Exploring Workflow Mechanisms and Task Allocation Strategies in Agile Software Teams

Zainab Masood*, Rashina Hoda, Kelly Blincoe
SEPTA Research, Department of Electrical and Computer Engineering,
The University of Auckland, Auckland, New Zealand
zmas690@aucklanduni.ac.nz,
{r.hoda,k.blincoe}@auckland.ac.nz

Abstract. Task allocation is considered to be an important activity in project management, independent of the method used for software development. However, the process of allocating tasks in agile software development teams has not been a focus of empirical research. This research used case study research method and involved semi-structured interviews with 12 agile software practitioners working within a software development organization in India. The results explain the process of task allocation including four different mechanisms of workflow across teams: team independent, team dependent, skillset/module dependent and hybrid workflow. These include some common set of practices whereas others are team-specific factors. Furthermore, five different types of task allocation strategies were identified based on increasing levels of team and individual autonomy: individual-driven, manager-driven, team-driven, manager assisted and team assisted. Knowing these workflow mechanisms and task allocation strategies will help software teams and project managers to make effective distribution and allocation of tasks for smooth agile software development. Future research can suggest alternative methods to overcome the hurdles faced by agile teams during allocation of tasks.

Keywords: Task Allocation; Workflow; Allocation Mechanism; Agile Software Teams; Task Allocation Strategies;

1 Introduction

Successful completion of a project depends on how well and effectively the project activities are planned and executed throughout the development of the project [1]. Primary project management activities include managing the resources, task allocation, tracking time and budget in the best possible way [2]. These activities are executed differently depending on the choice of the management approach. For example, in traditional projects, a project manager is responsible for planning these activities and

* Corresponding Author: Building 903, 386 Khyber Pass, New Market Auckland 1023,New Zealand. Tel: +64 9 923 1377

allocating tasks to the team members in a commanding role. Many studies researched task allocation aspects in global and distributed software development using traditional non-agile methods. Few assessed the task allocation mechanism practiced by community-based Free/Libre Open Source Software(FLOSS) development teams [13,14]. It should be noted that this research did not cover commercial projects rather the focus was the self-organized teams in organizations.

With the evolution of agile methods, the concepts of 'light touch' management and self-organizing teams were introduced [3]. Agile teams are meant to be self-organizing displaying high levels of autonomy and decision making in their everyday work [2,16,17] including project management activities such as task allocation [2,4]. In practice, however, agile teams are seen to display varying levels of autonomy as they gain experience of functioning in a self-organizing way [4]. How this is played out during task allocation is not well understood. In particular, it is unclear how work flows to and within the team, how tasks are allocated on an individual level, and what are the different types and autonomy levels of task allocation in agile teams.

We conducted case study research [5] involving data collection through face to face interviews from 12 agile practitioners from an organization in India. Thematic analysis [6] was performed to derive the different types of workflow mechanisms and task allocation strategies. We identified four workflow mechanisms namely: team independent, team dependent, skillset/module dependent and hybrid workflow. We also identified five types of task allocation strategies, namely: individual-driven, manager-driven, team-driven, manager assisted and team assisted. Identifying these mechanisms and strategies helped understand the flow and forms in which tasks arrives to the team and made it is easy to analyze the basis on which tasks are classified and allocated.

The rest of the paper comprises of: Section 2 which outlines the related literature on agile software development and task allocation mechanisms. Section 3 covers the research method. Section 4 presents the findings of this study with respect to team workflow mechanisms and the common strategies for task allocation in respective sub-secion . Sections 5 covers the discussions which is followed by the limitations of the study with future directions in section 6. The last section of the paper presents the conclusion.

