Methods and Metrics for Estimating and Planning Agile Software Projects

Completed Research

Edna Dias Canedo
Computer Science Department - University of Brasília - (UnB)
ednacanedo@unb.br

Ruyther Parente da Costa
Computer Science Department - University of Brasília - (UnB)
ruyther@me.com

Abstract

The nature of agile software projects is different from software projects that use traditional methodologies. Therefore, using traditional techniques of estimates of effort, time and cost can produce imprecise estimates. Several estimation techniques have been proposed by several authors and developers in recent years. This work performs a Systematic Literature Review (SLR) of current estimates practices in the development of agile software and the most used size metrics as inputs for these estimates, collecting and documenting them for a future comparison of their accuracy. With the realization of SLR, it was identified that Story Point and Point of Function are the most used metrics in agile projects as the basis for estimating size, time, effort, productivity and cost. Based on these two-size metrics, it was performed a case study with the estimation of effort, time and cost for an agile project of a software factory, where the actual development values were compared with the estimates made to analyze which provided the estimates that most closely approximated the actual values that were estimated.

Keywords

Agile software development, software estimates, software metrics, scrum.

Introduction

Agile software development is a set of iterative and incremental software engineering methods that are advocated on the basis of an “agile philosophy” described in the Agile Manifesto (Beck et al. 2001) and which emerged as an alternative to so-called traditional software development. Agile development emphasizes short development cycles, frequent deliveries, continuous "face-to-face" communication and learning.

Software metrics are often used to understand, control and improve what is done and how it is done in a software development process. Some of the motivations for using metrics are: project planning and estimation, project management and follow-up, quality understanding and business objectives, communication, processes, and improved tools for software development (Pulford et al. 1995). Thus, the measurement is applied in the process of software development or attributes of a product with the objective of improving it continuously. This technique used throughout the software development project assists in estimating, quality control, productivity assessment and project control (Pressman 1995).

When estimating effort, duration, and cost of software development projects, software size is a prerequisite. The functional size of the software to be delivered is a solid basis for estimating a software development project, and one of the most common ways of obtaining the functional size of software is through Function Point Analysis (FPA). Function Point Analysis is a method of estimating the size of a project by considering the input and output elements that are in the project and consolidates each type of operation into data or transaction function (Gamba et al. 2010).

Combining the size of a system, in function points for example, with other metrics, allows the accomplishment of estimates for the development project and definition of a plan of actions focused on meeting the goals (Dias 2003). The success of agile software development and estimates still continue to challenge current projects and organizations. Despite the importance of metrics and estimates for software

development projects, research related to the theme in the context of agile projects still remains scarce, making estimates and planning inefficient and/or imprecise (Kitchenham, 2010).

The main contribution of this article is to identify through the performance of a systematic literature review (SLR) which models and metrics are considered more adequate for agile software development. In addition, carry out a case study in a software factory to compare the accuracy of the two main methods of estimating the most widely used software agile projects according to the results of SLR. The result will contribute to the agile community, more specifically to Scrum, in surveying the metrics used as inputs to make estimates that help in planning agile software development projects.

Research Methodology

This work was developed in two stages, and in the first stage was defined the study process for the Systematic Review of Literature (SLR) based on the work presented by Kitchenham (Kitchenham 2009) and Galster (Galster et al. 2014) comprising the planning, execution and documentation phases of the review. The objective of this stage was to identify scientific works that present methodologies and metrics solutions for agile software development methodologies to identify: 1. common project measurement and control practices; 2. size metrics used; 3. research trends in agile development; and 4. open questions and research topics related to improving the estimates of agile development projects.

The work that was found during the SLR went through the systemic selection made based on previously defined specific criteria. Thus, articles/studies that met all the inclusion and exclusion criteria established became part of the RSL study base. The research protocol adopted in this RSL was composed of the following phases:

- Identification and selection of the works in the databases defined using the search string. The StArt tool (http://lapes.dc.ufscar.br/tools/start_tool) was used as support for the documentation, extraction and structuring of the primary studies.
- Reading the title, abstract and key words of the works applying the inclusion and exclusion criteria defined in the research protocol. Based on these criteria the works were pre-selected.
- Read the introduction and completion of the pre-selected works, re-applying the inclusion and exclusion criteria.
- Complete reading of selected papers and completion of the data collection form.

