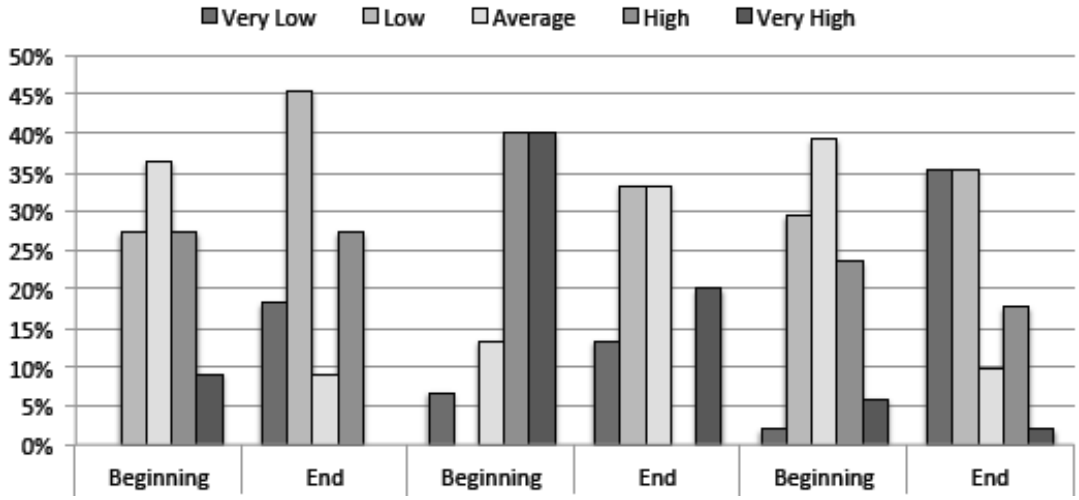

Figure 8. Organizational Strategies on different project levels

Standard method for measuring software development from the customer's point of view is always different. Quantifies functionality provided to the user based primarily on logical design. Measure software development and maintenance independently of technology used for implementation [6].

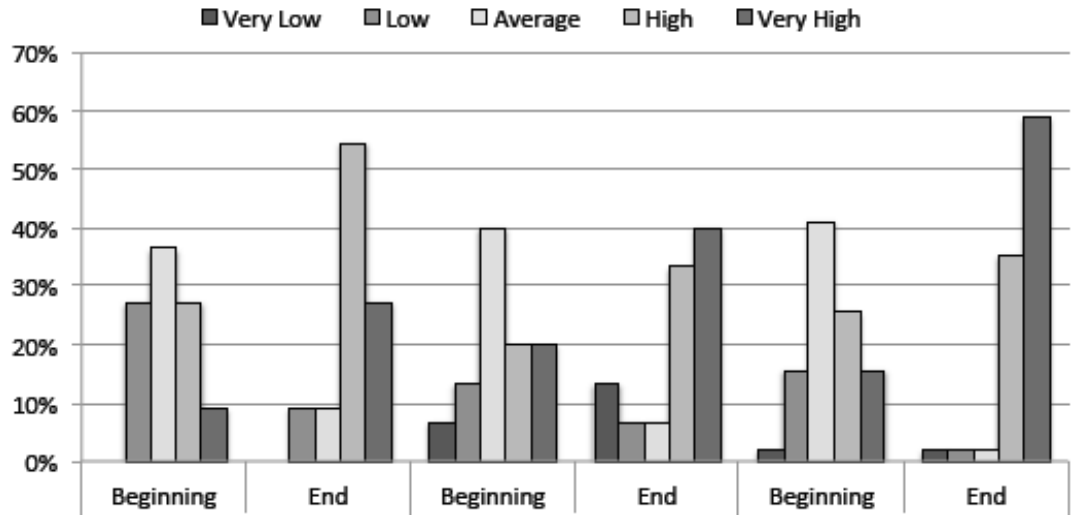

Figure 9. Social Interaction among team members in different project

VI. Conclusion

In this paper the author's approach is not to criticize the existing popular estimation models. Most of the models are useful. Different models cater different needs. Existing models focused either on size or line of code. Actual required effort estimated by the existing models. Besides, organizational strategies and social interaction among the group members are the key attributes in software estimation. The accurate estimation immediately affects the success of project. Data analytical techniques will help the estimators to track the project status and take a correct measure in order to improve the quality of product. With the help of Machine Learning and Big Data Analytics tools & techniques optimization is possible. Further research will be continued with sufficient amount of industrial data with other parameters.