2 Related Work

Agile development methodology emerged as an alternative to conventional, sequential and phase-based development [7]. It follows an iterative and incremental approach to development and is open to changes throughout the project [2]. Core values of agile are captured in twelve principles defined in Agile Manifesto [7]. Agile software development uses short iterations to present developed features as working product to the relevant

stakeholders hence adding value by their continuous feedback [8]. This enables the project team to gain confidence on product quality throughout the development lifecycle. The research on task allocation in software teams has been largely dominated by distributed contexts in global software development. Imtiaz et al. in their recent survey-based study identified "functional area of expertise and phase-based" task allocation as the most common way of allocating tasks in global software development [12].

Crowston et al. 2007 [13] through their empirical study demonstrated the possible mechanisms of tasks allocation in community-based Free/Libre Open Source Software (FLOSS) development in self-organised volunteer teams. They identified *self-assignment, assign to a specified person, assign to an unspecified person, ask an outsider, suggest consulting with others* as the possible ways of tasks allocation. Their findings support self-assignment as the most common way of assigning tasks in self-organised software teams where self-assignment is a self-directed task allocation strategy practiced by teams. Similarly Crowston [14] while demonstrating bug fixing process adopted by FLOSS teams, specified *assigned to team members, self-assignment, assigned to members external to the team* as the task allocation mechanisms. However, not much has been explored in the literature about task allocation mechanism outside the FLOSS domain and specifically for commercial software development.

Work on task allocation mechanisms in agile software teams across the literature is envisioned on a very limited scale. Literature based research explored the contribution with respect to task allocation in distributed software development that incorporated agile practices [10]. Despite some evidence in the literature, these task allocation methods have been rarely verified in practice. David et al. [11] proposed an approach for solving the issues faced by the manager related to task-coordination and allocation in an agile distributed software development environment. The study states that firstly tasks should be prioritized and at later stage team members are ranked according to the suitability of tasks. These two rankings are combined together for allocating tasks to the team members. Much remains to be understood about how work flows to and within agile teams and how task allocation is practiced by them.

3 Research Method

This research explores and understands the process of task allocation in software teams practicing agile methods using a case study research method [5] with thematic data analysis [6]. The main research questions governing the research were:
Question 1: When and how does task allocation happen in agile software teams?
Question 2: What are some of the strategies for allocating tasks across different projects and teams?
Prior literature on task allocation was explored. Subsequently, a set of questions was drafted for conducting the interviews. The questions were designed keeping the objective of the research in perspective. Broad research questions were classified into more precise

questions to collect appropriate details. Twelve software practitioners from a software company in India were interviewed personally by one of the authors. The results gathered through interviews were transcribed and analyzed using thematic analysis to generate the findings. This was led by one of the authors and supported by the other two through careful reviews and discussions.

3.1 The Interview Questions

The questions designed for the study consisted of three sections and aimed at covering major areas of task allocation. Overall the interview was semi-structured with a combination of open-ended and close-ended questions. The first section aimed at gathering the demographical data of the participant such as their professional experience and age range. The second section focused on collecting details related to the project and the team structure, such as team composition, agile methods used and project domain.

The third section focused on task allocation related information gathering i.e. process which developers follow in their daily work for allocation of tasks. First sub-section of these interview questions inquired how, when and from whom the tasks are received by the teams and individuals. These were open-ended questions to allow a range of answers, with some choices being given to facilitate the interviewees during the interview. As an example one of the sample questions was: “In what form does work arrive to your team and individually?”. This flow was further investigated with follow-up questions “How does it happen? Please provide an example.”

Second sub-section of the interview questions focused on the practices followed for task allocation by the participants. Some of these included: “How often are tasks allocated to you by a customer/manager/lead without your input?”, “How often are tasks allocated to you through team discussions and consensus?” These questions were aimed to explore self-assignment of tasks and allocation through other stakeholders of the project. These stakeholders included Managers, Technical Leads, clients/customers or other members of the team in managerial positions.

Last set of interview questions evaluated the current strategy of task allocation: "How well is the current task allocation style working for: 1. the project (outcomes) 2. the team (as a whole) 3. you (personally)?"