In the second step, a case study was carried out with the purpose of estimating the size, effort, time and cost of the first release of a project using two different techniques to compare the results with the actual values to identify which estimate is closest of reality.

Systematic Literature Review Results

The Systematic Literature Review (SLR) is a form of secondary study that aims to identify and analyze the relevant research for a given research question (Kitchenham et al. 2007). The systematic review involves three steps:

- **Review planning**: define the need for a systematic review; raise research questions; and define a review protocol: data sources, strategy and search terms, study selection criteria, study quality, data extraction, and data synthesis.
- **Realization of the review**: select and analyze the studies; answering research questions; and present the results, discussions and conclusions.
- **Reporting the review**: to write the review results and format the final document.

Research Questions

In order to characterize the methods for estimating agile software projects, four different research questions (RQ) were formulated, addressing different aspects of this area. The research questions are described as follows:

RQ.1. What are the metrics and methods used to make effort estimates, deadlines, and costs for agile software project planning?

RQ.2. Function point metrics can be used to make estimates of effort, deadlines, and costs for agile software planning? If so, is it the most appropriate estimate?

RQ.3. What metrics are used to verify the productivity of the agile software development team?

RQ.4. What kind of empirical studies in the area of metrics are being done in the agile software development environment by academia, and what results are being obtained?

Search String

The search strategy involved the use of Automatic Search (Silva 2015), which consists of searching through the Search String in the electronic databases. The Automatic Search was performed in the following databases:

- http://dl.acm.org/ - Digital library ACM;
- http://ieeexplore.ieee.org/Xplore/home.jsp - Digital Library IEEE Xplore;
- http://dblp.uni-trier.de/ - DBLP-Computer Science Bibliography;
- https://www.scopus.com/ - Digital Library Scopus.

The search string definition was based on the population, intervention, comparison and the result (Kitchenham et al. 2007). The selected search string for this paper was:

("AGILE SOFTWARE DEVELOPMENT") AND ((METRICS) OR ("FUNCTION POINT") OR (ESTIMATION"OR "PREDICTION)) AND ((EMPIRICAL) OR (VALIDATION) OR (EVALUATION)).

Figure 1 synthesizes the results obtained during data collection. As a result, a total of 291 articles were found. The results were filtered by reading their title, abstract, and keywords, and a total of 189 articles were selected. Another filter was applied according to the inclusion and exclusion criteria defined in the research, through the complete reading of the article, with the object of finding the answers to the research questions. After this, 27 primary studies were classified to data extraction.

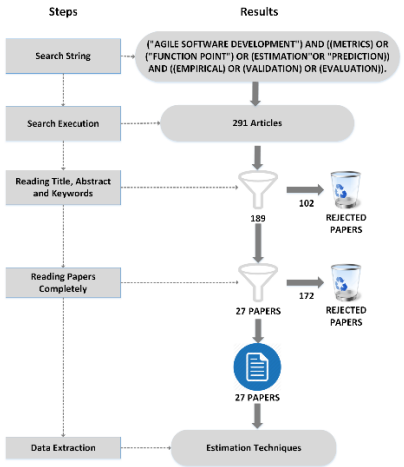

Figure 1. Articles selected Systematic Literature Review

Results

The results of the SLR according to each defined research question were:

RQ.1: What are the metrics and methods used to make effort estimates, deadlines, and costs for agile software project planning?

The most appropriate estimation model for agile projects is Story Point (Kang 2010). A Story Point is a metric used in agile project management and development to determine (or estimate) the effort (difficulty) of implementing a given story. In this context, a story is a particular business need assigned to the software

development team. Thereby, it has been realized that Planning Poker is one of the most popular techniques for agile teams in planning and estimating effort before starting each iteration.

