Master Thesis Proposal: Risk detection and risk control through sprint backlog analysis

Sytse Groenwold
13083228
sytse.groenwold@student.uva.nl

Zhiming Zhao
Supervisor
z.zhao@uva.nl

TBD
Examiner
email

ABSTRACT

DISCLAIMER:

Find the most recent version of my design through GitHub: https://github.com/SytseGroenwold/masters-thesis. I would immensely appreciate it if you can take the lastest version from there when providing feedback; I plan to finalise it by Sunday 27th of February, but I understand if you cannot wait until then.

I have only secured this project 3 weeks ago and only last week the supervisor and I settled on the subject. Hence my paper being in such a raw shape. It speaks for itself which sections are more fleshed out and which are mere summations of points I wish to talk about. I would greatly appreciate it if you can take this into account while reviewing it.

KEYWORDS

Agile, Scrum, Risk, User Stories, Sprint Backlog, Machine Learning

ACM Reference Format:

Sytse Groenwold, Zhiming Zhao, and TBD. 2022. Master Thesis Proposal: Risk detection and risk control through sprint backlog analysis. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 3 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

In software development, there are several risk events that can decrease product quality, delay delivery or even cause the project to fail[2]. Tangible examples of these events are budget overruns and losing personnel, while more subtle ones include a lack of clearly defining work to be done or team members being unaligned. (Early) detection (i.e. risk assessment) of such risks and controlling them effectively (i.e. risk control) is crucial for the success of a project[3].

The Agile development methodologies have demonstrated their advantages over traditional methods in reducing time to market, overall product quality and a closer alignment to business needs[5]. In Scrum, this is achieved through an iterative process over a short period of time, called sprints. The product manager creates a backlog of user stories, which are the business requirements translated to work for the development team. During a sprint, the team works on a selected number of these user stories and afterwards they reflect on both the work done and the collaboration within the team. Despite these improvements, Agile methodologies can still suffer from risks such as unclear user stories, inefficient decomposition, and scheduling of the sprint backlogs. Determining those risk through analysis of the user stories in the product backlog is mostly done by human experts. In a previous study, we sought to automate this process, by creating an machine learning model based on previous works and ensembling those models.

While those results show promise for a machine learning model to detect risks, improvement is needed to make the predictions more robust, especially to achieve more consistent scores between different datasets. We determined that incorporating the sprint plannings, instead of only analysing the individual user stories, could be a valuable addition to the model parameters in achieving this goal. In this study we plan to do that, and additionally investigate a way to single-out the specific risks that are inside a sprint backlog, so that teams can exercise risk control before they affect the team output.

1.1 Research question

To what extend can inclusion of sprint backlog analysis improve risk assessment within software teams using Scrum and how can this risk assessment lead lead to and improvement of risk control?

1.2 Sub questions

These are the sub questions

2 EXPLORATORY DATA ANALYSIS

Among the different software packages used by Scrum adaptors to track user stories, Jira is by far the most used one, having a large margin over other popular services such as Service-Now, Trello and Azure DevOps. For this study, due to widescale availability, only data sets extracted from Jira will initially be used. If it turns out that this is insufficient, other data sets can be explored.

The requirements for the data sets are straight-forward: they are collections of user stories that are gathered in both product backlogs and sprint backlogs. Additionally, each user story needs to contain the description the story, the date it was created, the date it is completed, and in which sprints backlogs it has been included. The sprint backlogs are not even required to be individually gathered: if each user story has a record of the sprints it was part of, it should be sufficient information for the model.

The data sets that will be considered for this study are publicly available data sets, which mostly come from github accounts of other researchers whom have gathered them for previous studies.

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

When these data sets prove to not be sufficient, there will be two options to gather a data set from organistations (Utrecht University's internal software development department or ING Bank's data sat of user stories stored in Service Now). There are previous studies who have already worked with these data sets, but as they are not publicly available and are constantly growing, they will only at at later moment be gathered if there is a necessity for them. Lastly, this project's supervisor is currently giving a course on DevOps and Cloud-based Software, in which different teams of students will need to work together using a user story backlog. At the end of this course (March), the students are to hand in an overview of their product/sprint backlogs as well and these can be considered for this study.

