Real-time Estimation Approach to Scrum Projects Effort: Task Point and Individual Reliability

Yu Guo, Jun Guo*, Mohan Shi, Aixuan Dong
School of Software College
Northeastern University
Shenyang, China

Abstract—**In software project management, it is a useful method to control the progress of the project through effort estimation. How to estimate the effort in real time in the Agile software development has always been a difficulty faced by Agile teams. Aiming at the current mainstream Agile Software Development techniques, Scrum framework, this paper puts forward the calculation approach for reliability of Scrum team members and the Sprint development time estimation approach based on individual reliability, i.e., the total working time estimation of the next Sprint is realized by estimating and adjusting the working time of the team individuals. 51 development teams with 5-9 people have put this method into practice for experimental verification, and the results show that the highest estimation accuracy of this approach can reach 84.1%.**

Keywords- Scrum; individual reliability; task point; effort estimation

I. Introduction

Agile software development is a software development method with strong adaptability and flexibility. There are many Agile development methods (e.g., Extreme Programming (XP), Agile Unified Process (AUP), Scrum), favored by enterprises. Scrum is one of the representative approaches [1]. In software project development, making a comprehensive development plan is the foundation of project success [2]. In order to achieve this goal, the team needs to estimate the effort of the project. Software project effort estimation refers to with the analysis of business objectives, considering current environmental conditions, human resources and other factors, the team estimates the time required for the completion of different stages of the project or the time required for the completion of the whole project [3]. Basing the estimation result, the team members can make a schedule for the completion time of their tasks, which is conducive to supervise the team members' work and effectively manage the software project.

It is one of the representative approaches of traditional software project effort estimation to estimate through function points [4]. This approach was first proposed by Albrecht, and IFPUG appeared later, developing Albrecht's idea. Function point is a function divided according to user requirements in software project. IFPUG realizes the effort estimation with the support of the quantified result of the input, output and file type of all function points and the adjustments of the estimation adjustment factor [4] [5]. With the further studies of researchers, approaches such as COSMIC were proposed to modify the deficiencies of IFPUG [6] [7] [8]. Although the traditional effort estimation approaches have gradually been more and more mature, it is hard for them to adopt to the unpredictability and huge changeability of requirements nowadays [9].

As time goes by, software development framework has gradually improved to deal with these problems. Agile software development framework represented by Scrum is the mainstream currently. Native Agile effort estimation methods focus on the thoughts of each team member, which is highlighted in the estimation methods such as Story Points, Poker Planning, Fibonacci Sequence and T-shirt Size. Many studies on Agile estimation researched the characteristics and shortcomings of these methods. In addition, some researches put forward new methods to solve these problems. In [10][11], machine learning was used for estimation. What's more, the authors in [10] compared machine learning, expert opinions and mathematical algorithm models, finally concluding that machine learning is more accurate in Agile estimation; By analyzing the text of user story description, Sudarmaningtyas and Mohamed [10] got 83% accuracy of story point estimation. In [12], the effort of Scrum-based IT projects was estimated with the usage of fuzzy logic model. Arora et al. [13] introduced the adaptive neuro-fuzzy inference system (ANFIS) and the novel Energy-Efficient BAT (EEBAT) technique to effort estimation in Scrum, by comparing state-of-the-art meta-heuristic and machine learning with ANFIS-EEBAT approach, concluding that ANFIS-EEBAT performs best. The one by Ramessur and Nagowah [14] used machine learning technique, considering 12 factors that have an influence on effort estimation, to predict a Sprint work time. Butt et al. [15] provided the knowledge which factors may lead to low accuracy of effort estimation in Scrum environment and the framework to avoid the bad impact. Authors in [16] mainly studied good and bad practices in the Agile software development projects, contributing to finding the key to correct estimation. And Almeida et al. [17] holds the opinion that the metrics related to the business value delivered and the success of sprint goals are the most relevant. They also investigate the correlation of other metrics to Scrum process monitoring.