References

[1] G. R. Finnie, G.E. Wittig and J-M. Desharnais, "A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models", J. Systems Software, Elsevier Science Inc, 1997; 39: 281-289.

[2] Hahn-Ming Lee and Bing-Hui Lu, "FUZZY BP: A Neural Network Model with Fuzzy Inference", IEEE, 0-7803-1901-X/94, Pp. 1583-1588.

[3] Girish H. Subramanian and Steven Breslawski, "An Empirical Analysis of Software Effort Estimate Alterations", Elsevier Science Inc, J. SYSTEMS SOFTWARE 1995; 31:135-141.

[4] Albert L. Lederer and Jayesh Prasad, "A Causal Model for Software Cost Estimating Error", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 24, NO. 2, FEBRUARY 1998; Pp. 137-148.

[5] Hu, Qing;Plant, Robert T;Hertz, David B, "Software cost estimation using economic production models", Journal of Management Information Systems; 1998; 15, 1; Pp. 143-163.

[6] Randy K. Smith, Joanne E. Hale, and Allen S. Parrish, "An Empirical Study Using Task Assignment Patterns to Improve the Accuracy of Software Effort Estimation", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 27, NO. 3, 2001; Pp. 264-271.

[7] Colin J. Burgess and Martin Lefley, Can genetic programming improve software effort estimation? A comparative evaluation, Information and Software Technology 43 (2001); Elsevier, Pp. 863-973.

[8] Kevin Strike, Khaled El Emam, and Nazim Madhavji, "Software Cost Estimation with Incomplete Data", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 27, NO. 10, 2001; Pp. 890-908.

[9] Ali Idri, Taghi M. Khoshgoftaar, and Alain Abran, "Can Neural Networks be easily Interpreted in software Cost Estimation?", World Congress on Computational Intelligence, Honolulu, Hawaii, May 12-17, 2002; Pp.1-7.

[10] Maurizio Morisio, Michel Ezran, and Colin Tully, "Success and Failure Factors in Software Reuse", IEEE Transactions on Software Engineering, Vol. 28, No. 4, April 2002; Pp. 340-357.

[11] Tim Menzies and Justin S. Di Stefano, "More Success and Failure Factors in Software Reuse", IEEE Transactions on Software Engineering, Vol. 29, No. 5, May 2003; Pp. 474-477.

[12] Maurizio Morisio, Michel Ezran, and Colin Tully, Comments on "More Success and Failure Factors in Software Reuse", IEEE Transactions on Software Engineering, Vol. 29, No. 5, May 2003; Pp. 478.

[13] Hamdi A. Bashir and Vince Thomson, "Estimating design effort for GE hydro projects", Computers & Industrial Engineering, Elsevier, 46 (2004), 195-204.

[14] Magne Jorgensen and Kjetil Molokken-Ostvold, "Reasons for Software Effort Estimation Error: Impact of Respondent Role, Information Collection Approach, and Data Analysis Method", IEEE Transactions on Software Engineering, Vol. 30, No. 12, December 2004; Pp. 993-1007.

[15] Cuauhtémoc López Martín, Jérôme Leboeuf Pasquier, Cornelio Yáñez M., and Agustín Gutiérrez T., "Software Development Effort Estimation Using Fuzzy Logic: A Case Study", Proceedings of the Sixth Mexican International Conference on Computer Science (ENC'05), IEEE Computer Society, 0-7695-2454-0/05, 2005.

[16] Stein Grimstad, Magne Jorgensen and Kjetil Molokken-Ostvold, "Software effort estimation terminology: The tower of Babel", Information and Software Technology, Elsevier, 48 (2006); Pp. 302–310.

[17] Chao-Jung Hsu and Chin-Yu Huang, "Comparison and Assessment of Improved Grey Relation Analysis for Software Development Effort Estimation", IEEE, 1-4244-0148-8/06, 2006; Pp. 663-667.

[18] Adriano L.I. Oliveira, "Estimation of software project effort with support vector regression", Neurocomputing, Elsevier, 69, 2006; Pp. 1749-1753.

