Overcoming Challenges in Self-organizing Agile Software Teams

Invited Paper

Latha Karthigaa Murugesan*
*Department of Electrical and Computer Engineering
The University of Auckland
Auckland 1010
Email: lmur778@aucklanduni.ac.nz

Abstract—Self-organizing agile team is an important aspect of agile software development and are meant for sharing project management activities such as planning, estimating and requirements elicitation with managers and customers. While existing literature has explored some challenges in self-organizing agile teams, little is known about how the high involvement of self-organizing agile teams overcome the challenges in day-to-day activities. Through a Grounded Theory short study involving two agile practitioners, we identified a set of challenges to practice self-organization. This article compares the literature and the experiences of the agile practitioners to overcome the challenges identified. The article also shares practical implications and guidelines for agile teams and their managers for practicing some of these agile practices.

I. INTRODUCTION

Self-organizing agile teams is one of the important aspect of agile software development and these teams are meant to share project management activities such as estimation, planning, task assignment, task review, etc. Agile methods emphasize the importance of people-oriented/user-centred software development [1] and it follows an iterative and incremental style of software development where collaborate self-organizing teams adjust to changing project requirements [2]. Agile software development also include fast delivery and continuous improvement of the software. Though this seems interesting to enforce continuous delivery, it poses several challenges while practicing the features of self-organization. This article through analysis of interview and the prior literature identifies the challenges imposed on self-organization and the agile practitioners’ views of overcoming those challenges. The research questions were carefully designed to incorporate and recommend useful practices to the agile practitioners. The research questions that we aim to answer in this study are

RQ1. What are the challenges pertaining to self-organizing agile teams?

RQ2. Which practices help self-organizing teams to eradicate those challenges?

The article follows the following structure. Section 2 describes the literature followed by research method and results in Section 3 and 4 respectively. Section 5 describes the implication for practice followed by the conclusion in Section 6.

II. LITERATURE REVIEW

Self-organization is one of the principles behind agile manifesto [3] and is considered to be the core idea of agile software development. The principles of self-organizing are minimum critical specification and bounded autonomy, requisite variety and redundancy of functions, and learning to learn [4], which is also applicable to self-organizing agile team practices such as balancing freedom and responsibility, balancing cross-functionality and specialization, and balancing continuous learning and iteration pressure respectively as proposed through grounded theory (GT) [5] by Hoda et.al. [6], [7]. There are various challenges pertaining to self-organizing teams in terms of process [8] and product.

Fig. 1. Research Method

III. RESEARCH METHOD

Fig. 1 shows the research method that is used to answer the two research questions. Firstly, the interviews had been conducted with the agile practitioners for 20-30 minutes each and the open-ended questions asked during the interviews were based on the research questions RQ1 and RQ2. The interviews were transcribed and they were analyzed using the software tool, NVivo as mentioned in [9] to identify the challenges and to overcome those challenges.

<table>
<thead>
<tr>
<th>Participant Id</th>
<th>Agile Experience</th>
<th>Current Role</th>
<th>Team Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>~ 3.5 years</td>
<td>Senior Software Engineer</td>
<td>8-10</td>
</tr>
<tr>
<td>P2</td>
<td>~ 7 years</td>
<td>Project Manager/Scrum master</td>
<td>14</td>
</tr>
</tbody>
</table>
This research uses a combination of literature and the experiences from agile practitioners to identify the challenges in self-organizing agile teams. Two agile practitioners (P1, P2 - described in Table I) were interviewed from Abu Dhabi and India and their experiences are compared to the literature to overcome the challenges through the day-to-day practices. The interviews from the practitioners were transcribed and analyzed using grounded theory’s [5], [10] open coding technique. The example below explains the open coding technique from Interview transcription to the results of category, Tools.

**Interview Transcription:** "...online user-friendly tools such as JIRA need to be used across the team members who are either co-located or distributed. In this way, we can stay up-to-date with the changes in the project. We should be able to chat, evaluate, review the features of the project..."