3.2 Data Collection

Invitation to participate was sent out to members of the agile software community of India. Upon their confirmation, one of the companies was approached for data collection. The interviewer started with elaborating the overall purpose of the study and then verbally asked the questions. The scales used for answers were described to clarify any confusion for the ratings based questions. On average it took around 30-40 minutes per interview. All the interviews were recorded with detailed notes taken during the interview. The interviews were transcribed and analyzed manually. The notes taken were cross-verified by the co-authors using the recordings to ensure the consistency and accuracy of the data.

3.2.1 Demographic Information

Most participants involved in the research were experienced software practitioners and were directly involved in software development related activities for software projects from diverse areas.

Table 1. Participants Demographics (Total Exp: total experience; Agile Exp: agile work experience)

Team	Participants	Job Title	Age Group	Total Exp (Yrs)	Agile Exp (Yrs)
T1	SP1	Tech Lead	31-35	10-11	6-7
T1	SP2	Software Engineer	21-25	2-2.5	1
T1	SP3	Assoc. Tech Lead	26-30	4-5	4-5
T1	SP4	Software Engineer	21-25	2.5	2.5
T2	SP5	Tech Lead	36-40	7	7
T2	SP6	Sr. Software Engineer	26-30	4	2
T2	SP7	Tech Lead	31-35	7.5	7.5
T3	SP8	Software Engineer	21-25	3.5	3.5
T4	SP9	Tech Lead	31-35	5.5	5-6
T4	SP10	Assoc. Tech Lead	26-30	4	2
T4	SP11	Sr. Software Engineer	21-25	3.5	1
T4	SP12	Assoc. Tech Lead	26-30	4.5	2

Table 1 lists the software practitioners who participated in the study. The software practitioners are referred by identifiers SP1-SP12 with teams referred by numbers T1-T4

3.2.2 Project and Team Related Information

The company of the participants is a digital technology company and as major part of their business they develop enterprise softwares and provide software related services with teams of over 750 professionals. They have clients and operations across 30 countries worldwide including United States, Europe, Middle-East, South-East Asia, Australia, and India. All the teams collaborate with other teams and sub-teams located in different countries. Project and team details are profiled in Table2.

Table 2. Project and Team's Contexts

Team	Team Size	Project Context	Project Duration	Project Domain/Area
T1	10-15	New Requirements + Maintenance	3 years	IT-Digital Marketing
T2	5-10	New Requirements	1-2 years	IT-Analytics
T3	2-5	New Requirements	2 years	IT-HealthCare
T4	15-20	Migration + Enhancement	1-2 years	IT-Cloud Services

3.2.3 Software Methods Related Information

All teams participating in this study were using agile methods, either Scrum or Kanban or a combination of both. The teams primarily practiced Daily Team Meetings, Release and Iteration planning, Pair Programming, Review meetings and Retrospectives as listed in Table 3. The iteration lengths varied from daily to 4 weeks. Teams were collaborating with off-shore customers or product teams in the USA through Google hangout, Skype or Webex. All the participants mentioned that they were happy with the level of customer's involvement and rated it as good or excellent. The project management tool used by all teams was Jira.

Table 3. Software Methods

Team	Software Method	Iterations	DTM	RP	IP	PP	CD	RM	RE
T1	Scrum	2 weeks	✓	✗	✓	✓	✗	✓	✓
T2	Scrum	2 weeks	✓	✓	✓	✓	✓	✓	✓
T3	Scrum& Kanban	Daily	✓	✓	✗	✓	✓	✓	✓
T4	Kanban	4 weeks	✓	✓	✗	✓	✓	✓	✓

DTM = Daily Team Meeting, RP=Release Planning, IP= Iteration Planning, PP=Pair Programming , CD=Customer Demos, RM=Review Meetings, RE=Retrospectives

4 Findings

4.1 Quantitative Analysis and Findings

A combination of quantitative and qualitative techniques has been used to analyze the data and generate the findings of this study. The data collected was represented using matrices and tables [15] and the results displayed in form of graphs and charts. Graphical representation of results through bar charts has been used to display the quantitative data, as it helps to organize any mechanisms and common behaviors. For the qualitative part, thematic analysis [6] was used to derive the common themes or strategies of work and task allocation.