[19] Alaa F.Sheta, "Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects", Journal of Computer Science ISSN 1549-3636, 2(2): 118-123, 2006 Science Publications.

[20] Sun-Jen Huang, and Nan-Hsing Chiu, "Optimization of analogy weights by genetic algorithm for software effort estimation", Information and Software Technology, Elsevier, Science Direct, 48, 2006; Pp. 1034-1045.

[21] Ayman Issa, Mohammed Odeh, and David Coward, "Software Cost Estimation Using Use-Case Models: a Critical Evaluation", IEEE, 0-7803-9521-2/06, 2006; Pp. 2766-2771.

[22] Nan-Hsing Chiu, and Sun-Jen Huang, "The adjusted analogy-based software effort estimation based on similarity distances", The Journal of Systems and Software, Elsevier, 80, 2007; Pp. 628-640.

[23] Simon I. K. Wu, "The quality of design team factors on software effort estimation", IEEE 1-4244-0318-9/06, 2006; Pp. 6-11.

[24] Magne Jorgenesen and Martin Sheppered, "A Systematic Review of Software Development Cost Estimation Studies", IEEE Transactions on Software Engineering, Vol. 33, No. 1, January 2007; Pp.33-53.

[25] Chao-Jung Hsu and Chin-Yu Huang, "Improving Effort Estimation Accuracy by Weighted Grey Relational Analysis During Software Development", 14th Asia-Pacific Software Engineering Conference, IEEE Computer Society, 1530-1362/07, 2007; Pp. 534-541.

[26] Kristian Marius Furulund and Kjetil Molokken-Ostvold, "Increasing Software Effort Estimation Accuracy – Using Experience Data, Estimation Models and Checklists", IEEE Computer Society, Seventh International Conference on Quality Software (QSIC 2007), 0-7695-3035-4/07.

[27] Petronio L. Braga, Adriano L.I. Oliveira, and Silvio R.L. Meira, "Software Effort Estimation using Machine Learning Techniues with Robust Confidence Intervals", 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI.2007.172), 1082-3409/07, Pp.181-185.

[28] Stefan Koch and Johann Mitlohner, "Software project effort estimation with voting rules", Decision Support Systems, Elsevier, 46 (2009); Pp. 895-901.

[29] Nan-Hsing Chiu, Sun-Jen Huag, "The adjusted analogy-based software effort estimation based on similarity distances", The Journal of Systems and Software, Elsevier, 80 (2007); Pp.628-640.

[30] Adil M. Salam, Nadia F. Bhuiyan, Gerard J. Gouw, and Syed Asif Raza, "A case study to estimate design effort for Pratt & Whitney Canada", International Journal of Management, Vol. 3 (2008) No. 4, Pp. 284-293.

[31] Sun-Jen Huang, Nan-Hsing Chiu, and Yu-Jen Liu, "A comparative evaluation on the accuracies of software effort estimates from clustered data", Elsevier, Science Direct, Information and Software Technology 50 (2008); Pp. 879-888.

[32] Wei Xia, Luiz Fernando Capretz, Danny Ho, and Faheem Ahmed, "A new calibration for Function Point complexity weights", Elsevier, Science Direct, Information and Software Technology, 50 (2008); Pp. 670-683.

[33] Parag C. Pendharkar, James A. Rodger, and Girish H. Subramanian, "An empirical study of the Cobb-Douglas production function properties of software development effort", Elsevier, Science Direct, Information and Software Technology, 50 (2008); Pp. 1181-1188.

[34] Jacky Keung and Ross Jeffery, "Automated Support for Software Cost Estimation using Web-CoBRA", IEEE Computer Society, APSEC.2008.44, 1530-1362/08; Pp. 519-526.

[35] CHEN Qinghang, Fang Shuojin, and Wang Wenfu, "Development of the Decision Support System for Software Project Cost Estimation", IEEE Computer Society, World Congress on Software Engineering, WCSE.2009.269, 978-0-7695-3570-8/09; Pp. 297-300.

[36] LI Jun, LIN Jianming, JIN Yongqin, and CHEN Qinghang, "Development of the Decision Support System for Software Project Cost Estimation", 2008 International Symposium on Information Science and Engineering, IEEE Computer Society, ISISE.2008.270, 978-0-7695-3494-7/08; Pp. 299-302.