**Key point:** “Improve team characteristics through user-friendly online tools”

**Code:** Tools, Team characteristics

IV. RESULTS

The important results of this article is to identify the challenges among the practitioners in self-organizing teams and the practices that are followed to overcome these challenges.

A. Answer to RQ1

This section identifies the challenges from the self-organizing teams and are categorized under three major categories. They are (a) distributed team structure, (b) multiple project goals, (c) lack of multi-featured user-friendly tools as discussed in Table II.

<table>
<thead>
<tr>
<th>Category</th>
<th>Challenge in each category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed team structure</td>
<td>1. Assigning over-qualified or under-qualified members to tasks</td>
</tr>
<tr>
<td></td>
<td>2. Lack of offline or uninformed discussion</td>
</tr>
<tr>
<td></td>
<td>3. In frequent communication due to time zone difference</td>
</tr>
<tr>
<td>Multiple project goals</td>
<td>1. Prolonged task switching</td>
</tr>
<tr>
<td></td>
<td>2. Less expertise if the project goals are irrelevant</td>
</tr>
<tr>
<td></td>
<td>3. Lack of cross-functionality</td>
</tr>
<tr>
<td>Agile software tools</td>
<td>1. Lack of multi-featured software tools</td>
</tr>
</tbody>
</table>

1) Distributed team structure: The most pointed out challenge both in literature and by agile practitioners is the distributed team structure [11]. The prominent role of self-organizing is to self-assign the tasks to be done [12]. Some of the challenges are (a) assigning overqualified or underqualified members to tasks, (b) lack of offline or uninformed discussion, and (c) infrequent communication due to time zone difference, as described below.

Allocating an under-qualified or over-qualified assignee which results in waste of time and cost of the project.

"...sometimes underqualified people who have no idea of the task assign it to themselves. Sometimes overqualified people self-assign the tasks. In both cases, both time and money gets wasted..." - P1, Senior Software Engineer

The co-located team finds it easier as they tend to discuss it online and as well offline. The offline or uninformed discussion among the team adds value to the co-located teams, whereas this opportunity seems to be impossible.

"...I find it easy to have discussion with ‘x’ working in my team here, but this will never happen with my onsite team..." - P1, Senior Software Engineer

The time difference between the onsite and co-located teams results in infrequent communication among the team members [13].

"... as most of our projects are with USA clients, who are 9 hours behind us, the communication between us is quite rare and the number of people communicating with each other is also very less..." - P2, Scrum master

2) Multiple Project Goals: Some of the team members may work in more than one project at the same time. In addition, same project might have several project goals to be achieved at the same time in terms of different phases of software development. For instance, a person may have multiple goals such as implementing a task/module or planning testcases for the task/module. This poses lot of challenges among the team members in several ways. They are (a) Prolonged task switching, (b) less expertise if the project goals are irrelevant, (c) When a person needs to concentrate on two different deadlines, switching from one task to another would take a lot of time.

"... when I was a developer, I once had a tough time managing two tasks at the same time... " - P2, Scrum master

In addition, having more than two goals needs to be avoided or the WIP (work in progress, as in Kanban [14]) limit for the project needs to be practiced and the time management needs to be enforced to achieve reasonable goals [15].

When the goals are irrelevant for a team member, the possibility that he has less expertise on one or more of the project goals is very high. In addition, people sometimes stick to similar kind of project tasks in terms of technology used, type of task, etc. They tend to lack the feature of cross-functionality.

"...I would like to make my team cross-functional. So, even in absence of anyone the team should proceed to move forward" - P2, Scrum Master

3) Agile Software Tools: The most prevalent component to enforce agility is the use of the agile based software tools for various phases of software development such as creating product backlog and sprint backlog, assigning tasks to the users, developing testcases, assigning testcases to testers, reviewing the product by checking against the acceptance tests, in-team chat facility, project status update, etc. There are several software tools available in the market ranging from licensed to open source. Though a software tool have several features, there are few instances where a tool might be able to handle from planning to maintenance.

