Improvement of Agile Software Development Size & Effort Estimation Methods

Anmar Abdullah Mohammed, Azizah Ahmad, Mazni Omar

Abstract: This paper aims to lay down the existing evidence concerning agile software estimation, it summarizes the empirical evidence of the existing agile estimation methods, identify gaps in the current research in order to suggest areas of further investigation and improvement, and to provide a basis for the improvement of agile software estimation research through a systematic review of previous work. The contribution of this paper is twofold, first is providing a basis for the improvement of agile software estimation through a systematic literature review of previous work. Secondly, it presents a recommendation for further research based on the results presented in this paper and the recommendations by well-known researchers in the field of software engineering in general and in software estimation in particular.

I. INTRODUCTION

Software development effort estimation is considered a crucial part of software development and software project management. In Software development, effort estimation can be defined as a method taken to estimate the most realistic effort usually measured in person-hours or money needed to complete a software development it is usually used in project planning stage by managers, Iteration planning, pricing process, project bidding, investment analysis. Generally, estimation is based on incomplete inputs, uncertain and noisy inputs through a process to predict the software effort required typically expressed in person-hours [1].

Generally, when software development carried out cost estimated is directly related to effort and some of the cost factors are personal in a sense it depends on the company and their policies required by development team for developing a software. Most of studies focus on human effort in software estimation that is due to human are the most dynamic part of software development and human effort it can be seen on almost all estimation methods proposed by the literature.

Revised Manuscript Received on May 22, 2019.

Anmar Abdullah Mohammed, School of Computing, University Utara Malaysia (UUM)

Azizah Ahmad, Institute for Advanced and Smart Digital Opportunities, School of Computing, University Utara Malaysia

Mazni Omar, School of Computing, University Utara Malaysia (UUM)

Due to that fact generally in literature there is no distinguish between cost and effort, in this research cost and effort is treated as effort [2], added this research focuses on the effort That's no surprise because software development has become the most expensive component of computer system projects [3]. On the other hand, some researchers focused on estimating software size before committing to a software project using sizing methods and metrics and its used as a base in some estimation models for effort due to its capabilities see figure 1. Software Sizing is the probable size of a software project and its unlike effort, as effort is the effort needed to accomplish building software. Software size is considered the tangible way of measure a software probable size same as a weight is the tangible size of a material [4].

The introduction of agile methodology in early 2000s open a new way of developing software systems, companies has been adapting agile since its early inception. Agile promotes the use of continues planning, iterative development, value delivery, and continues improvement, It promotes rapid changes to requirements [5]. Agile can be summarized as an approach to software development through requirements and solutions evolve through continues effort of self-organized team under which each member of the team poses cross-functional skills[6]. Agile methodologies in its core principles is accommodating requirement changes better, making it possible for development teams to react quickly when changes are necessary[7].

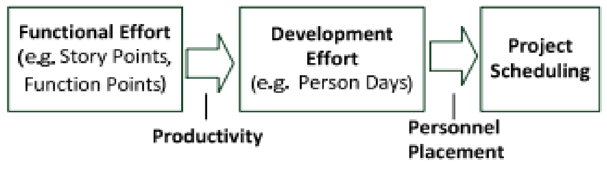
Functional Effort (e.g. Story Points, Function Points)
Development Effort (e.g. Person Days)
Project Scheduling
Productivity
Personnel Placement

Fig. 1 Size relation to effort and scheduling (Raith, F., Richter, I., Lindermeier, R., &Klinker, G., 2013, December)

Many studies exist in the literature in regards to software estimation based on traditional approaches of software development and they proved its value such as COCOMO, and COCOMOII. However, these techniques do not fit in the context of agile though there are some studies tried to fit COCOMO II Into agile such is in[8].

Furthermore, [9]defended COCOMOII model by elaborating that projects that use agile approach need to handle estimation differently due to its short iterations and high uncertainty in project especially in its early inception.

Systematic literature review has been carried out in software engineering field and on estimation previously like in [10], [11], [12]. this study is a complimentary to their work and adds more emphasis on methods adaptation. [12] tried to map out the literature from 2001 till 2013 with focus on the main methods used, metrics, cost drivers, and accuracy measurements used to evaluate methods, which was reported successfully, however, their research did not investigate the AI techniques in details as well as the current year of this research is 2019. [10] have reviewed several studies but with focus on few journals and not a full spectrum of the major research databases, and the research collection was up to 2007.

