Indonesian Journal of Electrical Engineering and Computer Science
Vol. 7, No. 3, September 2017, pp. 867 ~ 876
DOI: 10.11591/ijeecs.v7.i3.pp867-876

A Survey on Software Estimation Techniques in Traditional and Agile Development Models

B. Prakash*, V. Viswanathan
School of Computing Science and Engineering, VIT University, Chennai, India
*Corresponding author, e-mail: priyam.prakash2013@vit.ac.in

Abstract

Software projects mostly exceeds budget, delivered late and does not meet with the customer's satisfaction for years. In the past, many traditional development models like waterfall, spiral, iterative, and prototyping methods are used to build the software systems. In recent years, agile models are widely used in developing the software products. The major reasons are – simplicity, incorporating the requirement changes at any time, light-weight approach and delivering the working product early and in short duration. Whatever the development model used, it still remains a challenge for software engineer's to accurately estimate the size, effort and the time required for developing the software system. This survey focuses on the existing estimation models used in traditional as well in agile software development.

Keywords*: Estimation, Project Management, Software Engineering, Agile software development*

1. Introduction

There are number of software development models used for constructing software applications, each has formed based on certain objectives. At higher level, the development models are classified into two categories namely traditional software development and Agile (Dynamic) software development.

Traditional process is a step by step approach in which the development phases like requirements, design, construction and testing are performed one after another. First step in traditional approach is Requirement Elicitation and Analysis. Requirement Elicitation [1] is the process of gathering a complete set of requirements from the stakeholder. There are various techniques used by the development team for gathering the requirements such as Interviews, Story Boarding, Workshops and Brainstorming. The gathered requirements are analyzed by the development team, project manager and the clients and the agreed set of requirements are documented in the form of Software Requirement Specification (SRS). Further changes in the requirements are strictly not entertained once the specification is approved by the stakeholders. The developed SRS document act as an input for further phases-design, development and testing. Developers perform detailed high level and low level design based on the documented SRS followed by the coding phase. Once the product has been developed, different types of testing are performed on the developed product by the testers and business users and the software will be deployed at the customer environment and then the maintenance phase begins.

The most widely used traditional development models used for building software systems are Waterfall, V model, Spiral, Prototyping and Iterative models. All the above said development phases are performed in sequence irrespective of the development model chosen. In late 90s, agile processes are slowly started replacing the traditional approach and in recent years most clients only prefer agile model for their software development. Agile is a lightweight approach which adopts iterative software development methodology. According to Agile manifesto [2]-[3], following are the major strength about agile process over traditional approach:

a. Lightweight vs. Heavyweight
b. Customer collaboration over contract negotiation
c. Welcoming requirement changes even in later phase compared to following a rigid Change Management process;
d. Believes in working software rather than detailed documentation
e. Focus will be on people rather than artifacts

Received May 29, 2017; Revised July 15, 2017; Accepted August 23, 2017

In Agile, further there are number of dynamic models available for constructing a software product that includes Extreme Programming (XP), Dynamic systems development method (DSDM), Agile Modeling (AM), SCRUM, Crystals and Lean development. Each method has its own unique attributes and qualities, so choosing the right method is a critical task and it should be based on the project requirements. In recent years, software industries are showing interests in adopting the hybrid process models like combining the strengths of waterfall and agile or by combining the agile methods like SCRUM, XP and RUP together [4].

1.1. Project Planning and Estimation

As software development process has become integral part of many organizations, it is necessary to have an accurate estimation model for effectively manage, predict and plan the software development process. Inaccurate estimation leads to poor project planning and scheduling, under or over staffing, having too short releases which in turn impact the quality of the product. Software Estimation involves estimating the size, effort and cost required for the product to be developed. According to Standish group survey [5], only 16.2% of the projects were completed on time and within the estimated budget. Overall 52.7% of the projects were completed but exceed the budget and development duration. The study reveals further that the major reasons for the project failures are due to lack of user involvement, incomplete requirements, inefficient planning and estimation.

Software process model plays a major role in software estimation process. The estimation technique will vary based on the development model used. For example, estimation technique that suits for waterfall or spiral models may not be compatible for agile approach and vice versa. Therefore, it is essential to identify the suitable estimation technique based on the software development model used.