The size estimation techniques are grouped together with the effort estimation techniques, because within the context of agile development, effort estimation is often derived from Velocity estimates and calculations (Kang 2010) and (Usman 2014). The Expert Judgment and Use Case Points technique are also frequently used estimation techniques in the context of agile software development. Table 1 presents the techniques of estimating effort in the agile context and its occurrence in the selected works.

Technique	**Frequency**
Expert Opinion	8
Estimate based on model (COCOMO, etc.)	5
Planning Poker	11
Use-Case Points	3
Function Points	6
Custom Templates	2
Number of Lines of Code	4
Fuzzy based Framework for Estimation	1

Table 1. Occurrence of effort estimation techniques in selected works.

Through the number of Story Points and Velocity of the development team it's possible calculate the term of development of a certain functionality (Kang 2010). For example, if the total number of Story Points for the desired functionalities is 200 and the Velocity of the team is 20, then can be concluded the team will need 20 iterations to complete the development of the respective functionalities. However, the User Story Point (USP) is not objective and can not define a standard practice for estimating the size and complexity of the software (Javdani et al. 2013).

Innovative work has been identified in the area of agile project estimates. The paper (Raslan et al. 2015) for example, proposes a structure that depends on the use of fuzzy logic and aims to help in the production of accurate estimates. In the paper presented by (Nunes et al. 2011) it is proposed to modify the Use-Case-Points (UCP) method to make it suitable for agile software development by naming this new version for interactive UCP version (iUCP).

The work proposed by (Basri et al. 2016) presents a proposed model of effort prediction caused by changes in software requirements. The model integrates the analysis of the impacts of the changes with the COCOMO effort estimation model to improve the precision of the effort estimates from changes in agile software development projects.

The Bayesian Network model was proposed to help agile project managers estimate project effort. It is a graphical model that describes the probabilistic relations between the related variables (Dragicevic et al. 2017).

According to the works analyzed, Story Points is the most used size metric to carry out the estimations of agile projects, however it can also be verified that Function Points are still widely used, often being combined with other metrics.

RQ.2: Function point metrics can be used to make estimates of effort, deadlines, and costs for agile software planning? If so, is it the most appropriate estimate?

A dynamic cost estimating model for agile software projects can be used, namely, it has the ability to adapt during the development process and with changes in requirements (Kang 2010). This model adopts function points such as estimation metric and a tracking algorithm, the project time estimating the size of the functionalities using function points and calculating the cost to derive the duration of the project. To derive the project plan, three procedures are performed. 1. The project team calculates the function points of the desired functionalities; 2. The estimation model, which is composed of the remaining function points,

is generated; 3. The project team develops a plan for the release, iteration, and delivery using estimated cost metrics, such as: people per month and number of lines of code.

Although some papers suggest adaptations of traditional models for estimating and measuring agile software projects, one of the weaknesses of agile communities may still be the failure to estimate and measure projects using standard metrics such as function points that are largely known and used within the industry, but which cover only the functional requirements (Jones 1998), different from Story Points, for example, that do not correspond to a software size and not even the actual effort, but to estimates (not measurement), and that cover the functional and nonfunctional requirements.

Some selected studies also state that Point of Function is not suitable for estimating agile projects because of their granularity and insufficient support for feedback and requirements change. Therefore, they firmly support the idea that companies that work in a traditional or agile way collect traditional measures of size (such as Function Points) for portfolio management, project management and benchmarking; and that companies working according to an agile method also do this, in addition to collecting size measures in an agile-size metric (such as Story Points) for estimation purposes.

RQ.3: What metrics are used to verify the productivity of the agile software development team?