The model for the data sets gathered from Jira have the following features. In a later stage of the project, features which are not used can be left out.

3 RELATED WORK

Some introductory piece of text to ensure that the next paragraphs nicely indent the same amount of space.

3.1 Agile software development

Software development is known to face issues with delivering expected quality and overrun schedules and costs.[3]. Agile methodologies attempt to combat these issues through shorter delivery windows by iterative development, focusing on interaction and communication between all people involved (team members, business, stakeholders and end-users), and reducing the size of the intermediate results. This leads to teams being more capable to react to changing requirements and prevent waste of time and resources.[1]

The most widespread used Agile methodology is Scrum[2]. Following this methodology, teams work on *user stories*, which are the system requirements. These user stories are created by the *product owner* and based on the requirements as defined by the business, stakeholders, and the end-users. Together with the team, the product owner can *refine* the stories and assign it *story points*, which are arbitrary numbers that signal the amount of effort (not time) it takes to complete. The user stories are prioritized and stored in the *product backlog* until selected. The team works in cycles of work called *sprints* and are usually 2-4 weeks long. At the beginning of a sprint, the teams selects user stories from the product backlog to work on, which they intend to finish within that sprint. User stories can therefor never take longer than the duration of sprint; if they do, they need to be split up into smaller ones. User stories inside the sprint backlog do not change during the cycle. Every day there is a *standup*, where each team member shares their progress on the user stories and adjust if necessary to complete the work. At the end of the sprint, which is *timeboxed*, the team holds the *review* where the outcomes are discussed and during the *retrospective* they gather feedback to use during the next cycle. If some work is not done, it automatically rolls over to the next sprint, although this must be avoided.

Despite these benefits, a study in 2014 showed that many adaptors of the Scrum methodology (in Norway) still address theirs risks in their old, traditional approaches.[5] Additionally, even if some problems are solved through adaptation of Scrum, new risks can possibly be introduced, such as risks in the sprint planning, risks in the code base and technologies used, an increased separation between the development and the operational work, and increased impact of technical debt[4][6].

Table 1: Overview of features in the Jira user story data sets.

#	Feature	Description
1	Issue id	Issue is Jira's name for a user story
2	Summary	
3	Description	
4	Status	
5	Resolution	
6	Project key	
7	Priority	
8	Rank	
9	Impact	
10	Votes	
11	Story point estimate	
12	Change reason	
13	Change risk	
14	Change type	
15	Request type	
16	Epic name	
17	Environment	sbx, dev, tst, acc, prd
18	Security level	
19	Assignee ID	
20	Reporter ID	
21	Creator ID	
22	Watchers ID	
23	Created	
24	Updated	
25	Last Viewed	
26	Resolved	
27	Due date	
28	Actual end	
29	Actual start	
30	Original estimate	
31	Remaining estimate	
32	Time Spent	
33	Work Ratio	
34	Sprint#	A number of fields equal to the number of sprints it has been part of.

3.2 Risks assessment from sprint backlogs

In our previous study, we defined what we understand ask risk: ... We also established that previous work on the subject has done, but mostly on the individual aspects that together make up the broader issue of risk. We investigated and implemented an ensembling machine learning (ML) method that can assess a certain amount of risk inside an individual user story. One outcome of that study is

increasing the so called attributes of our model. So far, the *created*, *updated* and *resolutiondate* fields of the Jira story schema are not used or of low significance of the model. The same goes for the use of individual sprint backlogs, instead of only individual user stories.

End

3.3 Risk control

Most studies done in the past

4 METHODOLOGY

4.1 Machine learning models for risk assessment

Bla bla

4.2 Methods to share risk control

Besides having a model that can assess the risk, it also needs to share which specific elements are causing the risk. These need to be reported to the team.

