Collaborating Multiple Agile Teams: Scrum of Scrums

Trupti Khatavkar
tkhatavk@asu.edu
Arizona State University
Tempe, Arizona, United States

Srajan Gupta
sgupt182@asu.edu
Arizona State University
Tempe, Arizona, United States

ABSTRACT

Agile development model has its own advantages such as, flexibility, transparency, early and predictable delivery and it focuses on users. Taking these into consideration, large projects can be developed using agile techniques by dividing them into different components and distributing the components to different teams. This is generally termed as scrum of scrums. In such large project, each team should take the full responsibility of their own component and then the components are interfaced to build the whole system. The main challenge is to maintain minimum dependencies among teams as possible. This paper describes the challenges faced when multiple agile teams work on the same project and suggests methods to achieve successful large agile project. Also, methods to overcome architecture or design related risks are also put forward.

KEYWORDS

agile, components, dependencies, collaboration, communication, scrum, interaction

ACM Reference Format:
Trupti Khatavkar and Srajan Gupta. 2019. Collaborating Multiple Agile Teams: Scrum of Scrums. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

For developing any business, software development is a core component [2]. With great competition There has been a great investment in building high quality software. A software is developed in multi-site with multiple teams involved with a distributed environment. As discussed in [4]Past few years, in order to allow changes and catch up with the new technology, the development community is using the agile software development, where the software develops gradually and this done between self-organizing cross-functional teams[3].

This paper discusses some ways of achieving the practice of software development using agile method and how different agile teams come together to build a software[5]. It also discusses the challenges which agile principles help in overcome in projects which are distributed. Along with this, some of the techniques which can be incorporated in order to handle those challenges.

The paper is structured as follows : Section 2 discusses the methods of obtaining a large scale agile project which is distributed among many teams, what are some common practices to be followed among different team so that each one has the feel of the project, its design and architecture[1] and agile software development methodology as well. Section 3 concludes it and list down some benefits of this software development process.

2 CHALLENGES FACED DURING SCRUM OF SCRUMS

- Communication issues
  As there is informal communication among agile teams, there can be lack of communication. It would be difficult to express ideas to other teams. This can lead to lack of trust.
- Lack of coordination
  All teams would not be on same page, and that can be a problem while interfacing the components. Also, all developers would not have idea about the whole project, which can be an obstacle in the development.
- Handling dependencies
  As different components are being developed by different teams, dependencies need to be handled by communicating with other teams which is a complicated task.
- Loss of Centralized power
  Using Scrum of scrums technique, there is no centralized entity that makes decisions and distribute tasks.

3 METHODS TO ACHIEVE SUCCESSFUL SCRUM OF SCRUMS

The main challenges for working on a large project using multiple agile teams is communication, dependencies between components and interfacing those components. To achieve a successful project, following methods can be useful[5].

- Separate Product Backlog for each team
  As shown in Figure 1, it is very important that every team should have a strong product backlog in agile when working in a distributed environment. But, clear separation of work is equally important. Every day to day operations should be separated among the teams. Overlapping situations should be planned during the cross team meetings.

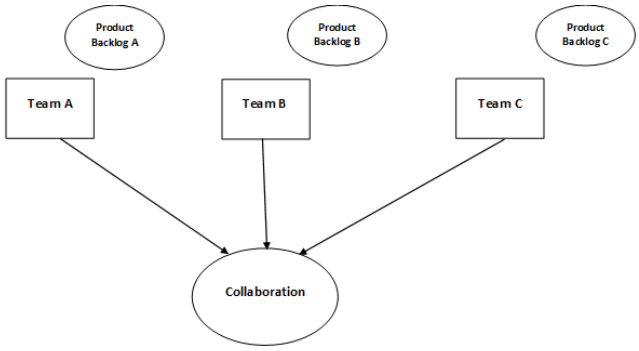

Figure 1: Collaboration of teams and their backlogs.