Velocity is often used in agile software communities rather than productivity. Velocity is defined as the number of user stories or number of complete story points in the iteration and is often used to estimate the time remaining until the end of the project (Raslan et al. 2015). However, this measure is very useful for stable projects that are always composed by the same individuals, since any change in team composition will invalidate the Velocity measure. Velocity must be measured for several successive iterations. This is the only way to ensure that it can be useful to make estimates, as presented in (Petersen 2011). Productivity measurement by function point developed is also possible. However, this approach is not always a good alternative because not all development effort is counted as function points, the calibration of the model depends heavily on the skills of the counter and is influenced by subjective judgment (Petersen 2011).

Even with a good method of measurement and/or estimation, there is a big problem in estimating the productivity of a team member in a given task. Some tests have shown that productivity can differ by more than 20 times from person to person. The estimation methods tend to overestimate the time required to complete small tasks but tend to underestimate the time required to complete large tasks (Čelar et al. 2014).

Productivity usually indicates the success of a particular work in relation to the resources used and it is known that there are more than 200 factors that affect the productivity of developers, the main factors that impact productivity: developers' ability and experience, cooperation of the users during elicitation of the requirements and design, schedule or resource constraints, methods used in the project, available tools, choice of an appropriate programming language, complexity of the problem, code complexity, data complexity, project organization structure and the physical work environment (Čelar et al. 2014).

RQ.4: What kind of empirical studies in the area of metrics are being done in the agile software development environment by academia, and what results are being obtained?

It was possible to identify several studies in the area of metrics, being the main:

- The work developed by (Torrecilla et al. 2015) presents a proposal to estimate, plan and manage Web projects by combining some existing agile techniques with Web development principles, presenting them as a unified framework that uses business value to guide resource delivery. The proposal is analyzed by means of a case study, including a real project, to obtain relevant conclusions. The results obtained after the use of the framework in a development project are presented, including interesting results in the planning and estimation of projects, as well as in the productivity of the team throughout the project. It was concluded that the framework can be useful for better managing Web-based projects through a continuous process of management and value-based estimates.
- In (Owais et al. 2016) is proposed a simple algorithmic estimation method for the development of agile software where the authors analyze and compare the results provided by the proposed method through a case study to verify the viability of the algorithmic method in real time. The result shows that the duration and cost of the project increase if the effort increases, but if resources are added increasing the effort, the duration and cost almost remain the same.
- In a survey conducted in 2005, 18 project managers from a Norwegian software development company were interviewed. Data on 52 projects were used to analyze schedule differences and excess effort between

projects using flexible process models (incremental and agile) and sequential process models. Thus, it was concluded that the projects that used a flexible process model had less excess of effort when compared to the projects that used sequential process models (Usman et al. 2015).

- The work developed by (Tanveer et al. 2016) performs a case study whose objective is to explore and understand the estimation process in relation to its accuracy in the context of agile software development from the perspective of the development team. The case study was applied to a German software development company called SAP SE. It was understood that all teams make estimates at each start of a Sprint. Estimates are made for the requirements of a User Story. The teams studied use MS Excel and the JIRA tool for project management. They make the estimates to prioritize the tasks, not for risk assessment, for example. In addition, teams make estimates to develop a common understanding between them as to what should be done in a particular Sprint. Teams A and B make individual estimates, estimating the effort from Worked Hours per Person. Team C performs a game where the effort estimate is made through a consensus, being made using Story Points (Tanveer et al. 2016).
- Three metrics were used: Ideal Day, Story Points and FPA in a case study of the organization. According to the authors, the results showed that Ideal Day and Planning Poker estimated time more accurately than FPA. The authors applied the method in a project with 34 requirements (Gamba et al. 2010).
- The paper (Mukker et al. 2014) presents a guide to optimizing cost and effort in projects using the Scrum methodology (Schwaber, 2002) and the function point with the COCOMO model. The proposal was used in a real case study to make estimates. In addition, the work shows the different tracking and rendering tools used to optimize performance, such as execution diagram, JIRA, SVN, and SCTM. This research concluded that agile methodology is very effective in software development. In addition, they concluded that by combining the two estimation models, one can improve the applied effort and save on costs.