Even though there are various literature studies exists in the field of software estimation, they all are either related to domain specific or focused primarily on one particular project attributes such as cost, effort or size and not all. The main focus on this survey is to compare various estimation techniques used in traditional and agile based projects and also focused on all possible estimation metrics like cost, effort and size.

The remaining section of the paper is organized as follows: Section 2 presents Estimation techniques in traditional software development and Section 3 is about Estimation techniques in agile software development. Finally, Section 4 concludes the paper with future directions.

2. Estimation Techniques in Traditional Software Development

Estimation is the major challenging tasks for project managers, clients and for the development teams. In traditional approach, even though the requirements are well-defined and not subject to changes, still it is difficult for managers to accurately estimate the time, effort and the budget required to develop a software system. There are many software estimation techniques available such as algorithmic based estimation, estimation based on Expert Judgment, Analogy based estimation, top-down and bottom-up methods as discussed in [6-14] and so on. But, due to its simplicity and success factors, most software organizations prefer the following estimation models [15-17] as their primary choice.

a. Algorithmic based model in which Use-Case Point Estimation, Function Point Analysis (FPA) and COnstructive COst Estimation MOdel (COCOMO / COCOMO –II) are widely considered.
b. Analogy based estimation, and
c. Expert opinion Method.

2.1. Algorithm Based Method

Algorithmic method is based on using the mathematical formulas and equations to perform the estimation. This technique is widely used with the help of multiple contributing factors like historical data, source lines of code (SLOC), number of functionalities involved in the system, skill levels and risk factors etc. Examples for algorithm based estimation models are Function point analysis (FPA), COCOMO and Putnam models. The major advantages of these methods are: highly efficient, ability to easily modify and customize the data and formulas. This

method becomes ineffective in the case of improper cost driver selection or absence of historical data.

Kenneth Lind and Rogardt Heldal [18] approach is to estimate the size of embedded software components. For embedded system application, estimating accurate code size early saves huge amount of cost and effort for developing the component. COSMIC Functional size measurement method has applied in different automotive industries. The study proves a strong correlation exists between functional size and code size, which is crucial for obtaining more accurate estimation outcomes.

Ali Bou Nassif *et al.* [19] compared decision tree forest (DTF) model against decision tree (DT) model and multiple linear regression model (MLR) for estimating the development effort required for a project. The above said models are evaluated and the results are compared against the evaluation criteria such as MMRE, MdMRE & PRED(x). The result proves that the DTF model outperforms the other two models (DT and MLR) in terms of effort estimation accuracy. However, due to this heavyweight approach, this model might not be suitable for agile based projects.

Ekrem Kocaguneli *et al.* [20] proposed a simple active learning method called QUICK, that reduces the complexity of data representation and provides a recommendation about which type of effort estimation (simple or complex) is suitable for simple project datasets. The activities involved in this method are (i) Grouping rows and columns based on their similarities, (ii) Discard the repeatable columns and outlier rows based on their similarities and uniqueness respectively and (iii) Generate an effort estimate with the remaining data from nearest example. This approach might not be suitable for complex methods (datasets) as this method focus is only on simple project datasets.

Florian Schnitzhofer *et al.* [21] introduced a new tool called pocket-estimator, a cloud based framework used to estimate the software development effort. Their main goal is to develop a huge software development project datasets. They have used a combination of expert weighted estimation algorithm with learning algorithms in order to predict the effort more precisely. However, this method does not address the other estimation factors such as cost, time and the size.

Existing estimation techniques has many challenges due to the recent development in emergent technologies and frameworks. Zia *et al*. [22] presented a cost estimation technique suitable for component based fourth generation environments. This method uses the existing COCOMO-II technique for size estimation. In addition, for making it compatible with today's component based environments, the total size of the components used in the development is determined by adding the number of database objects such as tables used in the software. This approach is applied in several component based development projects and results are proved to be robust, stable and improved accuracy level when compared with other similar approaches.

