Advantages of SCRUM in Open Source Software Development

Kan Valerii, Tri Do Cao, Folawunmi Oke
Abstract

1. Introduction

Research questions
Research method

2. Background theory

2.1 Agile software development
2.2 SCRUM

3. Scrum in open source software development

3.1 Difficulties of using SCRUM in a distributed OSSD environment
3.2 Evident case of applying SCRUM to OSSD
3.3 How SCRUM shaping the way people practice OSSD

4. Conclusions

References

Abstract

Open source software development (OSSD) is one of the main forces for driving the software industry forward. There are many literature discussing about the usage of OSSD in developing new software as well as its licences. However, there is still a gap in previous researches about particular methodologies impact on assisting open source software. This article will try to explore the usage of SCRUM, one of the most common methods of agile approach, in OSSD. Through literature review, we hope to shed a light on the advantages of SCRUM in OSSD as well as how to implement the theoretical part in real life practice. SCRUM and OSSD are similar that seem to be a perfect match to be used in any OSS project and being so they are applicable in achieving the same goal of fast and self-sustained development.

1. Introduction

Global software market is rapidly becoming extremely competitive. With the urge of fast development and long term maintenance, many software projects in multiple fields, both commercial and non-commercial ones, have decided to follow open source practice to utilize the strength of help from the vast developer community. Even though open source software development (OSSD) has been considered to be one of the most important phenomenon in term of thriving software development forward, there are no particular set of steps, processes on how to develop software in open source manner. Therefore, we feel the need of conducting this article to provide a better understanding of open source software development processes in practice. Since there is not any formulated process particularly for open source software project until now, the article will mainly focus on how SCRUM affects the OSSD processes.

Since SCRUM naturally requires extensive collocated collaboration, while OSSD is mostly distributed (with some of the developers might not even meet each other once), it is clear that
there are obstacles in applying SCRUM for OSSD. However, through literature review and our own empirical study, it is proven that SCRUM turns out to be one of the best methods to use in OSSD. With that to be said, this article will try to confirm the possibility of using SCRUM in OSSD then go on to emphasize how SCRUM actually enhance the practice of OSSD.

**Research questions**

On our way of researching the topic, we came across the article “Apply SCRUM in an OSS Development Process: An Empirical Evaluation” by Luigi Lavazza, Sandro Morasca, Davide Taibi, and Davide Tosi, in which they also discuss the about SCRUM in OSSD and state whether or not SCRUM can be used in distributed OSSD. We feel that their research question matches with our direction, therefore, we decide, for the first research question, to use somewhat similar research question with them in order to analyze their work and provide more of our thoughts on the topic.

- Can SCRUM be used in distributed OSSD?

However, we do not want to focus only on answering yes and no questions but also provide some more specific idea about how SCRUM actually enhance OSSD practice. In order to achieve that purpose, we come up with our second research question.

- How SCRUM shaping the way people practice OSSD?

With the two research questions, we want to give a brief look about SCRUM in OSSD and hope to bring some lights in future practice for OSSD practitioners.

**Research method**

In order to answer the questions above, we performed a search for articles using Google Scholar and gathered relevant literature which will guide our research. We found out some interesting articles about agile approach and SCRUM as well as OSSD. Moreover, we also go through a lot of articles have both SCRUM and OSSD in discussion. We read each of the literature by ourselves and try to define key answers for the research questions that stated above.

**2. Background theory**

**2.1 Agile software development**

The most common problems in software development are time and budget limits exceeding. Mostly the reason for that is incorrect project organization that requires optimization. Agile is a set of software development methodologies that speeds up development stage and minimizes
risk levels. There are six main agile methodologies: Extreme Programming, SCRUM, Crystal Clear, Agile Modeling, Adaptive Software Development and Feature Driven Development. In the article we describe the usability of SCRUM in OSSD, because it is the most popular method, which is used in more than a half of all agile development projects. (Cho 2008)

The term agile was identified on February 2001 in “The Manifesto for Agile Software Development” in Utah, USA. The agile ideas have been used before, but formulation the ideas at the separate document were determined as a starting point for agile development. It became an alternative software development model to “Waterfall model” that had been considered as a standard. For users it was tricky to manage “Waterfall” requirements because they did not always had clear image of own preferences. For that point agile methods flexibility was a good alternative to development process. (Cho 2008)