Case Study

The case study was carried out at the company Mirante Technology S.A., which defined a project for application of the research, which for contractual reasons will be called Project Test. The application of the study was carried out in release 1 of project Test, which follows the iterative and incremental lifecycle, based on Scrum (Schwaber 2002). The current design dimensioning is done using the technique of unadjusted function points. Function Point counts follow the rules set out in the IFPUG Version 4.3.1 Counting Practice Manual (CPM), plus the definitions of the Internal Counting Client Guide (most current version) and the SISP Software Metrics Roadmap in SISP version 2.0. The objectives of this case study are:

1) Estimate the size, time, cost and effort of Project Test release 1 based on the two size metrics that according to the SLR of this work are most commonly used in Agile Software Projects: Function Point (FP) and Story Point (SP).
2) Compare the estimates made at the beginning of the project with actual values of the project development to define which of the metrics provides estimates that are closest to the actual values.

Data were collected regarding the backlog, estimated values of SP and FP and the execution of Project Test Sprints, necessary for the accomplishment of the study estimates. At each completed Sprint, Sprint backlog data, SP score and FP User Stories, accepted and declined items, and work hours were stored for later comparison with the estimates made at the beginning of the project and Sprints.

Case Study Results

Once the release backlog has been defined, all user stories have been estimated by the team using Planning Poker. The stories were also measured in Function Points (FP) according to the Handbook of Counting Practices (IFPUG 2010) for the initial estimates of the Project. An estimated count on User Stories function points from the release 1 backlog was performed prior to the Sprints running.

After the estimated release count was performed, the effort was calculated. Based on effort, timing and costs were also estimated. For the estimation of effort with Function Points, the work was based on the simplified model proposed by (Vazquez et al. 2003) which consists of obtaining a productivity index in hours/FP for the specific project in question, and then multiplying the project FP size by the productivity index. The productivity index used was defined by the organization and the team, according to information from similar projects developed in the company.

The value of the release-term estimate in weeks was defined based on the estimated count in function points of the release scope and the production capacity in Sprint function points of the project development team declared by the project executing company. For estimating the total cost of the project using Function Points, the value of the company function point was multiplied by the estimated size of the software.

The estimated total value in function points of the release backlog stories is 387 points. From this it is possible to calculate the average productivity per Sprint of the 13 members of the project team, which was approximately 78 Function Points per Sprint. The desired duration of each Sprint is 2 weeks and the value in Real (BRL) per function point of the company is R$ 469.68. Based on this information, the initial estimates were obtained using Function Points with an uncertainty of ± 25%, as standardized by (Cohn 2005).

The actual values spent on each task were defined by the team and were also obtained for comparison and improvement of the estimates. Comparison of the estimated data with the actual data using function points were: in the third Sprint function points estimated was 80 ± 20 and function points delivered was 114. The real productivity was 90 and function points remaining 135.

Estimates of size, effort, time, and cost using Story Points were made using the data provided by the company related to the release 1 of Project Test. The model proposed by (Torrecilla et al. 2015) was used as the base. The release backlog was estimated with a total of 115 Story Points and the team set the value of 13.64 hours per Story Point. From there, estimates were made of Velocity (23 Story Points), Sprint Days (10 business days), Number of Sprints (5), Number of Hours Available per Sprint (313, 72 ± 78, 43 hours), Amount of Project Hours (1568, 6 ± 392, 15 hours), Sprint Cost (*R*$36.660, 00 ± *R*$9.165, 00), and Total Project Cost (*R*$183.300, 00 ± *R*$45.825, 00). Tables 2 and 3 present the initial estimates of each Sprint, comparing estimated Story Points with actual values and comparing estimated hours with actual hours. Considering an uncertainty of ± 25%.