Many researchers even today consider software size as a critical variable for estimating the effort for a given project [23][24][25]. One approach to estimate effort is to determine the ratio of effort for planning and requirements phase to the effort for whole development phase. Masateru Tsunoda *et al.* [24] built an effort estimation model based on the effort for pre-development activities and the software application size. The effort used for pre-development phase activities are considered as explanatory variable and the estimation accuracies are determined by comparing the proposed model with the size based effort estimation models. This model proves that there is a considerable improvement in the estimation accuracy when we use both effort required for pre-construction activities and application size as explanatory variable.

Ali Bou Nassif and team [26] proposed a new method CCNN, called Cascade Correlation Neural Network model for estimating cost based on use case diagrams. Project complexity, teams' productivity and the size of the software are given as input to the developed model. Multiple linear regression model was developed with the same set of input parameters. The proposed use case point method is validated against linear regression model and the results suggest that the use case point model outperforms other regression methods based on MMER and PRED criteria's.

2.2. Expert Judgement Approach

This method involves a discussion with a team of estimation experts and their domain knowledge and experiences are utilized to arrive at the estimation. Delphi is the widely used technique in this category.

The steps involved in Delphi methods are:

a. Project coordinator will schedule for a meeting with team of experts.
b. Coordinator elaborates the requirements and the experts anonymously fill out the estimation for each requirement.
c. The requirement that have significant difference in the estimate values are chosen further for detailed discussion.
d. The above steps are repeated for all available specifications.

There is no single best estimation method suitable for all type of projects. Thus, Ekrem Kocaguneli *et al.* [27] proposed a methodology that combines multiple estimation methods into one. The best solo independent estimation methods are chosen, the selected method has then applied to the datasets and validated using seven critical error measures. This paper confirms that ensembles of multiple solo methods are more consistent and accurate in estimation compared to the independent methods.

Barry Boehm [28] conducted a study based on literature analysis, conducting interviews and surveys with leading resource estimators and users, and concludes that the study on resource estimation has a direct impact with software engineering practices. Aihua Ren and Chen Yun [29] worked on most widely used size estimation models like analogy based Function Point, PERT, Delphi and other methods. Not all methods are suitable for all type of projects. Based on the advantages and limitations of each models, this approach suggests on how to choose the most appropriate method for the given project requirements. However, this approach might not be applicable for agile based projects because of the fact that the requirements in agile are not stable and also not well defined in up-front.

Nikolaos Mittas [30] proposed a statistical model, ranks several estimation models based on multiple comparisons algorithm. This method provides satisfying results by identifying the group of models that have significant differences in accuracy and clustering them into non-overlapping groups. The final decision on choosing appropriate model is based on experts and their personal preferences like familiarity of software and user experiences.

2.3. Estimation by Analogy

Estimation by analogy is a straightforward technique where the estimation is performed based on the comparison of proposed project with similar completed projects. The steps involved are: Categorize the project, compare the proposed project with the completed one based on the similar characteristics, and derive the estimate by analogy. The main advantages of this approach are: Estimation is based on actual project characteristics, estimator's experience and their domain knowledge are also taken into considerations. The drawback with this method is that since the referenced project data are subjective, there is a possibility of biased estimation due to the fact that the values of similar projects are known prior to the estimation.

Analogy-based effort estimation is one of the most prominent techniques used to handle noisy datasets. Same number of analogies may not be suitable for all type of projects to make accurate estimates. Mohammad Azzeh and Ali Bou Nassif [31] proposed a new method based on bisecting k-medoids clustering algorithm to come up with set of analogies for individual projects. The dataset characteristics are gathered by applying the above said algorithm that in turn able to automatically find the set of analogies for each project. The results are compared against the traditional analogy-based estimation and the proposed technique delivers more promising results with enhanced performance.

Ekrem Kocaguneli [32] identified the assumption about analogy-based effort estimation. These assumptions are then tested by obtaining binary tree of clusters and compare the variance of super-trees against sub-trees. It is found that estimating the variance of super-trees are smaller than sub-trees, implies that if cluster variance is lower than the estimation has minimal occurrence of errors. This paper concludes that dynamic selection of nearest neighboring project by using the project data with lesser variance significantly improves the estimation by analogy.