Traditional methods estimate the effort of software development in the stage of requirements analysis, which has great limitations in the changeable and volatile software development environment. Thus, the accuracy of estimation is very poor [18]. Furthermore, in the existing researches on Agile estimation methods, some put forward metrics that have an impact on the accuracy of estimation, while do not give a specific improvement scheme [16][17]; and some use theories

DOI reference number: 10.18293/SEKE23-001

or models such as machine learning, natural language processing, etc. [10][11][12][13][14] for effort estimation, which do not embody the characteristics of Agile in the models to a certain extent.

Therefore, we propose a novel concept of task point for Scrum-based IT projects, based on the concept of the story points estimation method. Task point measures the effort of individual in the team, and then the effort estimation of the whole team can be obtained according to individual task points. What's more, for the sake of improving the accuracy of estimation, the reliability of individual is considered to adjust the estimation result. After each Sprint, the reliability of individual will be adjusted and applied to the next effort estimate. The advantage of this approach is that it not only refines the estimation granularity of Scrum, but also takes full account of the reliability of individual estimation, which can help gain the effort estimation more accurately in real time in every Sprint, thus making the approach high availability.

II. Ideal Time and Task Point

In Scrum, each user story is closely related to the requirements in the project backlog. Project development serves user stories, so estimating the ease of development in Scrum usually requires estimating the ease of completing each user story. In that case, Story Point [19] is put forward for the purpose. We don't use specific units, such as man-months, to express the effort, but use the relative concept of story point to estimate the difficulty of user stories. It is commonly believed that team decisions are more significant in Scrum, so the story point of each user story is obtained by integrating the opinions of team members. That's the reason why ideal time and task point are proposed. Task point can be seen as the evolution of story point. In practice the effort measured by time is easier to understand. Therefore, the ideal time is proposed as the basic unit of task point size, and the task point is the individual's estimation of completing a task.

A. Ideal Time

Ideal time is the time to complete the work excluding other activities unrelated to the work. Its attributes are as follows:

1) Uniqueness: The user story task being estimated is the only work that the people needs to do.

2) Self-containment: When the task starts executing, everything needed is ready, independent of other tasks.

3) Independence: There is no interference in the process of doing it.

4) Generality: The efficiency of the person performing the task is normal.

B. Scrum Task Point

Task point is the amount of time a task takes, corresponding to the effort of the task in Scrum. During development, the basic unit of task point is ideal time.

The team's development consists of multiple Sprint processes in Scrum. In a Sprint cycle, the team will solve 1 ~ n user stories, and each story will be broken down into 1 ~ n tasks to be completed by different developers. After the Product Owner makes a backlog and each team member chooses tasks, the development work begins. Before a Sprint starts, team members need to estimate the ideal time required to complete every task, then the time recorded as task point.

By calculating the task points of all tasks contained in a user story, it is easy to determine the time required for the development of the user story. And the time of all user stories in this Sprint is the development time of this sprint.

III. Real-time Effort Estimation Approach Based on Scrum Team Member Reliability

In most of the methods based on experts to estimate effort, the experts' knowledge and development experience play an important role in estimating and controlling the progress of different periods of the project. However, the reality is that experts may ignore the overall level of the team or other factors when estimating, which makes the estimation result meaningless [20]. In the method described in this paper, we pay more attention to the estimation data of team members to complete tasks and their estimation accuracy. It is obvious that the estimation results based on the real level of team will be more reasonable.

A. Calculation Method for Scrum Team Members' Reliability

Before Scrum-Based IT project development, the team will estimate the story points of user stories. The common practice is to take the development difficulty of a certain user story as the baseline. Then the story points of other user stories will be determined relative to the baseline. Inspired by it, we propose a brand-new calculation method to estimate the time when the user story will be completed. In this method, the ideal time for members to complete tasks and the accuracy of individual estimation are the basis of calculation. After members pick up tasks, they estimate the completion time of each task. Assuming that the ideal time for a task to be completed is *x*. After the member completes the task, the actual time taken is *y*. Afterwards, the reliability of this task *R* is

$$R = \frac{y}{x} \tag{1}$$

When the *n*th task is finished in this sprint, the reliability of this task will be compared with the maximum and minimum values of the 1st ~ *n*-1 task reliability of this member. If it is greater than the maximum value, the maximum value will be changed to this value. While if it is smaller than the minimum value, the minimum value will be updated.