4.1.1 Individual Task Allocation Strategies

The task allocation practices followed by team members individually were analysed statistically based on the frequency rating of the practice followed. Each participant evaluated these practices on a 5-point Likert frequency scale. Subsequently, team wise average values of the answers were calculated and then aggregated to a single score for each team.

The graph in Fig. 1. is plotting different teams T1-T4 on the x-axis such that the lengths of the different grayscale bars representing practices are proportional to the frequency of practice occurrence on the y-axis with a scale from 0 to 5. The graph is portraying Team T1 as the first slot of grayscale bars representing the practices followed and listed on the right side of the graph. The results show T1 practicing "tasks are allocated to an individual with input from a manager/client/lead" as the most visible way, plotted with a light shade of black as the second bar in T1. This is achieved through an average rating of greater than 4, leading to a practice followed most of the time. Conversely, the graph

shows that the statement "tasks are allocated to an individual without input from a manager/client/lead" is true sometimes and so occasionally practiced.
T2, T3, and T4 rated that the task allocation through team mutually discussing tasks is the most frequent way. T3 in addition also agreed that self-assignment of tasks without any assistance is another common behavior observed by members of the team.

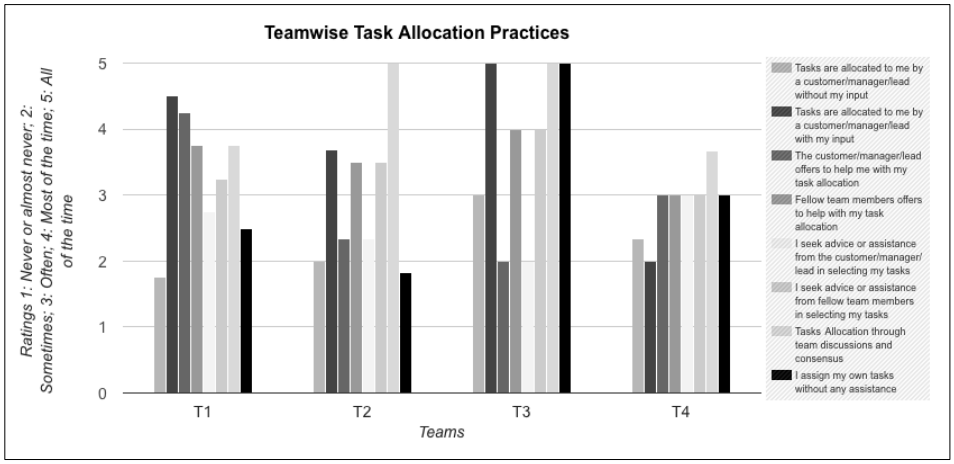

Fig. 1. Team-wise graphical representation of practices

4.1.2 Task Allocation Strategies

After analysing common individual task allocation practices and based on the task allocation mechanism followed by the teams on a daily basis, the task allocation strategies are broadly classified in five types namely: individual-driven, manager-driven, team-driven, manager assisted and team assisted. The Fig. 2. below summarizes the task allocation strategies.

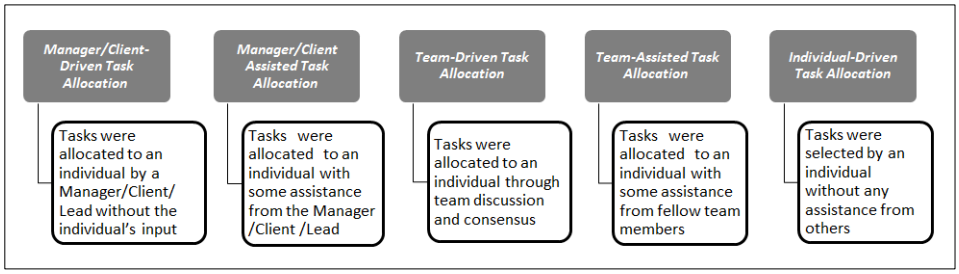

Fig. 2. Task Allocation Strategies

Another interesting aspect is uncovered in Fig.3 which correlates autonomy [16] with task allocations types/levels as (1) external autonomy: level of autonomy granted to the team by an outside team, manager or client (2) internal (or team) autonomy: autonomy within