[37] M. Pikkarainen, J. Haikara, O. Salo, P. Abrahamsson, and J. Still, "The impact of agile practices on communication in software development", Springer, Empir Software Eng. 10664-008-9065-9, (2008) 13: 303-337.

[38] Qin Liu, Wen Zhong Qin, Robert Mintram, and Margaret Ross, "Evaluation of preliminary data analysis framework in software cost estimation based on ISBSG R9 Data", Springer, Software Qual J11219-007-9041-4, (2008) 16:411-458.

[39] Sun-Jen Huang, Nan-Hsing Chiu, and Li-Wei Chen, "Integration of the grey relational analysis with genetic algorithm for software ", Computing, Artificial Intelligence and Information Management, Elsevier, European Journal of Operational Research 188 (2008); Pp. 898-909.

[40] Javier Aroba, Juan J. Cuadrado-Gallego, Miguel-Angel Sicilia, Isabel Ramos, and Elena Garcia-Barriocanal, "Segmented software cost estimation models based on fuzzy clustering", Elsevier, Science Direct, The Journal of Systems and Software, 81 (2008); Pp. 1944-1950.

[41] K. Vinay Kumar, V. Ravi, Mahil Carr, and N. Raj Kiran, "Software development cost estimation using wavelet neural networks", Elsevier, Science Direct, The Journal of Systems and Software, 81 (2008); Pp. 1853-1867.

[42] Bing Zhang and Rubo Zhang, "Evaluation Model of Software Cost Estimation Methods Based on Fuzzy-Grey Theory", 2009 Fourth International Conference on Internet Computing for Science and Engineering, ICICSE.2009.63, IEEE Computer Society, 978-0-7695-4027-6/10; Pp. 52-55.

[43] Jianfeng Wen, Shixian Li, and Linyan Tang, "Improve Analogy-Based Software Effort Estimation using Principal Components Analysis and Correlation Weighting", 2009 16th Asia-Pacific Software Engineering Conference, APSEC.2009.40, IEEE Computer Society, 1530-1362/09; Pp. 179-186.

[44] Jacky Keung, "Software Development Cost Estimation using Analogy: A Review", 2009 Australian Software Engineering Conference, ASWEC.2009.32, IEEE Computer Society, 1530-0803/09; 327-336.

[45] Ch. Satyananda Reddy and KVSVN Raju, "An Optimal Neural Network Model for Software Effort Estimation", Int. J. of Software Engineering, IJSE, Vol. 3, No. 1, January 2010, Pp. 63-78.

[46] Iman Attarzadeh, and Siew Hock Ow, "A Novel Soft Computing Model to Increase the Accuracy of Software Development Cost Estimation", IEEE, Vol.3, 978-1-4244-5586-7/10, 2010; Pp. 603-607.

[47] Chao-Jung Hsu, Nancy Urbina Rodas, Chin-Yu Huang, and Kuan-Li Peng, "A Study of Improving the Accuracy of Software Effort Estimation Using Linearly Weighted Combinations", 2010 34th Annual IEEE Computer Software and Applications Conference Workshops, COMPSACW.2010.27, IEEE Computer Society, 978-0-7695-4105-1/10; Pp. 98-103.

[48] Heejun Park and Seung Baek, "An empirical validation of a neural network model for software effort estimation", Elsevier, Science Direct, Expert Systems with Applications, 35 (2008); Pp. 929-937.

[49] Saleem Basha and Dhavachelvan P, "Analysis of Empirical software Effort Estimation Models", International Journal of Computer Science and Information Security (IJCSIS), Vol. 7, No. 3, 2010; Pp. 68-77.

[50] S.M.Nadgeri, Vidya P. Hulsure, and A.D. Gawande, "Comparative study of various regression methods for software effort", Third International Conference on Emerging Trends in Engineering and Technology, ICETER.2010.22, IEEE Computer Society, 978-0-7695-4246-1/10, 2010; Pp. 642-645.

[51] Ali Idri, Abdelali Zakrani and Azeddine Zahi, "Design of Radial Basis Function Neural Networks for Software Effort Estimation", International Journal of Computer Science Issues (IJCSI), Vol. 7, Issue 4, No. 3, July 2010; Pp. 11-17.