Sprint	Work Days	Estimated Velocity	Actual Velocity	Story Points Delivered	Story Points Remaining
0	10	23 ± 5, 75	15	15	100
1	10	25 ± 6, 8	18	21	79
2	10	26, 33 ± 7, 3	21,66	29	50
3	10	25 ± 7, 3	23,25	28	22
4	10	22 ± 7, 75	24,8	31	0

Table 2. Comparing Estimated Data with Actual Data Using Story Points (SP)

Estimated Hours	Actual Hours	SP Hours
341 ± 85, 25	726	48,4
871, 2 ± 217, 8	801	38,14
1029, 78 ± 257, 44	798	27,51
660, 41 ± 165, 1	837	29,89
927, 67 ± 231, 9	556,2	17,94

Table 3. Comparison between Estimated Hours and Actual Hours

The purpose of this case study is to answer the following research question: which of the metrics used as input to estimate time, cost and effort provide estimates that approximate the real values in an agile development project: Story Points?

In order to respond, it was performed separately estimates of effort in hours, time and cost in an agile development project of a software factory based on the size metrics that according to the SLR performed in this work are the most used in projects Software Agile: Function Point and Story Point. The estimation process and the results were presented in Tables 2 and 3. To determine which of the metrics is most

accurate, was chosen take the actual values of the project and compare them with the estimates carried out. Tables 4 and 5 present the project results in relation to the values of the initial work plan. The associated uncertainty is ± 25%.

Metric	Actual Value	Estimated Value	Variation	Deviation
Average Productivity	81,63	78 ± 19,5	3,63	4,65%
Total Hours Worked	3728,5	3900 ± 975	171,5	-4,39%
Total Function Points	417	387 ± 96,75	30	7,75%
Hours per Function Point	8,9	10 ± 2,5	1,1	-11%
Work Days	50	50	0	0%
Total Project Cost	R$ 195.856,60	R$ 183. 175,20 ± R$ 45.793,80	R$ 12.681,36	6,92%

Table 4. Project Results vs. Initial Estimates Using Function Points

Metric	Actual Value	Estimated Value	Variation	Deviation
Velocity Medium	20,54	23 ± 5, 75	2,49	-10,82%
Total Hours Worked	3728,5	1568, 6 ± 392, 15	2159,9	+137,69%
Total Story Points	124	115 ± 28, 75	9	+7,82%
Hours per Story Points	29,98	13, 54 ± 3, 41	16,44	+126,26%
Work Days	50	50	0	0

Table 5. Project Results vs. Initial Estimates Using Story Points

Analyzing the values of Tables 4 and 5, it is possible to verify that the estimates made from the Function Point approximated more than the real values of the project. The difference between the actual values of the Average Productivity of the Time, Total Hours of Work, Hours per Function Point, and Total Project Cost metrics, and the estimates had values within the range of uncertainty defined as acceptable in the project. The team already used the Function Point size metric to estimate their development projects, so they have more experience with the method. Thus, the initial productivity value and the hour value of 1 FP were closer to the real and made the other estimates more accurate as well.

Unlike the Function Point, the team rarely used Story Point metrics to make estimates in their projects. This explains the great difference between the estimated hours for the development of 1 SP and the amount actually spent. The team underestimated the development time and with that we had values of hours well away from the estimate. However, it is possible to verify that from Sprint 3 the estimates began to have values closer to the actual estimate. This difference can also be explained by the subjectivity of the metric, since the technique does not allow us to use data from other agile development teams, only from the projects developed by the team itself.

Conclusion

Through the execution of this work, it was possible to perceive the relevance of size, effort, cost and time estimates in the context of agile software development, and therefore, methods and estimation metrics have been increasingly discussed in scholarly works that seek the best and most precise metrics used in a given agile context.

Through the Systematic Literature Review (SLR) it was possible to identify the methods and the main size metrics used in estimations in the context of agile software development. Among the most used techniques are: Planning Poker, Expert Opinion and Function Point Analysis. The most used metrics for estimates are Story Points and Function Points.

The primary studies identified in the SLR showed that the methods and the metrics for estimates are mostly applied to a given context of agile development with adaptations in order to fit the project in question. Thus, it can also be concluded that the estimation metrics must always be adapted to fit the project context, since each project will have its own characteristics which influence the result of the estimates.