Analogy-Based estimation and artificial neural networks are the most commonly used methods to estimate the effort required for a software project development. Khatibi Bardsiri *et al.* [33] proposed a hybrid method, is a combination of fuzzy clustering, analogy-based and artificial neural network methods used to improve the accuracy of effort estimation. The related projects are clustered so that the inconsistent and irrelevant projects are reduced which in turn

improves the estimation accuracy. The proposed framework has been validated based on the performance metrics such as MMRE and PRED (0.25) and the promising results proven that this method outperformed all other methods.

Elham Khatibi [34] proposed a new technique to address the problem in Analogy based estimation. Analogy based estimation is more widely used technique in recent years, because of its simplicity and estimation capability used for estimating the effort required to develop a product. The earlier methods compared the two related projects without considering their internal attributes which leads to inaccurate and biased estimates. This paper focuses about developing a hybrid model to address the above said problem. The related projects are classified as group of clusters by considering the internal project attributes such as development platform, organization type and level of expertise. Then the attribute weighting process should be performed and the development effort was analyzed for each cluster of projects. The achieved results are validated by comparing with the existing model and this method delivers a promising result with respect to accuracy and performance metrics.

Magne Jorgensen [35] conducted a study to evaluate the effect of relative based estimation model. The assessments are based on simple comparisons of effort related attributes of the current projects with that of completed similar projects. Based on the analysis, author suggests the following guidelines that will benefit the developers in terms of accuracy while performing the estimation in order to improve the estimation accuracy. A) When using analogy based estimation, comparison with similar projects can be made in terms of work hours rather than percentages. B) Provide importance to the unique properties of the reference project. C) It is advised to perform the estimation on a sequential order of the task size. For example, start with the estimation of small size component first and then move on to medium and large size components. This will improve the estimation accuracy by addressing the logical dependence exists between the modules. Table 1 shows the comparison about different estimation models used in traditional software development. It is represented by grouping the work based on various estimation categories and highlights the estimation attributes the corresponding papers mainly focuses on.

Table 1. Comparison of Software estimation attributes in Traditional development Model

S.No	Estimation Category	Authors	Estimation Attributes			
			Size	Cost	Effort	Others
1	Algorithm Based Method	Kenneth Lind et al. [28]	□			
2		Ali Bou Nassif et al. [32]			□	
3		Ekrem Kocaguneli et al. [26]			□	
4		Florian Schnitzhofer et al. [37]			□	
5		Zia et al. [48]	□	□		
6		Masateru Tsunoda et al. [45]			□	
7	Expert Judgement	Ali Bou Nassif et al. [3]		□	□	
8		Ekrem Kocaguneli et al. [25]			□	
9		Barry Boehm [9]				□
10		Aihua Ren et al. [4]	□			
11	Analogy Based Estimation	Nikolaos Mittas et al. [30]		□		
12		Mohammad Azzeh et al. [3]			□	
13		Ekrem Kocaguneli et al. [24]			□	
14		Khatibi Bardsiri et al [22]			□	
15		Elham Khatibi et al. [23]			□	
16		Magne Jorgensen et al. [20]			□	

3. Estimation Techniques in Agile Software Development

Accurate estimation is the most difficult task among project managers and stakeholders; more so if this happens to be an agile project. Among agile teams, estimation is the most difficult aspect of the job. The most important research question among agile researcher is: How do we make the estimates as accurate as possible without losing the agility? Unlike the number of estimation methods available for traditional development models, agile has very simple and minimal set of estimation techniques that are widely used. To find answer for the above question, Dan Radigan [36] suggested the following contributing factors to make an efficient estimation for agile based projects.

3.1. Collaborating with Product Owners

In agile development, product owner is responsible for capturing and decomposing requirements from business, prioritizing product backlog but they do not have much technical knowledge on implementation details. Agile development team usually performs the estimation process by co-coordinating with Scrum Master and Product Owner. The estimates help the product owner to prioritize the product backlog.

3.2. Agile Estimation is a Team Sport

Unlike traditional approach, estimation in agile is a team effort. The team in agile involves developers, designers, testers, scrum master etc. Each member brings a different perspective and the work required to deliver a feature [37].

3.3. Agile Story Points

Traditional approach provides estimates in calendar days or weeks, but in agile, story point is the widely used metric for estimation. Story point is a measure used by agile teams to measure the effort required to implement a story. Story points are usually referred by using Fibonacci sequence or with the relative size such as small, medium and large.