- Tools for Interaction
  Choosing the correct tool for communication matters when working in distributed agile environment. Slack and GitHub are very effective for this.
- Agile Practices
  All the important agile practices such as stand ups, planning, deliverables and retrospectives should be defined by each team as they think is suitable.
- Face to face communication
  Using tools for communication is effective, but many times there are broken communications due to various technical issues such as poor connections, fault in the devices, etc. To overcome these, regular cross teams face to face communications are also important. Also, face to face communication is less time-consuming and gives a clear idea.
- Fixed Architecture/Design
  As there are multiple teams working on the same project, there can be a confusion with architecture and design of the system. It can get messed up as everyone would not have knowledge of everything. Thus, development should be done, keeping this in mind.

4 ARCHITECTURE/DESIGN FOR SCRUM OF SCRUMS

Architecture defines the structure of the project. A well-defined architecture makes a base for a successful project. As multiple teams work on a single project, it is important to have a fixed architecture so that different teams can work on different components of the architecture. If there is no fixed single plan, interfacing of the components would be a difficult task. Also, the architecture can get messed up if multiple teams update it. The integrity of the whole project can be affected if architecture is messed up.

To overcome this problem, there should be a single Chief Architect (or two, working in a pair) who will guide the teams and ensures they do not face problems with the architecture [2]. Chief architect works on a high level architectural issues that involves all the components and interfaces between them.

Another method would be to create the Architecture group. It is composed of each team's most skilled person. This group handles all the architectural issues rather than one person handling them. The group should not exceed more than 20 people, else communication problem arises. The group understands the subsystem's relationships and the architecture as a whole. Thus, the group decides how each team works and handles the project.

5 CONCLUSION

With different teams working on a same project using the agile methodology, it is possible to make the best use of available talent across the organization and not outsourcing it while increasing the cost of the project. It is a conscious decision to include multiple teams in one project considering the different strategies, talent, focus and management of different teams can be the factors in success, delivery time may cause dysfunctioning. But, these factors needs to be clearly understood beforehand.

There are many benefits of using different agile teams for software development. We can track the progress and evaluate the problems early in the stage, also handles the problem of isolated teams, where there are difficulties in communication. It can help in sharing knowledge between different domains while different teams work together and look after the development with all aspects.

With the benefits comes some challenges as well. Since there is informal communication in agile software development, it can lead to loss of information among teams and with this there can be a lack of trust. There is a lot of coordination required so that every team is on the same page and knows exactly at every step what is happening within the project. Each team should be making equal efforts towards the project development.

With proper coaching to themselves as a team is very important in collaborated development[4]. Hence, with a little and right amount of modification in the existing agile techniques can be very helpful in overcoming the challenges faced and in term increase the production of high quality softwares.

REFERENCES

[1] Mauricio José de Oliveira De Diana, Fabio Kon, and Marco Aurélio Gerosa. 2010. Conducting an Architecture Group in a Multi-team Agile Environment. In *Brazilian workshop on.*

[2] Henrik Kniberg and Andres Ivarsson. 2012. Scaling Agile @ Spotify with Tribes, Squads, Chapters & Guilds. (Oct. 2012). https://creativeheldstab.com/wp-content/uploads/2014/09/scaling-agile-spotify-11.pdf

[3] Nils Brede Moe, Darja Šmite, Aivars Šāblis, Anne-Lie Börjesson, and Pia Andréasson. 2014. Networking in a Large-scale Distributed Agile Project. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '14)*. ACM, New York, NY, USA, Article 12, 8 pages. https://doi.org/10.1145/2652524.2652584

[4] M. Paasivaara, S. Durasiewicz, and C. Lassenius. 2008. Distributed Agile Development: Using Scrum in a Large Project. In *2008 IEEE International Conference on Global Software Engineering*. 87–95. https://doi.org/10.1109/ICGSE.2008.38

[5] Sunit Parekh. 2015. Collaboration Techniques for Large Distributed Agile Projects. Retrieved Jan 18, 2018 from https://www.thoughtworks.com/insights/blog/collaboration-techniques-large-distributed-agile-projects