This study is divided into several sections, Section one introduces the topic and some of the literature related followed by sections two starts with review protocol used and the primary studies selected, section three states the data extracted from different studies and elaborate on the data from different prospective to answer this study research question. Section Four is the discussion on the results identified from section three and state the gapes and recommendation for further research, finally conclude the paper with thread and validity, and conclusion.

II. SYSTEMATIC LITERATURE REVIEW

This section details the systematic literature review, which includes the research questions that guided this systematic literature review as well as the procedures employed to perform the entire systematic literature review process. The systematic literature review detailed herein followed the guidelines defined by[10], [13]. The protocol was developed jointly by the first, second and third authors

Research Questions

The research questions were formulated on the bases that we needed to understand the trend in software estimation and its map by formulating research questions based on PICOC criteria by[14] and starts with i) Population –Agile Software Development. ii) Intervention - Effort estimation methods/ techniques/approaches/Estimation size/cost. iii) Comparison – Not used. iv) Outcomes-effort estimation methods/ techniques. v) Context - Any possible study, as long as it is an empirical study or academic theory from academic within the context of estimation in agile software development will be considered but not necessary included as a primary study until it passes inclusion criteria and quality assessment. Therefore, the addressed research questions are:
Research Question 1: What are the existing methods used for effort estimation under agile software development?
Research Question 2: What are the existing metrics for size estimation under agile software development?

Search Strategy

Formulating the Research was the first step in the systematic literature review, at this step a strategy for research studies need to be implemented therefore a decision where taken to adapt the strategy from the successful systematic literature reviews in software engineering which its guidelines where suggested by Kitchenham[15], it lays the ground for primary study selection and it help this research to avoid bias, the study selection is as follows:

- Derivation of major terms from the research questions
- Identification of alternative spelling and synonyms for major terms
- Identifications of keywords in relevant papers
- Usage of Boolean OR to incorporate alternative spellings and synonyms
- Usage of the Boolean AND to link major terms.

Following the previously stated rules the resulting search terms are described as follows:
Agile AND ("Estimat*" OR "Predict*") AND ("Cost" OR "Size" OR "Effort") However, each database requires specific rules to accept the above query therefore, some modification where necessary for each database without losing query objective.

Search Process

To answer our research questions, we performed an automated search based on the pre-constructed search terms on the following electronic databases: IEEE Digital Library, ACM, Digital library, Science Direct, and Scopus. The IEEE, ACM, Scopus, and Science Direct Digital Libraries were chosen because most of the publication venues of selected papers in the previous systematic literature review on software development effort estimation used it [10]–[12]. All the searches were limited to articles published from 2008 until august 2018 the rationale behind this time period was that most of the reviews limit their research to 5 or 8 or 10 years as seen in [10], [12] and suggested by [16] that if less than 10 years could severely limit the number of eligible studies. The search was conducted separately in the IEEE, ACM, Scopus, and Science Direct databases based on title, abstract, and keywords. The summary of results set stated in table 1

Study Selection

Inclusion Criteria

Any study under agile method, described in English, reported in peer reviewed conference or journal, and effort or size or cost estimation as the main theme are considered in this study.

Exclusion Criteria

As for study exclusion criteria goes through these criteria. Are not about effort or cost or size estimation, that are not conducted using any of the agile software development methods, are not described in English, are not published in peer reviewed conference or journal, deals with software testing, deals with software maintenance, deals with performance measurement such as velocity measurement, and finally a duplicate publication of the same study. Any study falls under any of these criteria will be excluded. After performing inclusions/exclusion rules results in table 1.

Table. 1 Study Selection Process Final Set

Database Name	Initial Results		Selected Article
IEEExplore	516		58
ACM D. L.	1621 - 1		20
ScienceDirect	288		27
Scopus	484		86
		Initial no.	191
		Duplication	40
		Final Initial set	151

Quality Assessment

At this stage a quality assessment checklist were included to improve the primary studies selection, study quality assessment was adapted from Kitchenham[15]. It also used by other researchers in performing systematic literature reviews resembling in [11], [12], [17] added they also have customized the quality assessment to their study objectives. In this study the aim to identify the current software estimation process in agile software development. Therefore, a quality assessment on the papers were carried out and rules are stated in Table 2. A scoring system used to weight each study by reviewing its content based on the questions stated in table 2 for every question is answered in the study a 1.0 score will be added if partially answered then a 0.5 score will be added if not answered then 0.0 score will be added on the total. Any paper does not reach a score of 3.0 will be excluded from the study. This way it helps increase the quality of the primary selected papers for this review.