B. Development Duration Estimation for Sprints Based on Individual Reliability

Having the definition of individual reliability and its calculation method, we can make effort estimation of a Sprint. During the *i*th round of Sprint, each member is assigned to n tasks. Before the software development begins, a task completion recording table for each one will be established. Table I shows how to record members' task completion.

TABLE I. MEMBER TASK COMPLETION RECORDING TABLE

Tasks	**Expected Working Time**	**Actual Working Time**
Task 1	a_1	a_1'
Task 2	a_2	a_2'
…	…	…
Task *n*	a_n	a_n'

The member primarily estimates the expected completion time of each task. For example, for task *i* the expected working time is a_i. After it is completed, the actual working time would be recorded as a_i'. With a_i and a_i', we can get this reliability of task *i* is R_i. After that we compare the reliability maximum R_{max} and minimum R_{min} of this member with R_i respectively to update the reliability range.

After the team has completed all the development of the *i*th Sprint, we can determine the reliability range of each member in this Sprint. Then we compare the individual reliability range of ith Sprint r_i with the reliability of i-1th Sprint r_{i-1}, if $r_i \subseteq r_{i-1}$, the reliability of this person will be updated to r_i. Otherwise, we should compare the reliability boundary values of the two Sprints respectively, i.e. the minimum boundary value of the latest reliability is the smaller value of the minimum boundary values between r_{i-1} and r_i, and the maximum boundary value of the latest reliability is the larger value of the maximum boundary values between r_{i-1} and r_i. After updating the individual reliability, it will be used to estimate the time of the next Sprint. At the beginning of the *i* +1th Sprint, members still give their own estimation time for each task, and then we can use their latest individual reliability to adjust and get their own range of task completion time. If a member's reliability range is $[R_{min}, R_{max}]$, and his estimation time for task m is a_m. Then the completion time of task m will be estimated as (2):

$$T_m = \begin{cases} t_{\max} = R_{\max} \times a_m \\ t_{\min} = R_{\min} \times a_m \end{cases} \quad (2)$$

With the continuous adjustment of reliability, the estimated completion time of each task to be completed is constantly changes, and the predicted completion time of the *i*+1 Sprint also gets persistently adjusted. Furthermore, the effort estimation can be visually displayed in the Scrum Kanban, thus making the team development process more transparent. Considering the result of this approach is a time range, it also gives the team a buffer in the development, which is beneficial for the team to allocate the work more reasonably in real time and improve the development efficiency.

IV. EXPERIMENTAL VERIFICATION

A. *Experimental Process*

In order to validate the methodology presented, we assigned 51 small student teams in Northeastern University to develop the same Scrum-Based IT project. Before the project development, the product backlog and the story point of each user story has been determined. This project is divided into three Sprints. In each Sprint, All the teams record the estimated work time and the actual work time of each task, calculating the reliability of each person in the development project. We continue to track the development process of each team and investigated the changes of the accuracy of team effort estimation in each Sprint.

The change of everyone's reliability in a 6-people team in three rounds of Sprint is shown in Table Ⅱ:

TABLE II. MEMBER RELIABILITY CHANGE

Members	**Reliability Range**		
	Sprint1	***Sprint2***	***Sprint3***
Member1	1-1.0669	0.98-1.0669	1-1.0551
Member2	0.98-1.0412	0.98-1.0322	1-1.0321
Member3	1-1.0333	1-1.0356	1-1.0235
Member4	1.0111-1.0612	1.0111-1.0532	1.001-1.0321
Member5	1.1231-1.222	1.1201-1.222	1.1101-1.222
Member6	1-1.0333	1-1.0301	1-1.0210

As you can see, after several rounds of Sprints, teammates gain a deeper understanding of their ability to complete tasks, thus making the estimation accuracy improved to some extent, which shows that it is effective to adjust the estimation through reliability. On the one hand, the time range of estimation is narrowed to obtain more accurate estimation results; On the other hand, it can urge members to complete positively tasks according to their own set time.