Flexibility is a qualificative word in agile software development. According to manifesto ideas clients and users satisfaction level is the main point of the project. The working process pays attention to interactions between people. Agile contracts do not set rigorous limitations and can be changed any time in order to support customer interests. The whole action in not only following instructions and documenting detailed, the aim is to create properly functioning product. (Cho 2008)

Risks minimization is a significant advantage of agile over Waterfall model. Any agile project gives to a customer an opportunity to get deep understanding of own preferences during the whole process. The project development is divided to five short time frames, usually from two weeks to one month length, that are called iterations: prototyping, requirements capture, analysis and design, implementation and stabilization. A client can observe the development process, adjust functionalities, test and run the product or even stop the project any time. Non-agile methods does not provide these consumer conveniences because of strict plans and stage orders that can not be modified at all. (Cohn & Ford 2003)

On the other side, the high flexibility level of agile methods can be considered as a main negative issue. The majority of agile project tends to omit product development plan creation. Flexible approach is oriented to work with small local product development and is not applicable to global projects. The customer’s possibility to make changes any time he/she wants results in regular new demands, often contradictory to current project architecture. The aspiration to satisfy every customer’s desire often causes non-effective overworking. Moreover, agile methods are oriented to solve problems in the fastest and simplest way, but ignoring code specification. As a result even small code change in future can lead to program defects. This approach results in low quality of produced software. (Cho 2008)

Agile development and Open source have several common ideas. High user satisfaction priority, continuous development and redesign followed with user interaction and regular product releases are the objectives of the both methodologies. Both agile and open source development do not follow define project plan that can not be modified. Final product design is often the result
of continuous and evolving understanding of customer’s or developer’s preferences. The product should be modified several times before the final release, every time improving for increasing user convenience, avoiding mistakes that can now be exactly evaluated before the development process starts. (Gandomani, et al., 2013)

2.2 SCRUM

SCRUM is an agile software development process developed by Ken Schwaber and Jeff Sutherland in the 1990s as a new option for the traditional software development life cycle and is nowadays often used when dealing with challenging projects. It is one of the most popular and widely used methodology of agile software development and it allows organisations to achieve development targets on time and with acceptable benefits. SCRUM is made up of a certain number of individuals acting as a team who work together to achieve organisation goals at the required time frame and less cost. (Beedle, et al. 1999)

SCRUM does not have the normal top to down approach but adapts based on what the project status presents. It also has a different approach from other agile methodologies in its organisation of work in that the main aim is to produce as much quality project output as the team can during the specified time frame. The team involved takes on as much project as they can and set a goal to complete as much project as possible within a specified period, usually one month. The specified period is termed as sprints. (Beedle, et al. 1999)

In SCRUM, individuals in the team has different roles such as the product owner, SCRUM master and SCRUM team. The product owner serves as the customer representative and is responsible for creating product requirements, maintaining the product backlog, accepting or rejecting project results, etc. The SCRUM master has some roles which include being the leader of the team, handling of the documentation needed, assisting the team achieve project goals, prevents team members from outside distractions, planning sprint meetings and daily SCRUM. The SCRUM team could be made up of maximum ten members and are tasked with executing project goals like developing the products, accountable for the success or failure of the project. The SCRUM team consists of individuals of various specialization and skills. (Mahalakshmi & Sundararajan 2013)

The customer requirements produced by the product owner at the beginning of the SCRUM development project is called product backlog while the leftover customer requirement which was left undone from the previous sprint and moved to the next sprint is called sprint backlog. The sprint backlog is managed by the team. It is the role of the product owner to accept or reject results at the end of each sprint or decide which unfinished project will progress or which will not be continued. SCRUM also involves meetings between those involved in the projects. They have meetings such as sprint planning meeting, daily SCRUM and sprint review meeting. A spring planning meeting takes place between the product owner, SCRUM master and SCRUM team and during this meeting, they make decisions on what to do and how to carry out the
tasks. The daily SCRUM meeting is attended by the SCRUM master and the SCRUM team. This usually take about 15 minutes and involves a short discussion about the progress of their tasks. It is usually done standing. The sprint review meeting occurs after every sprint and the product owner gives feedback afterwards. From this meeting, the sprint planning meeting is also revised by the product owner. (Mahalakshmi & Sundararajan 2013)