Table. 2 Quality Assessment Questioner

No.	Questions	Score		
		No	Partial	Yes
1.	Are the research aim clearly specified?	0.0	0.5	1.0
2.	Was the study designed to achieve these aims?			
3.	Are the proposed estimation techniques clearly described and their selection justified?			
4.	Are focused on Estimation as the core of the study and does not depart from the core.			
5.	Are estimation drivers stated and explained			
6.	An evaluation procedure where carried out			

The above questions where instigated on the 151 studies of the final set and the final set of studies selected as a primary study are 38 studies which are S1, S2, S3, S4, S5, S6, S7, S8, S9, A10, S11, S12, S13, S14, S15, S16, S17, S18, S19, S20, S21, S22, S23, S24, S25, S26, S27, S28, S29, S30, S31, S32, S33, S34, S35, S36, S37, and S38. A full reference in appendix A.

Data Extraction

The final set of papers gone through a full text reading and analysis of the paper by the first author and evaluated by the second author and third author. This process where continued until consensus was reached among authors. Finally, all the extracted data were discussed among authors and compared before final discussion were made to include, otherwise, other review were carried to reevaluate extracted data.

III. RESULTS

In the previous section a full detailed explanation of study selection protocol and the final set of papers. In this section, however, a description is presented of the SLR process and for each of the research questions specified in this research. As presented in 2. Concluded out research selection to 38 studies as the primary studies for this review to answer the research questions specified. This section is divided based on research questions to give a full focus and elaboration on the process of the answers concluded.

Research Questions 1: Estimation Methods

26 studies where selected after full reading for estimation methods (S1, S4, S9, S10, S11, S12, S13, S14, S15, S16, S17, S18, S19, S22, S24, S25, S26, S27, S28, S29, S31, S32, S33, S34, S36, S37). These studies where selected after a full reading and analysis of their content followed by data extraction from its content summarized in table 3. Some studies have used sizing method for effort estimation like Story points and FSM and that’s no surprise because in agile software development context effort estimation is highly dependent on sizing in many practices due to requirements consistence change[18][19].

Several studies have revaluated existing known methods and some have tried to extend it like in S10 and S1. Furthermore, most used methods reported in the studies and its derivative are Expert Judgment, Planning poker, Story Points, FSM, and it’s reported they provide a good estimate accuracy such in S28, S2, S26, and S31. A summary of the methods and its related studies presented in table 3 with its major classification.

Table. 3 Estimation methods

No.	Estimation Method/Technique	Study
1.	Planning Poker	S2, S28, S30, S24, S34, S1, S10, S8, S36, S15, S13, S19
2.	Expert Judgment	S28, S24, S1, S10, S11, S36, S14, S35, S19
3.	Use Case Points (UCP) Method	S28, S24, S36, S35, S19
4.	Historical Data (Analogy)	S28, S27, S25, S7, S36, S19
5.	Functional Size Measurement	S28, S27, S22, S31, S24, S25, S32, S34, S33, S36, S29
6.	Delphi Technique	S10, S36, S19
7.	Story Point	S26, S34, S9, S33, S36, S14, S18, S29, S37
8.	COCOMO	S32
9.	Deep-SE Deep Learning	S9
10.	Text Mining	S22, S24
11.	HyEEASe (hybrid effort estimation in agile software development)	S1
12.	AI Machine Learning	S12, S15
13.	Ontology Agent	S10
14.	Bayesian Network Model	S11
15.	Neural Network Models	S18
16.	Data Mining	S37
17.	Others	S4, S24, S26, S16, S17, S13

Research Question 2: Existing Size Metrics

Research question two was formed to find the current size estimation metrics used in estimation and through the primary studies five metrics seems to be the prominent metrics in size estimation under agile software development extracted from 18 out of 38 studies (S24, S19, S36, S5, S28, S34, S33, S7, S29, S27, S22, S31, S38, S24, S25, S32, S32, S35) and listed in table 4. However, Functional points and story points are the most used and studied metrics in the standard literature.

Table. 4 Software size metrics

No.	Size Estimation metric	Study
1.	Lines of Code	S24, S19, S36
2.	Story Point	S5, S28, S24, S34, S19, S33, S7, S36
3.	Functional Points and its derivatives	S28, S29, S27, S22, S31, S38, S24, S25, S32, S34, S19, S33, S36
4.	Use Case Points	S28, S24, S32, S19, S35, S36
5.	Object Points	S19, S36

IV. DISCUSSION

This SLR aimed at investigating the current literature for estimation under agile development as listed in the research questions Research Question 1: What are the existing methods used for Effort estimation under agile software development? and Research Question 2: What are the existing metrics for size estimation under agile software development?