Fig. 1 shows the estimation accuracy of the team in each Sprint:

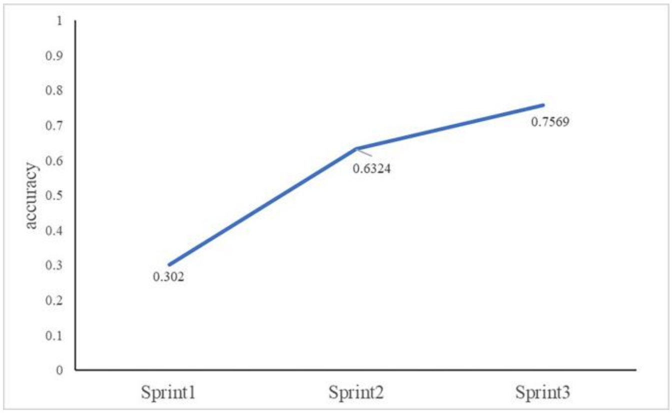

Figure 1. Changes in estimation accuracy.

The first Sprint initially establishes reliability, and the next two Sprints continuously update the personal reliability. Apparently, with the increase of the number of iterations, the estimated time adjusted by reliability gradually approaches the actual time.

B. *Result Analysis*

After three iterations, we collated and summarized the data of the 51 teams, then drawing a figure based on the data, which more intuitively demonstrates that as the number of iterations increased, the estimation accuracy also increased. In the figure, one dot represents the prediction result of a team, and different colors represent different Sprints. We take the center of the first estimation time range of each team in a Sprint as a reference, calculating the prediction accuracy k of the team according to (3).

$$k = \frac{d}{l} \tag{3}$$

Where d represents the distance from the actual time to the center, and l is the length of the predicted time range. When k is in the range of -0.5 to 0.5, we consider the estimation to be accurate. The estimation accuracy of each Sprint is the proportion of the number of teams accurately estimated. Fig. 2 shows the estimation results of different teams.

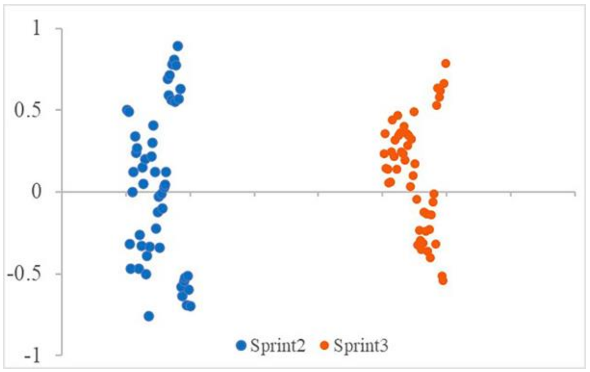

Figure 2. Sprint2 and Sprint3 prediction accuracy of each team.

The final experiment result shows that the estimation accuracy can reach about 60% in the second Sprint and about 84% in the third Sprint. Meanwhile, the estimation error of the third Sprint is reduced compared with the second Sprint.

V. CONCLUSION

This paper mainly explores how to estimate each Sprint time through the team individual estimation in Scrum. Firstly we summarize the shortcomings of existing methods. Then we put forward the concepts of task point and ideal time and the method of applying them to Agile estimation, finally verifying the feasibility and accuracy of the method.

The advantage of the method proposed in our study is to estimation the effort of the entire team from the individual, which is an approach to estimation the effort from the actual level of individuals at a smaller granularity. This method takes into account that the estimation accuracy of individual usually will change with the progress of the project, so the reliability is proposed to modify the results , improving the estimation reliability. It implements Agile ideas and is in line with Scrum development very well. The future work will focus on how to achieve more accurate prediction for the whole project at the early stage of development and get rid of the estimation limit of current approach.