SCRUM is also applicable to distributed sites when working together as a team on a software development project. It is not limited to local teams alone and will equally have the desired effect when implemented geographically. Sutherland et al. writes that the best practices for distributed SCRUM include daily SCRUM team meetings between the developers in all locations, daily meetings of Product Owner team, hourly computerised builds from a main repository, the developers at all locations and on the same team should be equally responsible and unified, and continuous incorporation of XP practices like pair programming with SCRUM (Sutherland, et al., 2007). A practical well-known distributed team model that is applicable comprises of isolated SCRUMs, distributed SCRUM of SCRUMs and totally integrated SCRUM. Isolated SCRUM describes teams that are in different geographical locations. Distributed SCRUM of SCRUMs describes remote SCRUM teams located in different geographical areas but work together in a common SCRUM process and have meetings frequently. Totally integrated SCRUMs involve different SCRUM teams who have members in different geographical locations. Distributed SCRUM is in use by both small cross-site projects and large cross-site projects alike. (Sutherland, et al., 2007)

3. Scrum in open source software development
3.1 Difficulties of using SCRUM in a distributed OSSD environment

To discuss about the difficulties of using SCRUM in OSSD environment, we have to mention firstly that OSSD is distributed by its nature (Gandomani et al., 2013). In OSSD, the reason why software has to be open source because, by that way, all of the developers can have the possibility to contribute in developing process (Feller, J. and Fitzgerald, B., 2000). With that to be said, the developers can be anybody from anywhere or we can say that they are mostly distributed. Even though there are projects that a specific organization is formed to develop a particular open source software, it’s still clear that the team members are mostly from different places without a single brick and mortar office.

As discussed above, while OSSD is distributed by its nature, SCRUM, on the other hand, requires extensive collaboration, cross-team communication and multi-channel informal information exchanges, etc (Cho J., 2008). In some extreme cases, SCRUM practitioners are even requested to follow the exact procedure in which the team must have daily time-box face-to-face meeting and weekly sprint review. Studies have proven that the more collocation practices, the more successful SCRUM projects can be (Cho J., 2008). This shows the fact that SCRUM and OSSD contradict each others in the original philosophy. Therefore, choosing SCRUM to be the method for OSSD is not as natural as other type of software development.
Moreover, it raises the practical difficulties whether communication in Open SCRUM project can be effective or not.

Since communication is the most important factor for the success of SCRUM, it’s more challenging in OSSD about how to close the distance between project members in order to achieve the target. Among many way of communication, informal channel is the main gate to achieve SCRUM philosophy in practice. With same-site development, there can be a lot of knowledge hub such as coffee corner, play room, etc where project members can share information in an informal way without restriction. That also help to build team spirit and trust among members. In OSSD, such things are not applicable, most of the communication is going through online communication and in some cases, it can mainly be asynchronous channels. Therefore, it might cause issues for OSSD project where the difference between team members are too much to overcome.

While less effective communication can be an obstacle for SCRUM in OSSD, it can also lead to delay of changes when problems arise in the middle of the development. With the use of asynchronous communication channels and key contributors from different time zone, a single change can take up to a really long time before getting solved. One example is the bug life cycle of Linux kernel where Linus Torvald is the only one that can accept the final submission which can take for months. Delay is a common phenomenon in software development, however it seems to be more serious in OSSD where the contributors are distributed in so many different way than normal same-site or even multi-site cases.

Knowledge management is another factor that can hinder the use of SCRUM in OSSD. In SCRUM, documentation is suggested to be minimized in order to achieve faster development (Llanos, J.W.C. and Castillo, S.T.A., 2012.). However, since OSSD has no face-to-face communication, a well written documentation is crucial to share common knowledge between contributors. This, again, shows that SCRUM and OSSD have contradicts in original philosophy. Therefore, some clarification about how SCRUM and OSSD can be matched with each others are required.

To wrap this part up, we can say that the difference in nature of OSSD and SCRUM is the same with the difference between Global software development and Agile approach, in a more extreme extend. Contributors in OSSD are distributed in term of location, time zone, culture and they mostly have no obligation upon their contribution. Therefore, without proper approach, OSSD possesses a lot of potential risks, especially OSSD with SCRUM as the main practice method.