the team (3) individual autonomy: on individual level. Interview questions targeting similar task allocation practices and settings were coupled together but their context measures autonomy differently, for this reason, autonomy scale is adjusted accordingly. e.g. with interview questions like "Tasks are allocated to me by client/manager/lead *without* my input" and "Tasks are allocated to me by client/manager/lead *with* my input". The response value to the latter question was flipped by subtracting it from the maximum autonomy scale value i.e. 5. This enabled both scores to be consistent and measured with the same frame of reference. Team wise average values of these scores were calculated and then combined to a single score for each team. The results are represented as a bar chart in Fig. 3, plotting computed autonomy scores against the teams highlighting the strategies through similar grayscale codes.

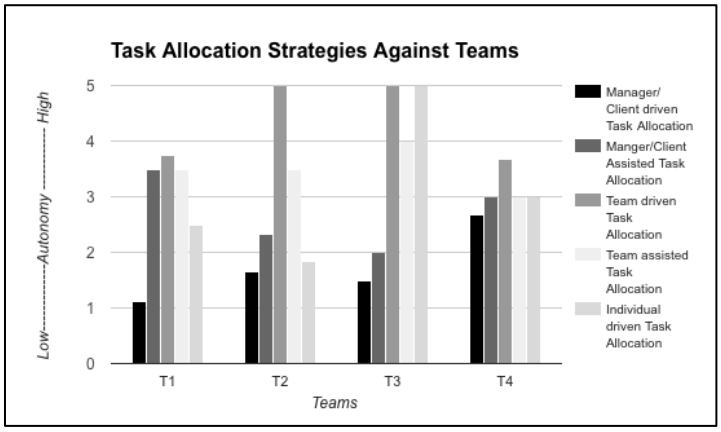

Fig. 3. Team-wise graphical representation of autonomy level

The graph in Fig.3 shows that in team T1, the manager/client/lead allocating tasks directly without individual's input is an infrequent occurrence. However, team assisting or driving task allocations is practiced often. Similarly, self-assignment without any assistance is sometimes observed. T1 shows high team/internal autonomy in selecting tasks, less individual autonomy but least external autonomy.

In T2, the manager/client/lead allocating tasks directly without individual input and self-assignment without any assistance is seen as an uncommon occurrence. But task allocation involving team members collectively is the most common behavior. T2 is showing higher internal autonomy but less individual and least external autonomy.

In T3, the team driving task allocations and self- assignment without any assistance are observed as happening most commonly, therefore leading to a high team/internal and individual autonomy. In contrast, the manager/client/lead allocating tasks directly to team members or providing assistance is an infrequent occurrence and so least external autonomy is comprehended with T3. T4 graph pattern indicates the manager/client/lead allocating tasks directly or providing assistance and self-assignment of tasks by

individuals is happening sometimes. So individual autonomy and external autonomy is witnessed moderately but team autonomy is comparatively higher, the team driving and assisting task allocations most frequently.

4.1.3 Impact of current strategy, challenges faced and improvements

To evaluate the effectiveness of the current strategy, the practitioners were asked to rate their current task allocation mechanism. Additionally, they were requested to share any potential problems they faced with their current method which could help the researcher to highlight the loopholes in this process.

Team T1 rated the current strategy as serving "well" for the project, "moderately well" for the team and "well" for themselves as individuals. Team T2 rated the current strategy as more than "well" for the team and the individual but "moderately well" for the project. T3 rated their strategy "very well" for the project, team and the individual. T4 rated current strategy to be working "well" for the team and the individual and "moderately well" for the project as shown in Fig.4.