The case study showed that the estimates using the Function Points metric had deviation percentages of the actual values from the estimated values lower than the estimates made using the Story Points metric. This is due to the fact that the team has a lot of experience with the use of estimates with the Function Points metric, so the initial values of the productivity estimates and the hour value of 1 FP were much closer to the actual value, which consequently made the rest of the estimates more precise.

However, it is necessary to perform other experiments using real projects of different characteristics to be able to affirm with certainty that the Function Points metric is more accurate than the Story Points metric in agile development projects.

REFERENCES

Basri, S., Kama, N., Haneem, F., & Ismail, S. A. 2016. Predicting effort for requirement changes during software development. In Proceedings of the Seventh Symposium on Information and Communication Technology (pp. 380-387). ACM.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Kern, J. (2001). Manifesto for agile software development.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., & Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. Journal of systems and software, 80(4), 571-583.

Buglione, L., & Abran, A. (2013). Improving the user story agile technique using the invest criteria. In Software Measurement and the 2013 Eighth International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2013 Joint Conference of the 23rd International Workshop on (pp. 49-53). IEEE.

Čelar, S., Turić, M., & Vicković, L. (2014). Method for personal capability assessment in agile teams using personal points. In Telecommunications Forum Telfor (TELFOR), 2014 22nd (pp. 1134-1137). IEEE.

Cohn, M. (2005). Agile estimating and planning. Pearson Education.

Dias, R. (2003). Análise por pontos de função: uma técnica para dimensionamento de sistemas de informação. Revista Eletrônica de Sistemas de Informação ISSN 1677-3071 doi: 10.21529/RESI, 2(2).

Dragicevic, S., Celar, S., & Turic, M. (2017). Bayesian network model for task effort estimation in agile software development. Journal of Systems and Software, 127, 109-119.

Farid, W. M., & Mitropoulos, F. J. (2013). NORPLAN: Non-functional Requirements Planning for agile processes. In Southeastcon, 2013 Proceedings of IEEE (pp. 1-8). IEEE.

Galster, M., Weyns, D., Tofan, D., Michalik, B., & Avgeriou, P. (2014). Variability in software systems—a systematic literature review. IEEE Transactions on Software Engineering, 40(3), 282-306.

Gamba, M. L., & Barbosa, A. C. G. (2010) Applying Software Metrics with Scrum. Anais SULCOMP, 5.

IFPUG (2010). Manual de Práticas de Contagem de Pontos de Função, Versão 4.3.1.

Javdani, T., et al. (2013). On the current measurement practices in agile software development. arXiv preprint arXiv:1301.5964.

Jones, T. C. (1998). Estimating software costs. McGraw-Hill, Inc.

Kang, S., Choi, O., & Baik, J. (2010, August). Model-based dynamic cost estimation and tracking method for agile software development. In Computer and Information Science (ICIS), 2010 IEEE/ACIS 9th International Conference on (pp. 743-748). IEEE.

Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering–a systematic literature review. Information and software technology, 51(1), 7-15.

Kitchenham, B. (2010). What's up with software metrics? A preliminary mapping study. Journal of systems and software, 83(1), 37-51.

Mahnič, V., & Hovelja, T. (2012). On using planning poker for estimating user stories. Journal of Systems and Software, 85(9), 2086-2095.

Matthies, C., Kowark, T., Uflacker, M., & Plattner, H. (2016). Agile metrics for a university software engineering course. In Frontiers in Education Conference (FIE), 2016 IEEE (pp. 1-5). IEEE.

Misra, S., & Omorodion, M. (2011). Survey on agile metrics and their inter-relationship with other traditional development metrics. ACM SIGSOFT Software Engineering Notes, 36(6), 1-3.

Mukker, A. R., Mishra, A. K., & Singh, L. (2014). Enhancing Quality in Scrum Software Projects. International Journal of Science and Research (IJSR), 3(4), 682-688.

Nunes, N., Constantine, L., & Kazman, R. (2011). IUCP: Estimating interactive-software project size with enhanced use-case points. IEEE software, 28(4), 64-73.