Although there are so many difficulties in theory when applying SCRUM in OSSD, with the proper tweak and approach for each specific case, SCRUM turns out surprisingly to useful for OSSD. We will discuss the successful cases of SCRUM in OSSD in later part and then find out how and why it can happen that way.
3.2 Evident case of applying SCRUM to OSSD

SCRUM and OSSD are very similar to each other so it is not a surprise that they can be applied to the same project. They both depend highly on teamwork which is the main factor why they are both successful individually. In this section, we will be searching for and writing about evidences of how SCRUM was applied to OSSD.

Lavazza et al. (2010) did an empirical evaluation of how agile methodology, SCRUM to be specific can be applied to OSSD. This study was based on a case study of a Java project called MacXim, an acronym for Model And Code Xml-based Integrated Meter which is basically applied to a source code to mine for static measures (Lavazza et al. 2010). OSSD is naturally not a structured process as developers work in cooperation with others and do not apply any particular methodology while SCRUM is done in an organised way with the application of a methodology. According to this article, the project was simulated to achieve a result of both before SCRUM was introduced and after it was introduced to show what the effects will be. (Lavazza et al. 2010)

The MacXim project was started normally following an unstructured process and the results pointed to the fact that the developers lacked organization and communication. SCRUM methodology was then introduced as a way of combating the problems of organization and communication which proved to be effective in solving the problems. Comparing the results of output quality, there was no negativity between pre-application and post-application of SCRUM and also did not affect the speed of production. SCRUM made a difference in the organization of the development process in the end and a production of better code quality was also recognized. (Lavazza et al. 2010)

This article shows that SCRUM can be applied to an OSSD project and have a positive impact. It adds organization to an OSSD project even though the team might not be able to be in contact with each other physically.

Another case to be considered will be the library ERP tool discussed by Sutherland et al. (2007) in their article titled Distributed scrum: Agile project management with outsourced development teams. This article is not an OSS project but an OSS tool was used in order to complete the project and since example articles were scarce, hence its inclusion. The article considered the business practices used by companies like Toyota, Honda, canon and Fuji-Xerox for implementing project management and identified SCRUM as the agile methodology utilized. They also picked an American company called SirsiDynix that develops software and other services for libraries all over the world and took a look at one of their projects which was carried out through the process of outsourcing. They use integrated SCRUM to ensure transparency, communication and teamwork between its cross-site teams in the US, Canada and Russia. (Sutherland et al. 2007)
SirsiDynix avoided the problems of additional hidden costs by outsourcing the initial development process and managed the risks using integrated SCRUM thereby increasing output per individual of a team. Integrated SCRUM model endeavors to achieve maximum output from a tight market situation with the limited resources available. They were also faced with the issue of producing a quality software, experience of teams involved in the dynamic business environment, additional requirement that arose during the project and the ongoing merger process (between Sirsi and Dynix). All these issues combined the SirsiDynix knowledge of SCRUM and also the experience of StarSoft with XP made them choose to implement Integrated SCRUM as a practice. (Sutherland et al. 2007)

The planned output was to be a functional library ERP system which uses the latest technologies, met current library standards and should be easy to sell to potential customers. It was decided that the programming language will be Java, has all the necessary support functions and functional on different platforms. (Sutherland et al. 2007)

Perforce which served as the OSS was chosen as the source code repository which has a proprietary license and is under the GNU public license. Perforce is client/server system that supports diverse platforms which are connected to each other. (PERFORCE End User License Agreement for Open Source Software Development) It proved to be the effective tool for the project since the teams involved were distributed.

### 3.3 How SCRUM shaping the way people practice OSSD

As it was mentioned above, SCRUM and OSSD basically have different interaction ideologies – SCRUM relies on close cooperation between developers with regular face to face meetings, while OSSD is mostly conducted by people from all over the world. However, in open source projects it is possible to use so called “Distributed SCRUM”.