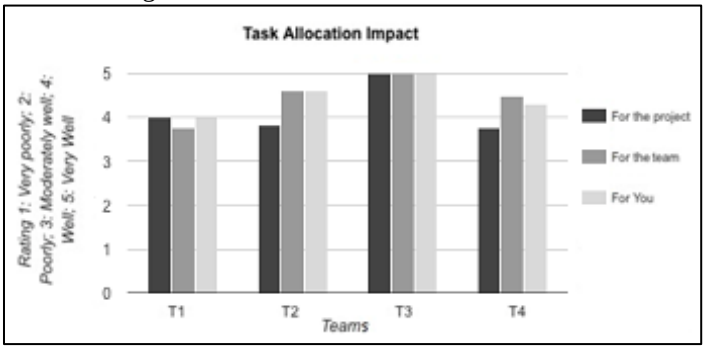

Fig. 4. Impact of the current task allocation strategies on project, team and individual

All the teams stated to be satisfied with their current task allocation strategy but shared few challenges faced with ideas for improvements.

Vagueness or missing clarity on tasks/tickets was the most commonly reported challenge faced by the practitioners as specified by one of the participants.

"I think the ticket description if it is more clear, then we could estimate it correctly… clarity on the tickets [is a problem]" ***SP6, Senior Software Engineer***

To handle such situations, the participants suggested having a technical person reviewing the tickets before being assigned and earlier discussions as an improvement to address such vague and open to interpret tickets.

"Yes sometimes actually the requirements team, they only decide, they create tickets after that if one technical person reviews and updates the ticket with technical details then it will be helpful for us" ***SP6, Senior Software Engineer***

One participant mentioned that with their current task allocation strategy, work at times is not evenly distributed.

"[My working] pair [or partner] takes advantage, most of the work I have to do." ***SP11, Senior Software Engineer***

One of the participants revealed drawback of picking tasks remotely. Since the client and the US team are co-located they were perceived to have an advantage in picking tasks, however, overall the India team was satisfied with the tasks they did as mentioned by SP1.

"Sometimes it happens... we are on remote call, client and US team are together in same room, when they start picking the tickets ...they have advantage of raising their hands they will quickly say 'hey I'm interested' ... majority of time we get the work which we are interested [in]." ***SP1, Technical Lead***

The next section presents the qualitative analysis and findings.

4.2 Qualitative Analysis and Findings

Through our thematic analysis, we identified four distinct workflow mechanisms that describe how the teams receive the work from the relevant stakeholders. These are named as team independent, team dependent, skillset/module dependent and hybrid workflow. Additionally, we found five different task allocation strategies based on how tasks were allocated within the team. We first present the team workflow mechanisms along with relevant quotes from the participants proceeded with task allocation strategies.

4.2.1 Team Workflow Mechanisms

Workflow and task allocation strategies followed by the agile software teams is the central aspect of this study. Based on the details shared by the participants while illustrating the task flow process, the most frequent team wise workflow and distribution mechanisms are summarized in Table 5. Fig. 2 illustrates the most common workflow mechanisms for Teams T1-T4.

Table 5. Summary of Workflow Mechanisms for Teams T1-T4

Team	Tasks Comes From(Mostly)	In form of(Mostly)	During(Mostly)
T1	Product Owner/ Manager in USA, part of team	User stories	Sprint planning meeting
T2	Client, part of Team in USA	User stories	Fortnightly iteration planning
T3	Product Owner in USA, not part of team, Client	High level requirements	Monthly planning call
T4	US sub-team who collaborates with client	Varies [features/high level user stories / technical tasks]	Monthly release mostly

Team Independent Workflow

For T1, Tasks comes to the team from Product Owner/ Manager in the US mostly in form of user stories during sprint planning meeting. Product owner creates high-level user stories as JIRA tickets with well-defined acceptance criteria, based on user requirements and the research work of the product team. After the arrival of tasks, the team members individually pick and break these user stories into technical tasks. Everyone including the

USA team is present in the sprint planning meeting and work allocation is done by volunteering for tasks through mutual discussions. SP1 described the flow as listed below.