V. RESEARCH QUESTIONS

Research Question 1

In this investigation a set of observations where concluded from reviewing the studies, such observation which are related to research question one as follows:

A curious result of table 3 that the most prominent methods for estimation were planning poker, FSM, Expert Judgment, story point, and Historical data (Analogy). All the stated methods there where an expert involved in the process of estimation. Using FSM as the main driver for estimating a software development project is interesting due to FSM is used for software sizing and usually used by management to measure size of the project which assist managers in bidding for projects. However, Implementing FSM requires a technical expertise in the inner of a software components communication.

Furthermore, table 3 shows that there are several studies experimenting with Data Mining, Neural Network, Deep Learning, Ontology Agents, and AI Machine Learning techniques. However, all the techniques are based on analogy because it requires accurate, clean, controlled previous data as a base for all these models. All these techniques are considered algorithmic. Most of the algorithmic techniques are data intensive and the major drawback of these techniques is that large data sets are required while in current industrial scenario the data available is mostly incomplete and inconsistent[20].

Another drawback of these techniques is the preparation for estimation, prepare data, set rules, train the model, evaluate its output usually by expert, only then team can start estimation. Added estimation by these techniques does not add into account team members skills, as these methods uses previous data from previous project by other teams or public access databases. Therefore, results out of these techniques are questionable in relate to its accuracy and eventually its reliability. It is worth for researchers to go further in this maybe by researching a possible set of rules for data to be used for these kinds of techniques also what kind of evaluation method can be adopted to evaluate such methods beside MRE and MMRE, due to REM and MMRE does not consider other factors of estimation such as time consumed, preparation, estimation complexity, previous data, etc, MRE and MMRE depends solely on results comparison with actual effort.

Research Question 2

The most prominent size metrics observed from the literature were Functional points followed by story points, and Use Case Points. Interesting that even some data mining techniques are using functional points as the unit for measurement after the model finishes mining the data either by using keywords or set of rules.
Use case points where effective but with systems that its requirements are somehow detailed and will be developed using object-oriented programming language which is not the case in many systems and the advancement of data mining and data analysis programming languages such as Python and R, which uses functional style of coding in many cases.

A similar sizing method is Object points which depends purely on the objects that will be generated by the system components which requires highly skilled developers in object-oriented programming and design and again that's the not the case in real life situations such as in small companies and medium size once, added experts don't come cheap in the industry. On the other hand, most used method for sizing is Functional points and its many derivatives been used extensively specially in companies where they have team(s) who have gone through several projects and they have experience with company flow, project, team members, and management. Added, FSM requires senior developers to be able to estimate do to FSM based mainly on how the system internally communicate.

There are still some projects uses LOC to measure size but it is mainly on methods that uses AI element in its estimation process such as the once mentioned in table 3 LOC of previous projects in organizations with its User Story can give some level of estimation but highlight of the limitation and inaccuracy of such results were discussed previously in question 1 discussion.

VI. CONCLUSION

The first motivation of this research was to gain an understanding of software effort estimation spectrum specifically under agile environment. Further, is to investigate and document the current state of the art in agile software development effort estimation, that's include but not limited to methods, techniques, limitation, adaptation, and what is the current state of art to handle its limitation.

There are evidences from the results discussed previously that estimation methods need further improvement especially in two main directions. First direction is clear from the adaptation of the light weight methods of effort estimation for the vast changes and movement of the economy, which present a need for more research towards light methods of estimation that can be adapted by a wide range of teams' classes with wide range of experience and not necessary for experts. Second direction is the AI techniques, a lot of research and testing needed in this direction due to many obstacles such as the data complexity, preparation, training of the methods, etc.

Software effort estimation been under research for more than four decades and yet there seem to be not much improvement in the accuracy of estimation. An apparent lack of improvement in estimation accuracy doesn't mean that we don't know more about effort estimation than before but rather it's the complexity of software's and dynamics of the economy to highlight a few that tells us that there are no best estimation model or method and each project is unique [21].

We settle that based on the state-of-the-art evidences that there is no best approach for software development effort estimation because of the huge differences in the estimation accuracy. Development teams prefer lightweight methods for estimation even if other methods provide more accurate estimation. Furthermore, if an estimation model provides a high estimation accuracy may not be the case with a different project in different context. We conclude that the need for tailored estimation methods is needed to suit the fast-changing dynamics of software development without reinventing the wheel by customizing what works to fit the context needed in, and maintain a simplicity in estimation methods so it can be adopted by different teams' members with different range of skills and experience.