In OSSD the role of the product owner and SCRUM master is combined. Usually it is a project founder or one of the core developers. In the commercial OSS projects this role is given to the product manager. As in every SCRUM project, this person in responsible for product backlog, software updates tracking and SCRUM teams managing. SCRUM team in OSS can be divided to two groups – core SCRUM team and extended SCRUM team. Core team follows the whole development, testing and documenting process, while extended team contribute to the project on occasions. (Purkayastha 2014)

Sprints length in OSSD is not strictly stated as in regular SCRUM projects. While usually time is limited from two to four weeks for completing a point from product backlog, in OSSD can take both less and longer time period. This difference can be explained by different size of SCRUM teams and geographical isolation of team members. Daily SCRUMs in fact are daily only at big commercial project with full-time involved developers. In small projects daily SCRUM session are held once per one or two weeks. (Purkayastha 2014)
However, SCRUM distinctly improves the level of project organization and evaluation. SCRUM organized project requires the developers and contributors to communicate regularly for essential SCRUM sprint discussion and receiving timely feedback. While in traditional daily SCRUM meetings workers present in the same location, in OSSD developers arrange communication with video conferences, forums or messengers. Voice and video communication provide the clear understanding of co-workers participation objectives and contribution. As in all agile approaches people and relationships are the most important point, it is significant that video and voice discussions increase the productivity level and project quality than rigor and official emails. (Lavazza et al. 2010)

In order to improve the development process SCRUM provides several approaches for use. Screen sharing and paired programming is an effective way to hold the design and implementation practices. Use of screen sharing software gives the same opportunities as joint programming in common areas of non-distributed projects. (Sutherland et al. 2007) Regular discussion with co-developers allows to avoid project complication. When workers do the projects separately from each other, it sometimes happens that they prefer not to depend on other members and tries to isolate the own project part, make it as much autonomous as possible. As a result, the whole project becomes an overloaded compilation of separate working modules. Applying SCRUM allows to create the joint product working as a complete and simple as it possible unit. Moreover, the distributed developer does not always have a capability to test the product in user's circumstances - the environments and machines performances are not the same. Working in the close collaboration with various developers provides more profound operability test stage. Specifically, SCRUM offers developers the idea of separate verification – the piece of code is finally accepted after checking and testing by the person different from the author. In consequence of such collaboration, the level of errors on production stage is substantially decreased and collective conception is increased. (Giera & Brown 2004)

Taking everything into account, it is possible to say that SCRUM methodology can be successfully used in OSSD process. SCRUM approach can simplify feature implementation process by sprint sessions, increase the level of developers communication and co-worker roles understanding. As a result the final product comes out in the most qualitative and efficient form.

4. Conclusions

In this article, we have tried to prove that SCRUM is totally viable and in fact becoming one of the most important method to do OSSD in practice. Even though SCRUM and OSSD seems to contradict each other at the first glance, they are in fact a perfect match to achieve the same goal of fast and self-sustained development. There are many different way of using SCRUM in OSSD but Distributed SCRUM or Open SCRUM is the most widely known.
Still, beside clear advantages of SCRUM in OSSD, there are not many example cases or empirical researches in the field yet to prove our theoretical conclusion. An empirical study could be considered for further study to improve the validity of this article.

Another thing to discuss in OSSD is that some of the most established projects don’t have urgency in the development project, meaning that some of the critical features might take a long time to be completed. However, there are also some other up and coming open source projects (some might not be mainstream) where the pressure for releasing new features is tremendous which require the community to strike forward all the time. In those cases, Open SCRUM excels the most.

This article is the first article of our team at Master degree level so there are apparently many things that can be improved. The number of literature to backup our argument is one of the things that might affect the validity of this article. Our chosen source for performing the literature search, which is google scholar, might also be too general for the particular topic. Using other literature source such as IEEE might provide deeper literature and more credential result. Time and resource limitation are the others factors that prevent us from providing more systematic and statistical example cases.

However, we hope that the article can help to open a new direction for readers to implement better practice in open source software project, especially in applying SCRUM. On the other side, this paper can be the initial source for other students or researchers to start their journey on exploring Open SCRUM.
References

http://hillside.net/plop/plop98/final_submissions/P49.pdf

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.466.2290&rep=rep1&type=pdf


https://ulir.ul.ie/bitstream/handle/10344/3327/Fitzgerald_2000_framework.pdf?sequence=4

https://bia.unibz.it/bitstream/handle/10863/1409/Applying%20SCRUM%20in%20an%20OSS%20development%20process.pdf?sequence=5&isAllowed=y


http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.413.2992&rep=rep1&type=pdf

PERFORCE End User License Agreement for Open Source Software Development
http://dl.ifip.org/db/conf/oss/oss2008/PetrinjaSS08.pdf


https://www.mountaingoatsoftware.com/articles/introducing-an-agile-process-to-an-organization


https://scholarworks.iupui.edu/bitstream/handle/1805/8897/purkayastha_2014_openscrum.pdf