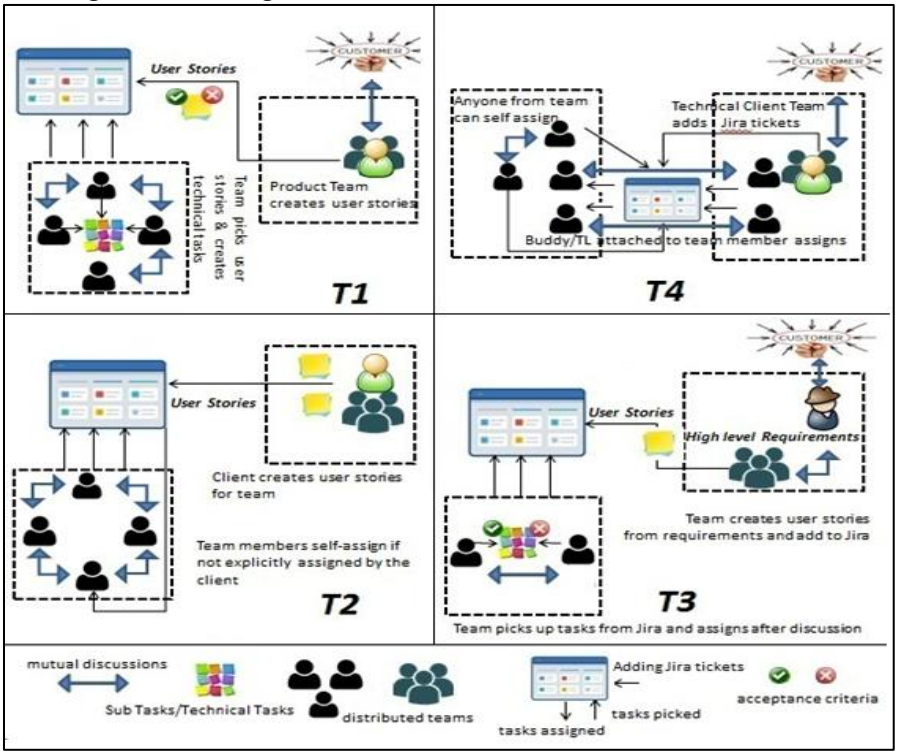

Fig. 2. Teamwise Task Allocation Mechanisms (T1: team independent workflow; T2: team dependent workflow; T3: skillset/module dependent workflow; T4: hybrid workflow)

"They bring up the user stories to us … they do all the research and bring whole description of the ticket... Everyone is in sprint planning meeting, every developer I should say and then ticket by ticket…they do not assign any name; these are all unassigned tickets..." ***SP1, Tech Lead.***

Team Dependent Workflow

Tasks come from the client as user stories during fortnightly iteration planning meeting for Team T2. Before sprint planning meeting, the client creates user stories for the team and before the meeting, the team goes through those stories and allocates the task to themselves either voluntarily or with mutual consensus. SP7 described the workflow as follows:

"Mostly user stories, sometimes features... normally client creates user stories then one day before sprint planning we go through those stories which are meant for India team and we decide and pick whatever we want to do." ***SP7, Tech Lead***

Skillset/Module Dependent Workflow

For Team T3, work mostly comes in form of high-level requirements during monthly release call by the client. These requirements are captured either verbally or through Google docs, excel sheet, Minutes of meetings, then the team or the individuals categorize tasks module-wise and break them down into user stories/sub tickets with defined acceptance criteria. SP8 illustrated the workflow process as:

"High level requirements that need to be built, based on that we plan out, create user stories, write some acceptance criteria ….based on these monthly calls we do… normally we categorize the kind of work….like one of us work on integration and all integration work going through him" ***SP8, Software Engineer.***

Hybrid Workflow

Team T4 was seen to follow multiple workflow mechanisms, but tasks typically are allocated during monthly release from USA team, who collaborates with the client. The form of the work varies from features to high-level user stories or sometimes technical tasks. For a sub-team, USA Technical client team creates tickets in Jira (a work management tool) with team wise classification, marked with a set priority and complexity level.