The authors are under the impression that the objectives of the research where achieved and a further study will be undertaking to further investigate the literature and analyze the methods discovered and extract the drivers used for estimation by each method which will assist in developing a tailored estimation model for a specific group of software development teams under agile environment.

VII. ACKNOWLEDGMENT

We would like to acknowledge and extend a special thanks to the school of Computing in University Utara Malaysia (UUM) for supporting this work.

REFERENCES

1. S. Ramacharan and K. V. Rao, “Parametric Models for Effort Estimation for Global Software Development,” Lect. Notes Softw. Eng., vol. 1, no. 2, p. 178, 2013.
2. A. Lamersdorf, J. Munch, A. F. V. Torre, C. R. Sanchez, and D. Rombach, “Estimating the Effort Overhead in Global Software Development,” in 2010 5th IEEE International Conference on Global Software Engineering, Princeton, NJ, USA, 2010, pp. 267–276.
3. Z. Chen, T. M. D. Port, and B. Boehm, “Feature Subset Selection Can Improve Software Cost Estimation Accuracy,” p. 6, 2005.
4. A. Kaur, “Estimation of Size in Software Development: A Case Study,” system, vol. 5, no. 7, p. 10, 2016.
5. D. Pfahl, “There exists more than one Agile Method,” p. 12, 2014.
6. M. Fowler and J. Highsmith, “Facilitating change is more effective than attempting to prevent it. Learn to trust in your ability to respond to unpredictable events; it’s more important than trusting in your ability to plan for disaster.,” p. 7, 2001.
7. [M. R. R. Braga, C. I. M. Bezerra, J. M. S. Monteiro, and R. M. C. Andrade, “A pattern language for agile software estimation,” in Proceedings of the 9th Latin-American Conference on Pattern Languages of Programming - SugarLoafPLoP ’12, Natal, Rio Grande do Norte, Brazil, 2012, pp. 1–15.
8. F. Paz, C. Zapata, and J. A. Pow-Sang, “An approach for effort estimation in incremental software development using cosmic function points,” in Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 2014, p. 44.
9. B. Boehm and R. Turner, Balancing Agility and Discipline: A Guide for the Perplexed, Portable Documents. Addison-Wesley Professional, 2003.
10. B. A. Kitchenham, E. Mendes, and G. H. Travassos, “Cross versus within-company cost estimation studies: A systematic review,” IEEE Trans. Softw. Eng., no. 5, pp. 316–329, 2007.
11. D. Azhar, E. Mendes, and P. Riddle, “A systematic review of web resource estimation,” in Proceedings of the 8th International Conference on Predictive Models in Software Engineering, 2012, pp. 49–58.
12. M. Usman, E. Mendes, F. Weidt, and R. Britto, “Effort estimation in agile software development: a systematic literature review,” in Proceedings of the 10th International Conference on Predictive Models in Software Engineering - PROMISE ’14, Turin, Italy, 2014, pp. 82–91.
13. D. Budgen and P. Brereton, “Performing systematic literature reviews in software engineering,” in Proceedings of the 28th international conference on Software engineering, 2006, pp. 1051–1052.
14. M. Petticrew and H. Roberts, “Systematic reviews in the social sciences: a practical guide.,” Malden USA Blackwell Publ. CrossRef Google Sch., 2005.
15. B. Kitchenham and S. Charters, Guidelines for performing Systematic Literature Reviews in Software Engineering. 2007.
16. T. Meline, “Selecting studies for systematic review: Inclusion and exclusion criteria,” Contemp. Issues Commun. Sci. Disord., vol. 33, no. 21–27, 2006.
17. B. A. Kitchenham, E. Mendes, and G. H. Travassos, “Cross versus within-company cost estimation studies: A systematic review,” IEEE Trans. Softw. Eng., no. 5, pp. 316–329, 2007.
18. M. COHN, Agile Estimating and Planning Pearson Education. Inc, 2006.
19. E. Coelho and A. Basu, “Effort estimation in agile software development using story points,” Int. J. Appl. Inf. Syst. IJAIS, vol. 3, no. 7, 2012.
20. M. Vyas, A. Bohra, C. Lamba, and A. Vyas, “A Review on Software Cost and Effort Estimation Techniques for Agile Development Process,” 2018.
21. M. Jorgensen, “What We Do and Don’t Know about Software Development Effort Estimation,” IEEE Softw., vol. 31, no. 2, pp. 37–40, 2014.