Planning poker is most widely used technique for estimation in agile. Agile development team along with product owner and scrum master will gather in a room and the agile team will take a user story from product backlog, discuss it briefly and suggest an estimate in terms of story points. If all estimators selected a same value for a given user story, then that becomes the estimate. If there is any difference exists between the estimators, then the user story is picked for further detailed discussion. The process continues until consensus among the team is achieved. The role of the Product Owner is to elaborate the user stories to the team to perform the estimates, clarifying the development team questions on user stories and the final decision maker. The role of the Scrum Master is to coach the team, resolve the ambiguity or contradictions that arise among the teams, facilitate the session and the central point-of-contact for Product Owners and development teams.

Agile story point is a relative measure which has no direct relation with calendar days. This makes the agile team to think abstract about the effort required to complete the feature. Table 2 lists the various estimation techniques widely used in Agile based software development projects. The main advantages of using story points over calendar days [36] are:

a. Calendar days do not account non project related tasks like meetings, emails and interviews.
b. Normally calendar days have an emotional attachment.
c. Each team uses different scale to estimates, in turn their velocity becomes different.

Since agile is the most recommended development model being adopted by many organizations in recent years, it is essential to understand how the estimation process is performed in agile. Most software engineering researchers believes that the development phases in agile is easy and simple to be performed compared to traditional method [38-42]. C.J. Torrecilla-Salinas et al [43] proposed a framework for estimation and planning of web based projects suitable for Scrum based agile projects. This approach is based on value-based perspective by combining several existing agile techniques. The proposed framework is validated by real-life project case studies in order to obtain the accurate conclusion. This approach is highly suitable for managing, planning and estimating web based agile projects.

Ishrar Hussain *et al.* [44] developed a method to approximate the functional size of COSMIC standard. COSMIC is an ISO standard used to measure the functional size of software based on user requirements. However, this approach is not suitable for agile process, as this

requires the user requirements to be formalized and decomposed. This paper address the issues by approximate the COSMIC functional size from in-formal textual requirements that suits with the agile process.

Sungjoo Kang *et al.* [45] created an estimation model for agile based projects based on function points. Function points are widely used to estimate the cost and effort that are required to develop a product. This approach is widely used in traditional approach. In agile, most widely accepted estimation technique is based on story points. In this paper, they have incorporated the function point approach in addition to story points to achieve the highest level of accuracy. The project status is dynamically tracked with the help of Kalman filter algorithm. The validation is performed with the help of case study by comparing the results with the traditional approach.

Use case point estimation is one of the famous models used for estimation in agile software development. Parvez [46] developed a new layer in the existing use case point estimation model in which they have introduced two contributing factors namely: efficiency and risk for estimating the effort required for testing. The existing use case point method considers only the properties of the project but this paper focuses on the team properties in addition to the project. The important factors to be considered in the new layer are Test team resources, Cycle length, Weight of testing, Efficiency factor and risk factors. The introduction of new layer in the existing use case point improves the effectiveness and performance of the estimation.

Sakshi Garg *et al.* [47] proposed cost estimation model that suites for Agile software development projects. Sakshi identified the attributes that have maximum correlation and introduced Principle Component Analysis (PCA) for reducing the number of considerable attributes. The proposed approach is suitable even in the absence of statistical data and expert opinion. The result from this approach proves to have a better precision and accuracy of cost estimation in agile software development projects.

Story Point Approach (SPA) is the most widely used approach in agile software effort estimation. Aditi Panda *et al.* [48] improves the estimation accuracy in agile projects based on neural networks. This approach considers different types of neural networks like General Regression neural networks (GRNN), polynomial neural networks and probabilistic neural networks to improve the accuracy of the effort estimation. This method considers best fit for estimating the effort, however it does not address other aspects of estimation like cost, duration or risk.

Kayhan Moharreri et al. [49] proposed an automatic estimation method called “Auto Estimate” for estimating effort for agile based projects. This approach is complementing to widely used manual planning poker technique. The best learning technique is selected automatically by performing the following steps: Data collection by using story cards, extracting the features using textual analysis, constructing the model with extracted features and performs analysis by measuring the performance. It also provides promising results with respect to accuracy as compared with usual planning poker technique.