[52] Tuan Khanh Le-Do, Kyung-a Yoon, Yeong-Seok Seo, and Doo-Hwan Bae, "Filtering of Inconsistent Software Project Data for Analogy-based Effort Estimation", 2010 34th Annual IEEE Computer Software and Applications Conference, COMPSAC.2010.56, IEEE Computer Society, 0730-3157/10; Pp. 503-508.

[53] Adriano L.I. Oliveira, Petronio L. Braga, Ricardo M.F. Lima, and Marcop L. Cornelio, "GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation", Elsevier, Science Direct, Information and Software Technology, 52 (2010); Pp. 1155-1166.

[54] Ch.V.M.K.Hari, Prasad Reddy P.V.G.D, M.Jagadeesh, and G. Sri Ram Ganesh, "Interval Type-2 Fuzzy Logic for Software Cost Estimation Using TSFC with Mean and Standard Deviation", 2010 International Conference on Advances in Recent Technologies in Communication and Computing, ARTCom.2010.40, IEEE Computer Society, 978-0-7695-4201-0/10; Pp. 40-44.

[55] F.Ferrucci, C. Gravino, R. Oliveto, F.Sarro and E.Mendes, "Investigating Tabu Search for Web Effort Estimation", 2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications, SEAA.2010.59, IEEE Computer Society, 978-0-7695-4170-9/10; Pp. 350-357.

[56] Iman Attarzadeh and Siew Hock Ow, "Proposing a New Software Cost Estimation Model Based on Artificial Neural Networks", IEEE 2010 2nd International Conference on Computer Engineering and Technology, Vol. 3, 978-1-4244-6349-7/10; Pp. 487-491.

[57] M. Ochodek, J. Nawrocki, and K. Kwarciak, "Simplifying effort estimation based on Use Case Points", Elsevier, Science Direct, Information and Software Technology, 53 (2011); Pp. 200-213.

[58] Zheng Li and Jacky Keung, "Software Cost Estimation Framework for Service-Oriented Architecture Systems using Divide-and-Conquer Approach", 2010 Fifth IEEE International Symposium on Service Oriented System Engineering, SOSE.2010.29, IEEE Computer Society, 978-0-7695-4081-8/10; Pp. 47-54.

[59] Vu Nguyen, Barry Boehm, and Phongphan Danphitsanuphan, "A controlled experiment in assessing and estimating software maintenance tasks", Elsevier, Science Direct, Information and Software Technology, 53 (2011); Pp. 682-691.

[60] J.N.V.R.Swarup Kumar, T.Govinda Rao, Y.Naga Babu, S.Chaitanya and K.Subrahmanyam, " A Novel Method for Software Effort Estimation Using Inverse Regression as firing Interval in fuzzy logic", IEEE, 978-1-4244-8679-3/11, 2011; Pp. 177-182.

[61] Ricardo de A. Araujo, Adriano L.I. Oliveira, and Sergio Soares, "A shift-invariant morphological system for software development cost estimation", Elsevier, Science Direct, Expert Systems with Applications 38 (2011); Pp. 4162-4168.

[62] Jing Liu, Samik Basu, and Robyn R. Lutz, "Compositional model checking of software product lines using variation point obligations", Springer, Autom Softw Eng (2011) 18: 39-76.

[63] T.Wijayasiriwardhane, R. Lai, and K.C. Kang, "Effort estimation of component-based software development – a survey", The Institution of Engineering and Technology 2011, Vol. 5, Issue. 2; Pp. 216-228.

[64] Ali Idri and Sanaa Elyassami, "Applying Fuzzy ID3 Decision Tree for Software Effort Estimation", International Journal of Computer Science Issues (IJCSI), Vol. 8, Issue. 4, No. 1, July 2011; Pp. 131-138.

[65] Joao Carlos Cunha, Sergio Cruz, Marco Costa, Ana Rita Rodrigues, and Marco Vieira, "Implementing Software Effort Estimation in a Medium-sized Company", 2011 34th IEEE Software Engineering Workshop,SEW.2011.19, IEEE Computer Society, 1550-6215/11; Pp. 92-96.