The team should be able to adjust the user stories added to a sprint and immediately receive feedback on the risk score of the current sprint planning.

Stretch ambition might be to automatically suggest which user stories from the top of the product backlog should be added to a sprint. For this to become a reality, team velocity based on story points should be present, possibly from other models.

A platform to offer such a service could be through Atlassian Marketplace, where various Jira Apps are offered to be used inside the planning software.

5 RISK ASSESSMENT

A few potential risks can be identified regarding the research as described in this thesis design. In this section, these risks will be discussed, and if needed, a back-up plan will be formulated.

While finding data sets including product backlogs with user stories is relatively simple, it will only be useful if it includes sprint data. This can be resolved by using data sets that have not yet been processed to remove this information.

The found data sets from different planning software might require different types of processing. Fortunately it is already labeled, so it should comprise mostly of cleaning the data sets. Another benefit is the availability of good scripting skills and available budget to outsource it.

Besides different models in which the data resides, entries in the data sets can widely vary between organisations and even between teams of the same organisation. This might introduce a risk of the model only being able to properly train on its own data set. This could possibly be turned around into an advantage, because the moment this occurs and is resolved, it should mean that the data model is applicable to new data sets as well.

For this study, I am dependent on previous study to supply the ML model. In the worst case scenario, I should be able to fall back to more basic models found in literature.

There is always the risk of the ML model accuracy not being significant. If this is the case, the main focus of the project should switch from risk assessment to risk control or vise versa.

6 PROJECT PLANNING

Figure 1 below shows an overview of the week-by-week planning for the thesis project. Yellow bars are the deadlines weeks set by UvA as feedback moments. Calendar week 4 is Monday 14 of February, week 26 is Monday 27 of June.

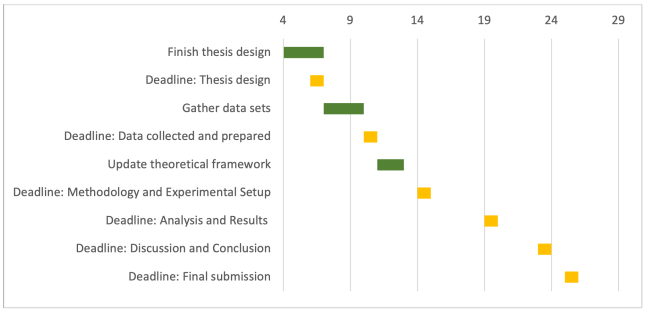

Figure 1: Gantt chart on planning per calendar week.

REFERENCES

[1] David Cohen, Mikael Lindvall, and Patricia Costa. 2004. An introduction to agile methods. *Adv. Comput.* 62, 03 (2004), 1–66.

[2] Pete Deemer, Gabrielle Benefield, Craig Larman, and Bas Vodde. 2010. The scrum primer. *Scrum Primer is an in-depth introduction to the theory and practice of Scrum, albeit primarily from a software development perspective, available at: http://assets. scrumtraininginstitute. com/downloads/1/scrumprimer121. pdf* 1285931497 (2010), 15.

[3] Torgeir Dingsøyr and Yvan Petit. 2021. Managing layers of risk: Uncertainty in large development programs combining agile software development and traditional project management. *arXiv preprint arXiv:2103.09034* (2021).

[4] Philippe Kruchten, Robert L Nord, and Ipek Ozkaya. 2012. Technical debt: From metaphor to theory and practice. *Ieee software* 29, 6 (2012), 18–21.

[5] Lubna Siddique and Bassam A Hussein. 2014. Practical insight about risk management process in agile software projects in Norway. In *2014 IEEE International Technology Management Conference*. IEEE, 1–4.

[6] Wojciech Walczak and Dorota Kuchta. 2013. Risks characteristic to Agile project management methodologies and responses to them. *Operations Research and Decisions* 23 (2013).