Collaboration among Agile Teams

Carnic
clnu2@asu.edu
Arizona State University
Tempe, Arizona, United States

Rishab Mantri
rsmantri@asu.edu
Arizona State University
Tempe, Arizona, United States

ABSTRACT

The technology industry is moving at such a fast pace, that it led to the introduction of the 'agile methodology' that allows changes even after the requirements have been stated. The agile manifesto talks about certain rules that a SCRUM team has to follow in order to be a self-organised team. However, having said that, SCRUM can be implemented only on a small team. It does not talk about how multiple teams following *agile* should work and respond to changes. Many projects are larger and need more than one team to succeed, which may result in issues for project managers on how to effectively coordinate. Although coordinating large, multi team projects is not new, but doing it in agile projects is new.

This paper presents a solution that can be used to solve this problem. Making rules similar to that of scrum, will lead to an effective collaboration among the teams in order to result in the desirable product. It is highly important to be able to perform project management in the best way possible to grow in this market, especially if the project is at a large scale.

KEYWORDS

agile, components, architecture, dependencies, collaboration, communication

ACM Reference Format:
Carnic and Rishab Mantri. 2019. Collaboration among Agile Teams. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In the last decade, software development is expanding. Software is now used into almost all fields and is becoming more complex. With customer requirements changing frequently it has made it more difficult to develop software. Due to tremendous competition in the market, changes are frequent to any software product which is under development[4]. The priority of requirement also vary, so software development is done to a specific module which is required[5]. Changes and improvement are done later for the remaining modules. Classical process models are no longer able to satisfy the new requirements of the market in the best way. To solve such problem, new software development approaches are evolved, as agile methodologies. The new methodologies include modifications to software development processes, to accommodate changing requirements and make them more flexible and productive.

The Agile Manifesto emphasizes on individuals and interactions over processes and tools, motivating face-to-face communication. As it is the most effective way of exchanging information between Agile teams. Effective collaboration is a necessity for teams to come up with solutions to complex problems. However, this is constantly challenging for teams having a large number of people and located in different places.

2 PROPOSED APPROACH

Although the rules for following SCRUM are pretty straightforward when it comes to one group, it is uncertain as to how multiple teams should collaborate together. However, following the laid back pattern of agile, we can form something similar to what we can call the scrum of scrums[1]. The teams will work individually following the rules specified in the Agile Manifesto, however, similar ones can be applied to make the teams collaborate effectively.

2.1 Software Architecture

Architecture-relevant decisions mostly involve systematic mapping of key requirements to software architecture. Only when done in a systematic way, development teams can verify they implemented the key requirements correctly. Software architecture works as a blueprint of the system and the project. Developing complex or larger softwares can often fail if the architecture is ignored. Architecture usually implies the mapping of key requirements to the system. When followed correctly, the verification of the system and interaction with the stakeholders can become a lot easier.

2.1.1 Particularly in multiple teams. Although agile provides flexibility in terms of change in requirements, having the basic architecture predefined gives the developers an understanding of how the system is supposed to look like because the product vision overall remains the same. It is vital for all members to not just understand the requirements, but also the architecture of the project. Because it is being implemented on such a large scale, each member should know where their contribution will be linked to the final product[2]. This means that they should have a clear idea about the stage of the project when they have to add their product. Knowing the architecture is basically knowing the plan of the system. Therefore it makes the communication among the teams and stakeholders much easier and the decision making process much more efficient. It also means that the model of the software will be ready and will be available for reuse if required. Having multiple teams working together, with each having their own product being built somewhat separately, can cause major problems during integration if they don't understand the base of how their system is supposed to work. This can lead to the failure of the entire project as well.

2.1.2 Software design. While software architecture will provide the developers with the model, they need to design the software as well to divide the work in order to work as independent teams. It will prevent redundancy and ensure that their work does not interfere with each other, minimising dependencies[3].This will help the stakeholders and developers get a clear idea on how the software will look on completion. It is the initial vision of the

project, and if understood well, the teams will not have a hard time combining their separate products in the end.

2.2 Important factors

2.2.1 Healthy product backlog. Following agile at such a large scale can be pretty challenging. With the structure in mind, the team members should build a healthy product backlog and be aware of the requirements. Work distribution is key here. Each team should have a clear separation of work. That is, there should be as little dependency as possible. The ideal scenario would be when the multiple small projects just need to be integrated together in the end.

2.2.2 Communication. The teams need to coordinate with each other by always communicating with each other, in person, or using tools like Slack, Trello etc to make sure that one does not hesitate to ask. Distributed agile in general is a difficult tool, and helping each other along the way will get it done effectively.

2.2.3 The scrum of scrums. As it works in agile, each team should have multidisciplinary members with various skillsets so that they have a necessary cultural and structural change to support themselves. Within the team however, they should work like they do following scrum, and have their independency in order to be self organised. The scrum masters of each team can have something similar to a 'Daily Stand up' at the end of each sprint in order to discuss the progress of each team and talk about the impediments faced. If a team feels they need help from another, they can decide amongst themselves to come across some agreement. This will map cross team dependencies and help them understand how their separate efforts will contribute. If there are multiple product owners, they can check in more often to talk about the scope and acceptance criteria and prioritise their product backlogs whenever required. Cross team retrospectives will also be a part of this process to allow the teams to examine their progress and identify items that need to be improved. When a few teams start working together, they can have a separate short meeting to discuss the way they will go about it. This will reduce inconsistency when the individual efforts are integrated.

3 CONCLUSION

Agile development targets to make the best use of the talent available across different teams in an organisation. This helps in reducing the cost of the project. Benefits of having different teams work on a project is that it creates multiple point of views hence broadens the aspects of development. Also helps to share knowledge of different domains between the teams. We can evaluate problems at an early stage and also keep track of progress each team has made.

There are also certain challenges faced by teams. Agile uses informal communication between the teams that may lead to loss of information which in turn can cause serious problems and trust issues between the teams. A lot of coordination and communication is required for every team to know other teams progress. Each team should make equal efforts on development as well as to coordinate updates and progress with other teams for the project to be completed on time.

Hence for agile teams to work together effectively and to overcome the challenges, some modifications may be required in current agile methodologies. These modifications may vary from team to team.

REFERENCES

[1] [n. d.]. Collaboration Across Agile Teams. ([n. d.]). https://tech.gsa.gov/guides/Collaboration_Across_Agile_Teams/
[2] [n. d.]. The importance of a good software architecture. ([n. d.]). https://apiumhub.com/tech-blog-barcelona/importance-good-software-architecture/
[3] Nikolai Hyldmo. 2015. Coordination Between Teams in Agile Projects. (june 2015). https://brage.bibsys.no/xmlui/bitstream/handle/11250/2351134/13975_FULLTEXT.pdf?sequence=1/
[4] Sonia Thakur & Amandeep Kaur. 2013. Role of Agile Methodology in Software Development. (oct 2013). https://ijcsmc.com/docs/papers/October2013/V2I10201315.pdf
[5] SIEW HOCK OW MALIK HNEIF. 2009. REVIEW OF AGILE METHODOLOGIES IN SOFTWARE DEVELOPMENT. (Oct. 2009). Retrieved Jan 18, 2018 from https://www.arpapress.com/Volumes/Vol1/IJRRAS_1_01.pdf