Improving Agile Planning for Reliable Software Delivery

Jirat Pasuksmit, Fan Jiang, Kemp Thornton, Arik Friedman, Natalija Fuksmane, Isabelle Kohout, Julian Connor
{jpasuksmit,fjiang2,kthornton,afriedman,nfuksmane,ikohout,jconnor}@atlassian.com
Atlassian
Sydney, Australia

***Abstract*—Agile software development prioritizes the delivery of working software. However, there are challenges in the sprint planning process that could impact the reliability of sprint delivery. In this paper, we list three challenges related to the sprint planning process. We also discuss future work directions to facilitate the sprint planning process and mitigate those challenges for software teams.**

I. Introduction

To reduce time to market, an Agile team continuously delivers software increments in a short time-boxed iteration (i.e., sprint, as in the Scrum methodology) [4, 12, 13]. However, it has always been challenging to create a reliable plan and develop the software according to the plan. To identify challenges in the sprint planning process, we conducted a literature survey and collected insights from semi-structured interviews with practitioners from over 50 companies that use Jira Software. During the interview, we asked for general information about their teams, the tools being used, and how they monitor and improve their software development effectiveness. In this paper, we summarized the challenges in the sprint planning process with future research directions to help teams create a more reliable sprint plan.

II. Challenges

This chapter summarizes three main challenges to be mitigated in order to facilitate the sprint planning activities.

Prolonged Sprint Planning Process. During sprint planning, a software team selects a set of issues (i.e., fine-grained work items to develop the software) to work on during the sprint [4, 12, 13]. The selected issues typically (1) align with the sprint goal, and (2) are ready to be worked on (i.e., small, detailed, and their accumulated size fits the sprint capacity) [12]. However, it is a tedious and time-consuming task for the team to determine which issues need to be prioritized and which of them could fit the team's capacity.

To mitigate this challenge, future work may focus on prioritizing the product backlog based on the alignment with the sprint goal and readiness of each issue. Moreover, future work may also identify the dependencies among the issues. Such abilities will minimize the time taken in sprint planning, especially in reviewing the product backlog and selecting the next best issues for the sprint.

Lack of Actionable Insights to Mitigate Uncertainty. A delay of an issue in a sprint may block all dependent issues or even the sprint delivery. We identified several prior studies conducted in this area, e.g., predicting delays [2, 3, 8], predicting inaccurate estimations [5], and visualizing the uncertainty [6, 7]. However, there is a lack of actionable insights for individual issues for which a high level of uncertainty was identified.

To provide an actionable suggestion, future work may provide localized suggestions based on the factors that contribute to the uncertainty related to a specific issue. In addition, future work should properly handle data leakage due to misalignment between the time that the collected data pertains to and the prediction time. In other words, predictions and insights should be driven by the information available at the sprint planning time (e.g., [10, 11]) or apply sliding or expanding window validation techniques (e.g., [1, 8]), rather than relying on information that became available at a later time, with the benefit of hindsight. Such an approach would mimic real-world usage scenarios, where information is not always complete. Moreover, we suggest that researchers should consider the complexity of code changes associated with each issue by linking Jira Software to the code repositories used by the team. Providing a localized recommendation while considering those suggestions would allow the team to progressively mitigate and better prepare for the uncertainty surrounding an issue during sprint planning.

Unforeseen Bottlenecks. An unexpectedly high number of tasks of a specific type may cause bottlenecks during the sprint. For example, receiving too many requests for code review from other teams or team members, unexpected back-and-forth code reviews, or even a lack of computing resources for code deployment. These phenomena may cause an unexpected workload for a team or for individual developers, increasing the risk of sprint delivery delays.

To mitigate this challenge, future work may focus on automatically identifying, classifying, or visualizing bottlenecks in each area (e.g., [9, 14]). One possible approach could be to capture and analyze events beyond Jira Software by integrating it with other systems (e.g., Bitbucket, Synk). Such contributions could take the form of predictions of whether a bottleneck will occur in the current or the next sprint, along with actionable recommendations to mitigate the anticipated bottlenecks. Such abilities will help the team avoid the impact of bottlenecks, leading to reliable sprint delivery.

REFERENCES

[1] SM Didar Al Alam, Dietmar Pfahl, and Guenther Ruhe. 2016. Release Readiness Classification – An Explorative Case Study. In *ESEM*. 1–7.

[2] Morakot Choetkiertikul, Hoa Khanh Dam, Truyen Tran, and Aditya Ghose. 2015. Predicting Delays in Software Projects Using Networked Classification. In *ASE*. 353–364.

[3] Morakot Choetkiertikul, Hoa Khanh Dam, Truyen Tran, and Aditya Ghose. 2017. Predicting the delay of issues with due dates in software projects. *EMSE* 22, 3 (2017), 1223–1263.

[4] Mike Cohn. 2006. *Agile estimating and planning*. Pearson Education.

[5] Marco Conoscenti, Veronika Besner, Antonio Vetrò, and Daniel Méndez Fernández. 2019. Combining data analytics and developers feedback for identifying reasons of inaccurate estimations in agile software development. *JSS* 156 (2019), 126–135.

[6] Simon Grapenthin, Matthias Book, Thomas Richter, and Volker Gruhn. 2016. Supporting Feature Estimation with Risk and Effort Annotations. In *Proc. of the SEAA*. 17–24.

[7] Jo Erskine Hannay, Hans Christian Benestad, and Kjetil Strand. 2019. Agile Uncertainty Assessment for Benefit Points and Story Points. *IEEE Software* 36, 4 (2019), 50–62.

[8] Elvan Kula, Arie van Deursen, and Georgios Gousios. 2021. Modeling Team Dynamics for the Characterization and Prediction of Delays in User Stories. In *Proc. of the ASE*.

[9] Rita Marques, Miguel Mira da Silva, and Diogo R Ferreira. 2018. Assessing Agile Software Development Processes with Process Mining: A Case Study. In *Proc. of the CBI*, Vol. 1. 109–118.

[10] Jirat Pasuksmit, Patanamon Thongtanunam, and Shanika Karunasekera. 2022. Story Points Changes in Agile Iterative Development: An Empirical Study and a Prediction Approach. *EMSE* 27, 6 (2022), 156.

[11] Jirat Pasuksmit, Patanamon Thongtanunam, and Shanika Karunasekera. 2022. Towards Reliable Agile Iterative Planning via Predicting Documentation Changes of Work Items. In *2MSR*. 35–47.

[12] Kenneth S Rubin. 2012. *Essential Scrum: A Practical Guide to the Most Popular Agile Process*.

[13] Ken Schwaber and Jeff Sutherland. 2020. *The Scrum Guide, The Definitive guide to Scrum: The Rules of the Game*.

[14] Miroslaw Staron and Wilhelm Meding. 2011. Monitoring Bottlenecks in Agile and Lean Software Development Projects – A Method and its Industrial Use. In *Proc. of the PROFES*. 3–16.