Table 2. Comparison of various estimation methods in Agile based projects

s. No	Authors	Estimation Technique Used	Estimation Attributes			
			Size	Cost	Effort	Others
1	C J Torrecilla-Salinas et.al. [43]	Value Based	✓	✓	✓	
2	Ishrar Hussain et. al. [19]	COS MIC, planning poker	✓			
3	Sungjoo Kang et. al. [40]	Function Point based, planning poker	✓	✓		
4	Parvez et. al. [35]	Use case Point based, planning poker				✓
5	Sakshi Garg et.al. [16]	Principle Component Analysis		✓		
6	Aditi Panda et.al. [1]	Neural Network based			✓	
7	Kayhan Moharreri et.al. [21]	Auto estimate, planning poker			✓	

The following are the observations we make on the available estimation models used in agile approaches.

a. Planning poker is the most widely used technique, which is highly dependent on expert judgements and availability of historical data. This technique will not be useful if we do not have any experts or unavailability of historical data in our projects.

b. Estimation is mainly based on Story points, which are relative in nature. A small change in user stories in product backlog will eventually leads to the changes in story points for all other user stories which makes the estimation meaningless.
c. No straight forward technique is available to determine the teams' velocity. It is usually calculated based on assumptions and trials which might impact the project deliverables.

4. Conclusion

It is essential for any organization to find the most accurate software estimation models for taking a valid management decision and useful for project managers, development teams and clients to have a proper project planning. There are many estimation techniques available for both traditional and agile models, but the estimation methods that suits for one may not be compatible for other. Therefore, it is essential to identify an appropriate estimation model based on the development process model used for the software construction. Also, no single estimation techniques are suitable for all kinds of projects. Therefore, it is necessary to choose the appropriate estimation models by keeping the project characteristics and requirements in mind. For any type of software development, estimation is always a critical and important task.

We have summarized the various estimation approaches that are widely followed in traditional as well in Agile based development projects. In general, it is very difficult to say which estimation method is exceptional due to the fact that it is also highly dependent on project size and various other factors. But, COCOMO-II and Function point based are largely used in traditional approach and planning poker is the most widely accepted one in Agile based projects.

In recent years, agile development models are getting widely adopted for developing software products in software organizations. But, based on research literature reveals that only limited amount of work has been carried out so far in the area of estimation in agile based projects. Further, an interesting future direction in the field of software estimation for Agile development projects could be focus on: (1) In addition to Scrum, estimation of cost and effort in other agile methods such as Extreme Programming (XP), DSDM, crystal, Lean and so on, (2) Identification of various risks that might influence the project and its outcomes in the earlier stage of sprint cycle, (3) It is essential to develop more standardized estimation methods like algorithmic based that we have in traditional environments, (4) Validation of estimation results with large scale industry projects.

References

[1] Cohn T, Paul, R, Cohn, T M, Paul, R C, " *A comparison of requirements engineering in Extreme Programming (XP) and conventional software development methodologies*," Proceedings on AMCIS, 2001: 256.
[2] Agile Manifesto, "Manifesto for Agile Software Development", http://agilemanifesto.org/,December2007
[3] Beck, K, *et al* "Manifesto for Agile Software Development". Agile Alliance, 2001.
[4] Nidhi Sharma, Manoj Wadhwa, "eXSRUP: Hybrid Software Development Model Integrating Extreme Programming, SCRUM & Rational Unified Process", *TELKOMNIKA Indonesian Journal of Electrical Engineering*, 2015; 16 (2): 377 – 388.
[5] The Standish Group International, "The CHAOS Report", http://www.standishgroup.com/sample_research/chaos_1994_1.php
[6] Black, R K D, Curnow, R P, Katz, R, Gray, M D, BCS software production data, Final Technical Report, RADC-TR77-116, Boeing computer services Inc., 1977: 5-8.
[7] Boehm, B W, Software engineering economics, Englewood Cliffs, NJ: Prentice-Hall, 1981.
[8] Donelson, W S, Project Planning and Control, Datamation, 1976: 73-80.
[9] Herd, R J R, Postak, J N, Russell, W E, Steward, K R, "Software cost estimation study: Study results, Final technical report," *RADC-TR77-220, Doty Associates, Inc., Rockville, MD* 1977; I: 1-10.
[10] Park, R E, "*PRICE S: The calculation within and why,"* Proceedings of ISPA Tenth annual conference, Brighton, England, 1988: 231-240.
[11] Parkinson, G N, Parkinson's law and Other studies in Administration, Houghton-Miffin, Boston, 1957.