REFERENCES

[1] Butt, Shariq Aziz, Abbas Khalid, and Arshad Ali. "A software development for medical with a multiple decision taking functionalities." Advances in Engineering Software 174 (2022): 103294.

[2] Govil, Nikhil, and Ashish Sharma. "Estimation of cost and development effort in Scrum-based software projects considering dimensional success factors." Advances in Engineering Software 172 (2022): 103209.

[3] Agilealliance. [Online].Available: https://www.agilealliance.org/agile101/the-agile-manifesto/

[4] Cuadrado-Gallego, Juan J., et al. "Early functional size estimation with IFPUG unit modified." 2010 IEEE/ACIS 9th International Conference on Computer and Information Science. IEEE, 2010.

[5] Edagawa, T. A. Y. S. S. T. T., et al. "Function point measurement from Web application source code based on screen transitions and database accesses." Journal of Systems and Software 84.6 (2011): 976-984.

[6] Cuadrado-Gallego, Juan J., et al. "An experimental study on the conversion between IFPUG and COSMIC functional size measurement units." Information and Software Technology 52.3 (2010): 347-357.

[7] Rabbi, Md Forhad, Shailendra Natraj, and Olorisade Babatunde Kazeem. "Evaluation of convertibility issues between IFPUG and COSMIC function points." 2009 Fourth International Conference on Software Engineering Advances. IEEE, 2009.

[8] Cuadrado-Gallego, Juan J., Fernando Machado-Piriz, and Javier Aroba-Páez. "On the conversion between IFPUG and COSMIC software functional size units: A theoretical and empirical study." Journal of Systems and Software 81.5 (2008): 661-672.

[9] Alshammari, Fahad H. "Cost estimate in scrum project with the decision-based effort estimation technique." Soft Computing 26.20 (2022): 10993-11005.

[10] Sudarmaningtyas, Pantjawati, and Rozlina Mohamed. "A Review Article on Software Effort Estimation in Agile Methodology." Pertanika Journal of Science & Technology 29.2 (2021).

[11] Ratke, Cláudio, et al. "Effort estimation using bayesian networks for agile development." 2019 2nd International Conference on Computer Applications & Information Security (ICCAIS). IEEE, 2019.

[12] Saini, Abhishek, Laxmi Ahuja, and Sunil Kumar Khatri. "Effort estimation of agile development using fuzzy logic." 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO). IEEE, 2018.

[13] Arora, Mohit, et al. "An efficient ANFIS-EEBAT approach to estimate effort of Scrum projects." Scientific Reports 12.1 (2022): 7974.

[14] Ramessur, Melvina Autar, and Soulakshmee Devi Nagowah. "A predictive model to estimate effort in a sprint using machine learning techniques." International Journal of Information Technology 13.3 (2021): 1101-1110.

[15] Butt, Shariq Aziz, et al. "A software-based cost estimation technique in scrum using a developer's expertise." Advances in Engineering Software 171 (2022): 103159.

[16] Łabędzki, Maciej, et al. "Agile effort estimation in software development projects-case study." The Central European Review of Economics and Management (CEREM) 1.3 (2017): 135-152.

[17] Almeida, Fernando, and Pedro Carneiro. "Performance metrics in scrum software engineering companies." International Journal of Agile Systems and Management 14.2 (2021): 205-223.

[18] Lenarduzzi, Valentina, and Davide Taibi. "Can Functional Size Measures Improve Effort Estimation in SCRUM?." ICSEA-International Conference on Software Engineering and Advances, Nice, France. 2014.

[19] Pasuksmit, Jirat, Patanamon Thongtanunam, and Shanika Karunasekera. "Story Points Changes in Agile Iterative Development."

[20] Usman, Muhammad, et al. "Developing and using checklists to improve software effort estimation: A multi-case study." Journal of Systems and Software 146 (2018): 286-309.