Scrums effects on software maintainability and usability

Gustav Ernberg
guser350@student.liu.se

January 19, 2015

Synposis

I have been working as a web developer with advanced web applications on a number of different projects. In all projects I have been working with an agile approach except for in one project where there was an experienced project manager who enforced a waterfall-method (that project did end up wasting a lot of developer hours for functionality that the customer in the end did not want). I am writing this paper in order to find out how maintainability and usability are affected by using Scrum.

This paper is intended for software project managers who want to know more about how Scrum effects maintainability and usability, two software quality aspects I value highly. After reading this report you will have a better understanding how Scrum may have an impact on usability and maintainability and practices using Scrum that has a positive impact on usability and maintainability.

The main point in this paper is that Scrum in itself allows for high end-user interaction which in general improves usability, however one needs to pay attention to the user stories and the definition of done to make sure that all developers make sure that usability is an important software quality factor. Maintainability appears to be improved by using Scrum but the metrics used does not reveal which parts of the Scrum methodology are involved.

The major sources in this article are scientific articles, books and thesis works.

1 Introduction

Today there are many different software development methods availiable, many companies are changing from waterfall methods to agile methods. Among the agile methods, Scrum is one of the most popular one.

Compared to the waterfall model Scrum employs less thourough planning for future features and requirements, there is also much shorter development cycles and frequent end-user interaction. These differences are likely to have an impact on the maintainability and usability in a software development project.

This report will present you results from case-studies and reports in order to describe how maintainability and usability is effected by using Scrum as compared to a waterfall method.

2 Scrum

Scrum is an agile software development method. Some of the major concepts of Scrum are: user-stories, product and sprint backlog, short development cycles (called sprints, less than 30 days), delivery-ready after each sprint, short daily "start up" meetings. [1] [2, 9-9]

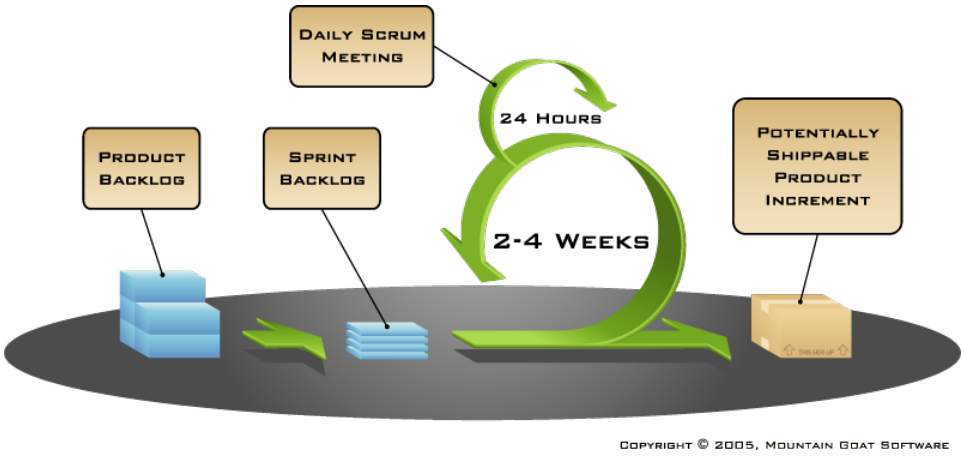

Figure 1: Work-flow using scrum. *(CC BY 2.5, Mountain Goat Software)*

2.1 User story

An user story is a description of a desired functionality. The user story also contains information about for who the functionality is, why it is valuable, en estimate of how much work the story requires and a acceptance criteria (in order to know when it is implemented). [1] An example of a user story could be:

> "As a website visitor I want to be able to be able to buy a flight-ticket so that I can travel."

2.2 Backlog

There are two types of backlogs in a Scrum project; one product backlog which consists of all user stories which are not yet done and one sprint backlog which consists of user stories that are to be done in the current sprint. Both backlog's user stories are prioritized so that the user stories most important to the customer are done first. [1]

2.3 Sprint

A sprint is a short period of time 2-4 weeks in which a number of user stories are being worked on. Every day in a sprint starts with a daily scrum meeting which is a short meeting where everybody in the team meets and describes: what they did yesterday, what they plan to do today and any problems that they see right now. [1]

Any member in a Scrum team is allowed to work on any user story and there is no restrictions on what user stories a member may pick other than those that are currently being worked on. A user story has at least three states (more may be agreed upon by the team), the three basic states are: Todo, doing, done. [1]

In Scrum there is a "Definition of Done" that the team agress upon, either per user story or for the project as a whole. Whenever a user story is done, it mean that it is ready to be used in the software. [1]

3 Maintainability and Scrum

Maintainability is how easy a software system is to modify. [3]

Maintainability is a non-functional requirement and hard to measure, however it has been established that the following factors affect the maintainability: complexity, defects, developer experience and skills, software

age, size and system understanding. [4] Using Scrum maintainability improving tasks is usually what H. Kniberg would consider a tech heavy user story or a "tech story". H. Kniberg has found through his own experience that stories that these tech stories needs to be defined as user stories with a better focus on the bigger picture or otherwise they risk de-prioritization, which has a negative effect on the maintainability of the software. His recommendations are that tech stories are to be transformed to user stories, integrated into existing user stories or as a last resort define them as a tech story and keep a separate list of them so that they can be prioritized correctly together with the product owner. [5]