[12] Putnam, L H, "A general empirical solution to the macro software sizing and estimating problem," *IEEE Trans. Soft. Eng.,* 1978: 345-361.
[13] Tausworthe, R, Deep Space Network Software Cost Estimation Model, Jet Propulsion Laboratory Publication, 1981: 67-78.
[14] Walston, C E, Felix, C P, "A method of programming measurement and estimation," *IBM Systems Journal*, 1977; 16 (1): 54-73.
[15] Bruce Benton, "Model-Based Time and Cost Estimation in a Software Testing Environment." Information Technology: New Generations, (IING), 2009: 801-816.
[16] Krishna Mohan, K, Verma, A K, Srividya, A, "Software Reliability Estimation Through Black Box and White Box Testing at Prototype Level," *Reliability, Safety and Hazard (ICRESH),* 2010: 517-522.
[17] Li-Xin Jiang, Wan-Jiang HAN, Chen-Chen YAN , Bo-Ying SHI, "*Research on Size Estimation Model for Software system Test based on testing steps and Its Application,"* International Conference on Computer Science and Information Processing (CSIP), 2012: 1245-1248.
[18] Lind, K, Heldal, R, "A Practical Approach to Size Estimation of Embedded Software Components," *in Software Engineering, IEEE Transactions on, 2012*; 38 (5): 993-1007.
[19] Nassif, A B, Azzeh, M, Capretz, L F, Ho, D, *"A comparision between decision trees and decision tree forest models for software development effort estimation,"* in Communications and Information Technology (ICCIT), Third International Conferencce on 2013: 220-224.
[20] Kocaguneli, E, Menzies, T, Keung, J, Cok, D, Madachy, R, "Active learning and effort estimation: Finding the essential content of software effort estimation data," in Software Engineering, IEEE Transactions on, 2013; 39 (8): 1040-1053.
[21] Schnitzhofer, F, Schnitzhofer, P, "*Pocket Estimator – A commercial Solution to provide free parametric software estimation combining an expert and a learning algorithm,"* Software Engineering and Advanced Applications, 38th EUROMICRO conference, 2012: 422-425.
[22] Zia, Z, Rashid, A, Zaman, U, "Software cost estimation for component based fourth-generation-language software applications," *in IET software*, 2011; 5 (1): 103-110.
[23] Boehm, B, Software Engineering Economics, Prentice Hall, 1981.
[24] Shepperd, M, Schofield, C, "Estimating software project effort using analogies," IEEE Transactions on Software Engineering, 1997; 23 (11): 736-743.
[25] Tsunoda, M, Kamei, Y, Toda, K, Nagappan, M, Fushida, K, Ubayashi, N, "*Revisiting software development effort estimation based on early phase development activities,"* Mining Software Repositories (MSR),10th IEEE Working Conference on , San Francisco, CA, 2013: 429-438.
[26] Nassif, A B, Capretz, L F, Ho, D, *"Software Effort Estimation in the early stages of the software life cycle using a cascade correlation neural network model,"* Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD), 13th ACIS International Conference on Kyoto, 2012: 589-594.
[27] Kocaguneli, E, Menzies, T, Keung, J W, "On the value of ensemble effort estimation," in Software Engineering, *IEEE Transactions on,* 2012; 38 (6): 1403-1416.
[28] Boehm, B., Valerdi, R, *"Impact of software resource estimation research on practice: a preliminary report on achievements, synergies, and challenges,"* in Software Engineering (ICSE), 33rd Internation Conference, 2011.
[29] Aihua Ren, Chen Yun, "*Research of Software Size Estimation Method,*" Cloud and Service Computing (CSC), International Conference on, Beijing, 2013.
[30] Mittas, N, Angelis, L, "Ranking and Clustering Software Cost Estimation Models through a Multiple Comparisons Algorithm," in IEEE Transactions on Software Engineering, 2013; 39 (4): 537 – 551.
[31] Azzeh, M, Nassif, A B, "Analogy-based effort estimation: a new method to discover set of analogies from dataset characteristics," *in IET Software,* 2015; 9 (2): 39-50.
[32] Kocaguneli, E, "Exploiting the essential assumptions of Analogy-Based Effort Estimation", *IEEE Transactions on Software Engineering*, 2012; 38 (2): 425-438.
[33] Khatibi Bardsiri, V, Jawawi, D N A, Hashim S Z M, Khatibi E, "Increasing the accuracy of software development effort estimation using projects clustering," in IET Software, 2012; 6 (6): 461- 473.
[34] Khatibi, E, Khatibi Bardsiri, V, "Model to estimate the software development effort based on in-depth analysis of project attributes," *in IET software*, 2015; 9 (4): 109-118.