[66] Vachik S. Dave and Kamlesh Dutta, "Neural Network based Software Effort Estimation & Evaluation criterion MMRE", International Conference on Computer & Communication Technology (ICCCT – 2011), IEEE, 978-1-4577-1386-611; Pp. 347-351.

[67] Qinbao Song and Martin Shepperd, "Predicting Software project effort: A grey relational analysis based method", Elsevier, Science Direct, Expert System with Applications 38 (2011); Pp. 7302-7316.

[68] Rolan Abdukalykov, Ishrar Hussain, Mohamad Kassab, and Olga Ormandjieva, "Quantifying the Impact of Different Non-Functional Requirements and Problem Domains on Software Effort Estimation", 2011 Ninth International Conference on Software Engineering Research, Management and Applications, SERA.2011.45, IEEE Computer Society, 978-0-7695-4490-8/11; Pp. 158-165.

[69] Asiegbu B C and Ahaiwe J, "Software Cost Drivers and Cost Estimation in Nigeria", Interdisciplinary Journal of Contemporary Research in Business, Vol.3, No.8, December 2011; Pp.431-451.

[70] Z. Zia, A. Rashid, and K. uz Zaman, "Software Cost Estimation for component based fourth-generation-language software applications", The Institution of Engineering and Technology, 2011, Vol. 5, Issue. 1, Pp. 103-110.

[71] Srimam Srichandan, "A new approach of Software Effort Estimation Using Radial Basis Function Neural Networks", International Journal on Advanced Computer Theory and Engineering (IJACTE), Vol.1 Issue 1, 2012; Pp.113-120.

[72] Karel Dejaeger, Wouter Verbeke, David Martens, and Bart Baesens, "Data Mining Techniques for Software Effort Estimation: A Comparative Study", IEEE Transactions on Software Engineering, Vol.38, No.2, March/April 2012; Pp.375-397.

[73] V. Khatibi Bardsiri, D.N.A. Jawawi, S.Z.M. Hashim, and E. Khatibi, "Increasing the accuracy of software development effort estimation using projects clustering", The Institution of Engineering and Technology 2012, Vol. 6, Iss. 6, Pp.461-473.

[74] A. Bonetti, S. Bortot, M. Fedrizzi, R.A. Marques Pereira, and A. Molinari, "Modeling group processes and effort estimation in project management using the Choquet integral: An MCDM approach", Elsevier, SciVerse ScieneDirect, Expert Systems with Applications, 39, 2012; Pp. 13366-13375.

[75] Syed Ali Abbas, Xiao Feng Liao, Afshan Azam, and Ayesha Kulsoom Khattak, "Neural Net Back Propagation and Software Effort Estimation: A comparison based perspective", ARPN Journal of Systems and Software, Vol.2, No.6, June 2012; Pp.195-207.

[76] Kristin Borte, Sten R. Ludvigsen, and Anders I. Morch, "The role of social interaction in software effort estimation: Unpacking the "magic step" between reasoning and decision-making", Elsevier, SciVerse Science Direct, Information and Software Technology, 54, 2012; Pp. 985-996.

[77] Ali Bou Nassif, Danny Ho, and Luiz Fernando Capretz, "Towards an early software estimation using log-linear regression and multilayer perceptron model", Elsevier, SciVerse ScienceDirect, The Journal of Systems and Software, 86, 2013; Pp. 144-160.

[78] Ye Yang, Zhimin He, Ke Mao, Qi Li, Vu Nguyen, Barry Boehm, and Ricardo Valerdi, "Analyzing and handling local bias for calibrating parametric cost estimation models", Elsevier, SciVerse ScienceDirect, Information and Software Technology, 55, 2013; Pp.1496-1511.

[79] Leandro L. Minku and Xin Yao, "Ensembles and locality: Insight on improving software effort estimation", Elsevier, SciVerse ScienceDirect, Information and Software Technology, 55, 2013; Pp. 1512-1528.

[80] Moataz A. Ahmed, Irfan Ahmad, and Jarallah S. AlGhamdi, "Probabilistic size proxy for software effort prediction: A framework", Elsevier, SciVerse ScienceDirect, Information and Software Technology, 55, 2013; Pp. 241-251.