"...open source agile software that we use do not have ability of chat facility with onsite team...” - P1, Senior Software Engineer
B. Answer to RQ2

Several studies identified solutions to overcome the above said challenges [16], [17], [18]. This section help compare those solutions and develop certain agile practices based on the interview from agile practitioners and the literature.

1) Distributed team structure: Although self-assignment is one of the features of self-organizing teams, the assignment could be reviewed by the scrum master before allocation to avoid wastage of time and money. There should be a balance between the agile and traditional software development feature. The customised agile practices are carried out by the agile practitioners.

"...we always mix agile practices and traditional software development practices to carry out the project smoothly without any delay or wastage of time and cost..." - P2, Scrum Master

2) Multiple Project Goals: When a team member is doing multiple project or multiple goals in a project, he needs to limit the amount of work that is done in a particular period of time. As there is a Kanban board for each project, there needs to be Kanban board for each team member who is doing multiple projects at the same time. In that way, they can track their progress and achieve maximum results by limiting the work in progress (WIP) limit of the team member. Also, it may be quite important for that team member to choose or to be assigned with the relevant multiple tasks rather than irrelevant ones. The reason behind this is that the relevant tasks requires less task switching time whereas irrelevant ones such as different projects or different phases of projects might result in increased project cost which needs to be avoided. Moreover, this may end up in doing tasks by the team member with less expertise. This may be avoided by allocating more resources to the project or pipelining the goals appropriately.

"...I feel pressured when I have several deadlines..." - P1, Senior Software Engineer

To overcome the challenge of solving the problems with less expertise while chasing multiple goals, a good motivational team needs to be recruited who has the enthusiasm to learn and adopt to the changes in the project environment. Every member in the team needs to be trained to work cross-functionally inculcating the importance of it. This doesn’t stop the project development from being stalled.

"I wish my team members are cross-functional..." - P2, Scrum Master

3) Agile Software Tools: The agile software team faces problems of managing and practising agile software development due to lack of good software tools. There are several planning and tracking tools exist in the market such as JIRA, Trello, Scrumwise, Acunote, MASE, etc. As Nerur et.al. [17] said, the organizations must be able to invest in software tools to facilitate and support rapid iterative development. Also, while using a visualization tools, it is easier for the team to relate what other team members are accomplishing [19].

The organization must be able to invest in more than one tool to accomplish several tasks. Also, various tools support plugins to impart additional values to the software development. Also, internal project development team should invest to develop various plugins or relevant tools suitable for teams to practice agile. Moreover, they should have the ability to customize the feature in the tool for optimized usage.

V. DISCUSSION

A. Implication for Practice

The software team must be able to do the following practices to overcome the challenges pertaining to the agile software development. They are (i) several customized software tools with extended plugins need to be used, (ii) one team member should be limited to not more than one task to ensure finishing it on time and to avoid prolonged task switching, (iii) Avoid time and project cost wastage by equipping the team members cross functionally, (iv) distributed team challenges could be avoided by increasing the communication time between the teams efficiently, (v) The scrum master must be able to help the assignment of tasks to the team members without any micro management. Last but not the least, infrequent communications lead to lot of problems in the agile software development and the agile practitioners should always consider avoiding this issue in distributed agile teams.

VI. CONCLUSION

Grounded Theory (GT) approach is an useful method to analyze and develop theory for any raw data such as interview transcription, literature information, etc. The use of NVivo eased the process of GT to identify the challenges in self-organizing teams and the way to overcome those problems. The challenges were categorized under three categories as distributed team structure, multiple project goals and lack of multi-featured tools. There were several subcategory challenges that were identified to design the agile practices in more practical way. There were several customized agile practices which were discussed that can help the team to overcome the challenges identified in the study.

Though self-assignment is one of the primary features of self-organizing teams, it can still be customized to impart the combination of agile and traditional software development practices. The implication for practice can be very well utilized by the agile practitioners to overcome the challenges faced in self-organizing agile teams.

ACKNOWLEDGEMENT

The author would like to acknowledge the participants for their participation in this research.

REFERENCES