[35] Jorgensen, M, "Relative Estimation of Software Development Effort: It matters with What and How You compare," *in IEEE Software,* 2013; 30 (2): 74-79.
[36] Dan Radigan, "Collaboration, abstraction, and other secrets of agile estimation", http://atlassian.com/agile/estimation.
[37] Rajesh H Kulkarni, P Padmanabham, Manasi Harshe, K K Baseer, Pallavi Patil, "Investigating Agile Adaptation for Project Development", *International Journal of Electrical and Computer Engineering (IJECE)*, 2017; 7 (3): 1278 - 1285.
[38] Chandra, S, Kumar, V, Kumar, U, "Identifying some important success factors in adopting agile software development practices," *The Journal of systems and software*, 2009; 82 (11): 1869-1890.
[39] Eva-Maria Schon, Jorg Thomaschewski, Maria Jose Escalona, "Agile Requirements Engineering: A systematic literature review", Computer Standards and Interfaces, 2017; 49: 79-91.
[40] Green, R, Mazzuchi, T, Sarkani, S, *"Communication and Quality in Distributed Agile Development: An Empirical Case Study,"* Proceeding in World Academy of Science, Engineering and Technology, 2010; 61: 322-328.
[41] Stephen, A L, "Improvisation and agile project management: A Comparative consideration", *International Journal of managing projects in business*, 2009; 2 (4): 519-535.
[42] Tsirakidis, P, *"Identification of Success and Failure Factors of Two Agile Software Development Teams in an Open Source Organization,"* The 4th International Conference on Global Software Engineering, Limerick, Ireland, 2009: 2-3.
[43] Torrecilla-Salinas, C J, Sedeno, J, Escalona, M J, Mejias, M, "Estimating, planning and managing Agile web development projects under a value-based perspective," *Information and Software Technology*, 2015; 61: 124-144.
[44] Ishrar Hussain, Leila Kosseim, Olga Ormandjieva, "Approximation of COSMIC functional size to support early effort estimation in Agile," *Data & Knowledge Engineering*, 2013; 85: 2-14.
[45] Sungjoo Kang, Okjoo Choi, Jongmoon Baik, *"Model based dynamic cost estimation and tracking method for Agile Software Development,"* in Computer and Information Science (ICIS), IEEE / ACIS 9th International conference on, 2010: 743-748.
[46] Parvez, A W M M, "*Efficiency factor and risk factor based use case point test effort estimation model compatible with agile software development,"* Information Technology and Electrical Engineering (ICITEE), International Conference on, Yogyakarta, 2013: 113-118.
[47] Garg, S, Gupta, D, "*PCA based cost estimation model for agile software development projects,"* Industrial Engineering and Operations Management (IEOM), International Conference on, Dubai, 2015: 1-7.
[48] Aditi Panda, Shashank Mouli Satapathy, Santanu kumar Rath, "Emprical validation of Neural Network Models for Agile Software Effort Estimation based on Story Points," *Procedia Computer Science*, 2015; 57: 772-781.
[49] Kayhan Moharreri, Alhad Vinayak Sapre, Jayashree Ramanathan, Rajiv Ramnath, *"Cost-Effective Supervised Learning Models for Software Effort Estimation in Agile Environments",* IEEE 40th Annual Computer Software and Applications Conference, 2016.