[81] Ekrem Kocaguneli and Tim Menzies, "Software effort models should be assessed via leave-one-out validation", Elsevier, SciVerse ScienceDirect, The Journal of Systems and Software, 86, 2013; 1879-1890.

[82] Moataz A. Ahmed, Irfan Ahmad, and Jarallah S. AlGhamdi, "Probabilistic size proxy for software effort prediction: A framework", Elsevier, SciVerse ScienceDirect, Information and Software Technology, 55, 2013; Pp. 241-251.

[83] V. Khatibi Bardsiri, D.N.A. Jawawi, S.Z.M. Hashim, and E. Khatibi, "Increasing the accuracy of software development effort estimation using projects clustering", The Institution of Engineering and Technology 2012, Vol. 6, Iss. 6, Pp.461-473.

[84] Magne Jorgensen, Simula Research Laboratory, Practical Guidelines for Expert-Judgment-Based Software Effort Estimation, May/June 2005, IEEE SOFTWARE, Pp.57-63.

[85] M. Jorgensen, "A Review of Studies on Expert Estimation of Software Development Effort," J. Systems and Software, vol. 70, nos. 1–2, 2004, pp. 37–60.

[86] M. Jorgensen and D.I.K. Sjoberg, "An Effort Prediction Interval Approach Based on the Empirical Distribution of Previous Estimation Accuracy," J. Information and Software Technology, vol. 45, no. 3, 2003, pp. 123–136.

[87] B V A N S S Prabhakar Rao & P Seetha Ramaiah, " An efficient framework for practical Software Estimation with Machine Learning and Big Data Analytics, IJDI-ERET, Vol. 2, No.2, July-December 2013, Pp. 50-55.

[88] Krishnamoorthy Srinivasan and Douglas Fisher, "Machine Learning Approaches to Estimating Software Development Effort", IEEE Transactions on Software Engineering, Vol.21, No.2, February 1995, Pp. 126-137.

[89] S. McConnell, Code Complete: A Practical Handbook of Software Construction, Microsoft Press, second ed., 2004.

[90] M.A. Parthasarathy, Practical Software Estimation – Function Point Methods for Insourced and Outsourced Projects, Infosys Press, Pearson, First Impression, 2007.

[91] Gopalaswamy Ramesh and Ramesh Bhattiprolu, Software Maintenance - Effective Practices for Geographically Distributed Environments, TMH, 2009.

[92] Roger S Pressman, Seventh Edition, Software Engineering, A Practitioner.s Approach; McGraw Hill International Edition.

[93] Thimothy C. Lethbridge and Robert Laganiere, Object-Oriented Software Engineering Practical software development using UML and Java, Tata McGraw-Hill Publish Company Limited, 2008.

[94] Rajib Mall, Fundamentals of Software Engineering, Third Edition, PHI Learning Private Limited, 2011.

[95] S. Rajasekaran, G.A. Vijayalakshmi Pai, Neural Networks, Fuzzy Logic, and Genetic Algorithms, Synthesis and Applications, PHI Learning Private Limited, 2009.

[96] Yegnanarayana, Artificial Neural Netwoks, PHI Learning Private Limited, 2010.

[97] B V A N S S Prabhakar Rao *& P Sita Ramaiah;* Adaptive System for Estimating Development Effort, Journal of CNS; ISSN 2277-1735, Vol 1, Issue 1, Jan, 2012, pp 52-56.

[98] G. R. Finnie and G.E. Wittig and J-M. Desharnais, "A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models", J. Systems Software, Elsevier Science Inc, 1997; 39: 281-289.

[99] B V A N S S Prabhakar Rao and P Seetha Ramaiah, Software Size Estimation Using Fuzzy Backpropagation Network Method; International Journal of Computer Science Issues, Vol. 9, Issue 1, No 1, January 2012, ISSN : 1694-0814, pp. 339-348.

[100] M. Morisio and M. Ezran, and C. Tully, Success and failure factors in software reuse, IEEE Transactions on Software Engineering, volume 28,4, 2002, pages 340--357

[101] B V A N S S Prabhakar Rao and P Seetha Ramaiah, A Novel Approach to Design Neuro-Fuzzy Expert System for Software Estimation; International Journal of Engineering Research & Technology, Vol. 2, Issue 12, December 2013, ISSN: 2278-0181, pp.3012-3017.