D. Knippers establish in his thesis that the factors: software age, developer experience and skills are not affected by using an agile software development method. He discovers that in agile development methods has a reduced complexity and fewer defects which has a positive impact on maintainability. He also discovers that changing the design has a negative impact on maintainability and that this is a cruical part of all agile methods. It needs to be addressed that his thesis has a Extreme Programming (XP) focus, and eventhough XP and Scrum has some similarities it is important to know this as perhaps some practices of XP (for example TDD) is likely to have an impact on maintainability. [4]

In a master thesis by D. Wallenstorfer he evaluates how different metrics affect maintainability. The subject of the study is an software application that is being developed by a company that switches from waterfall to Scrum. In his thesis he finds a clear correlation between how these metrics improves by using Scrum. The results in the study are interesting, but since the study only evaluates one single company there needs to be more work done in order to draw any conclusions from this. [6]

4 Usability and Scrum

Usability is a measurement of the ease of use and efficiency of the software. [3]

Usability is a non-functional requirement affected by a range of different factors making it hard to specify when and how to focus on usability. In order to address this M. Singh has found that some teams has usability engineers who work ahead of the current sprint providing developers with wire-frames during the sprint, some teams has usability acceptance tests (definition of done) for user stories and some organization just let the software developers incorporate usability on their own. M. Singh has discovered that teams in general using Scrum does not have an usability focus but in-

stead is focusing on getting user stories done as quickly as possible, resulting in that the first user interface design that comes to mind is likely to be used regardless of it's usability. [7]

M. Singh also states that some developers responsible for user interfaces somtimes lack experience and knowledge to develop user-centric solutions. The reason for this is probably that usability is not included as a part of Scrum but instead is up to the Product Owner to demand and prioritize this. M. Singh argues that Scrum is too focused on task completion than on enhancing user experience. [7]

5 Discussion

One key component to improve usability and maintainability seems to be that we need to address these quality factors in our user stories. [8]

Since Scrum relies on constant change, it is not far fetched to think that developers adapt their programming style to this, making the software easier to change which has a positive effect on maintainability. The thesis by D. Wallerstorfer seems to support this but his study is not conclusive and is not Scrum specific since other factors were changed aswell.

One problem when researching how Scrum affects maintainability and usability is that Scrum is often combined with other methods making it hard to determine if it is Scrum or some other practice that has affected the different qualities. This problem is not fixed easily since a team switching from waterfall to Scrum is likely to change other practies at the same time to become more agile. Scrum does not promote any particular software engineering practices, however due to the fact that the enviroment is constantly changing and that user stories can be performed by any team member it is preferable that there are automatic testing at all times. In order to ensure this, TDD is generally a good practice to keep the tests up to date with the codebase enabling others to make refactoring and code alterations with greater confidence.

My personal experience after developing with TDD is that it has a positive impact on maintainability - however there are not enough studies made on this subject to say anything certain yet. [9]

When I started writing this paper I expected to find many positive reports on usability when using Scrum but I have not found any conclusive reports backed up by real metrics supporting this. There are many claims about that Scrums short sprints and frequent deliveries with high end-user interaction improves usabilitiy and it seems likely, however the paper from

M. Singh clearly states that usability is something that needs additional work with Scrum. The fact that M. Singh proposes a U-Scrum methodology is a clear sign that there are many ways that we can improve usability even when usign Scrum. [7]

I think that one way to make sure that usability is not lost in the process when using Scrum is to use established guidelines about usability in the definition of done, so that no user story is "release ready" until the usability aspects of the implementation are evaluated. I also think a good addition to any teams definition of done could be in the form of a checklist derived from the following guidelines by Shneiderman and Plaisant [10]:

- Strive for consistency
- Cater to universal usability
- Offer informative feedback
- Design task flows to yield closure
- Prevent errors
- Permit easy reversal of actions
- Make users feel they are in control
- Minimize short-term memory load

[10]

It would be interesting to perform a large study based on software developed by companies that have switched from Waterfall to different agile approaches in order to find out if there are any conclusions to be drawn between the different agile development methods and their respective impact on a project. Only evaluating Scrum is hard due to the fact that it is often combined with a change in other practices aswell.

6 Conclusion

Maintainability and usability appears to be improved by using Scrum if one designs user stories and/or has a "definition of done" that includes these quality factors. However more work is still needed to draw any certain general conclusions. [11] [6] [8] [4]

References

[1] C. Sims and H. L. Johnson, *Scrum: a breathtakingly brief and agile introduction.* [Foster City, Calif.] : Dymaxicon, c2012, 2012.

[2] P. Bourque and R. E. D. Fairley, Eds., *SWEBOK Version 3.0.* IEEE, 2014.

[3] *24765-2010: Systems and software engineering — Vocabulary.* IEEE, 2010.

[4] D. Knippers, "Agile software development and maintainability," BSc thesis, Universiteit Twente, 2011.

[5] H. Kniberg, *Scrum and XP from the Trenches - How we do Scrum.* C4Media, 2007.

[6] D. Wallerstorfer, "Improving maintainability with scrum," Master's thesis, Technischen Universität Wien, 2011.

[7] M. Singh, "U-scrum: An agile methodology for promoting usability," in *Agile 2008 Conference.*

[8] R. Davies, "Non-functional requirements: Do user stories really help?" *Methods & Tools*, 2010.

[9] M. Siniaalto and P. Abrahamsson, "A comparative case study on the impact of test-driven development on program design and test coverage," in *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*, Sept 2007, pp. 275–284.

[10] . P. C. Shneiderman B., *Designing the user interface: Strategies for effective human–computer interaction (5th ed.).* Addison-Wesley, 2009.

[11] M. Agarwal and R. Majumdar, "Article: Software maintainability and usability in agile environment," *International Journal of Computer Applications*, vol. 68, no. 4, pp. 30–36, April 2013.