Olague, H. M., Etzkorn, L. H., Gholston, S., & Quattlebaum, S. (2007). Empirical validation of three software metrics suites to predict fault-proneness of object-oriented classes developed using highly iterative or agile software development processes. IEEE Transactions on software Engineering, 33(6), 402-419.

Owais, M., & Ramakishore, R. (2016). Effort, duration and cost estimation in agile software development. In Contemporary Computing (IC3), 2016 Ninth International Conference on (pp. 1-5). IEEE.

Parvez, A. W. M. M. (2013). Efficiency factor and risk factor-based user case point test effort estimation model compatible with agile software development. In Information Technology and Electrical Engineering (ICITEE), 2013 International Conference on (pp. 113-118). IEEE.

Petersen, K. (2011). Measuring and predicting software productivity: A systematic map and review. Information and Software Technology, 53(4), 317-343.

Popli, R., & Chauhan, N. (2014). Cost and effort estimation in agile software development. In Optimization, Reliabilty, and Information Technology (ICROIT), 2014 International Conference on (pp. 57-61). IEEE.

Pressman, R. S. (1995). Engenharia de software (Vol. 6). São Paulo: Makron books.

Pulford, K., Kuntzmann-Combelles, A., & Shirlaw, S. (1995). A quantitative approach to software management: the AMI handbook. Addison-Wesley Longman Publishing Co., Inc.

Raith, F., Richter, I., Lindermeier, R., & Klinker, G. (2013). Identification of inaccurate effort estimates in agile software development. In Software Engineering Conference (APSEC), 2013 20th Asia-Pacific (Vol. 2, pp. 67-72). IEEE.

Raslan, A. T., Darwish, N. R., & Hefny, H. A. (2015). Towards a fuzzy based framework for effort estimation in agile software development. International Journal of Computer Science and Information Security, 13(1), 37.

Schofield, J., Armentrout, A., & Trujillo, R. (2013). Function points, use case points, story points: Observations from a case study. CrossTalk, 26(3), 23-27.

Schwaber, K., & Beedle, M. (2002). Agile software development with Scrum (Vol. 1). Upper Saddle River: Prentice Hall.

Schweighofer, T., Kline, A., Pavlic, L., & Hericko, M. (2016). How is Effort Estimated in Agile Software Development Projects? In SQAMIA (pp. 73-80).

Silva, F. S., Soares, F. S. F., Peres, A. L., de Azevedo, I. M., Vasconcelos, A. P. L., Kamei, F. K., & de Lemos Meira, S. R. 2015. Using CMMI together with agile software development: A systematic review. Information and Software Technology, 58, 20-43.

SISP (2016). Roteiro de Métricas de Software do SISP. Ministério do Planejamento, Desenvolvimento e Gestão.

Tamrakar, R., & Jørgensen, M. (2012). Does the use of Fibonacci numbers in Planning Poker affect effort estimates?

Tanveer, B., Guzmán, L., & Engel, U. M. (2016). Understanding and Improving Effort Estimation in Agile Software Development—An Industrial Case Study. In Software and System Processes (ICSSP), 2016 IEEE/ACM International Conference on (pp. 41-50). IEEE.

Torrecilla-Salinas, C. J., et al. (2015). Estimating, planning and managing Agile Web development projects under a value-based perspective. Information and Software Technology, 61, 124-144.

Usman, M., Mendes, E., & Börstler, J. (2015). Effort estimation in agile software development: a survey on the state of the practice. In Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering (p. 12). ACM.

Usman, M., Mendes, E., Weidt, F., & Britto, R. (2014, September). Effort estimation in agile software development: a systematic literature review. In Proceedings of the 10th International Conference on Predictive Models in Software Engineering (pp. 82-91). ACM.

Vazquez, C. E., Simões, G. S., & Albert, R. M. (2003). Análise de Pontos de Função: medição, estimativas e gerenciamento de projetos de software. Editora Érica, São Paulo, 3.