"Now that teams have been divided so they have to work according to the tasks that are assigned to those particular teams only so it's not like that X team can work on team Y cards" ***SP10, Associate. Tech Lead***

For other team members, work comes as features through Jira with a defined priority and release name from the USA team. As per SP11, the workflow is based on priority.

"*So the client decides the criticality and to which release these card will belong so once the lead has decided that so the pair can pick up".* ***SP11, Sr. Software Engineer***

These are selected by the Tech Lead who further breaks these features into tasks and sub-tasks and allocates to the buddy in India. As specified by SP12:

"Usually we do have features like my buddy divides into small tasks and creates cards, we start splitting tasks into small cards…" ***SP12, Assoc. Tech Lead***

4.2.2 Task Allocation Strategies

The previous section described how work arrives to the team, i.e. the workflow mechanisms. This section presents the task allocation strategies, i.e. how tasks are allocated individually within the team. Based on the workflow mechanism followed by the teams on a daily basis and specific questions asked about their task allocation strategies (e.g. see questions in section 3.1), the task allocation strategies are broadly classified in following five types and summarized below.

In **Manager/Client driven Task Allocation**, the manager/client/technical lead allocates tasks to the team members with names against the tasks as stated by practitioner below

"Nowadays I am given task by my buddy" ***SP12, Assoc. Tech Lead***

In **Manager/Client assisted Task Allocation**, tasks are allocated with some assistance from the manager/client/technical lead. As a technical lead, SP1 mentioned assisting team member with picking tasks.

"Hey xyz you should do this, let say he is new and he doesn't know I help him, pick this one because this is lesser complex". ***SP1,Tech Lead***

The team discusses and mutually decides who will do which task in **Team-driven Task Allocation.**SP6 mentioned practicing team discussions for allocating tasks.

"We are three people so mutually decide who will do [what]" ***SP6, Sr. Software Engineer***

Every team member self-assigns tasks with some assistance from other team members in **Team assisted Task Allocation** as supported by comments of SP11.

"So any of the pair can pick up [a task]". ***SP11, Sr. Software Engineer***

In the **Individual-driven Task Allocation**, tasks are self-directed i.e. selected and managed individually without any assistance from others. SP4 quoted practicing self-driven allocation.

"Mostly we volunteer it." ***SP4, Software Engineer***

5 Discussion

This section discusses and compares our research findings with prior related research and states the implications of the findings.

5.1 Comparison to Related Works

During this research study, while exploring the workflow mechanisms, it was found that all the teams exhibited their own way for requirements elicitation. For instance, T2 develops the user stories with client involvement and T4 acquires them as high-level requirements directly from the customer. As opposed to previous studies [9,10,12], this is a new aspect that identifies four mechanisms for task allocation. Although the previous literature does address diverse perspectives in task allocation but does not illustrate the teamwise workflow in agile teams. Similarly, identifying the granularity of the work, e.g. features, high-level requirements, user stories, when allocated to teams or individuals seems to be a new contribution.

Another interesting aspect identified is that for high priority tasks all mechanisms agree on a common allocation method i.e. task is directly allocated to a skilled person.

"Client or Team Lead can assign the ticket depending on that this person has specially work on this module and he has more knowledge on [it]." ***SP10, Assoc. Tech Lead***

In such situations, the mechanisms deviate from workflow being directly allocated to someone who is experienced and skilled enough to perform the task. Previous research on task allocation also highlight this scenario where a task is directly allocated to a skilled person [13,14].

This research supports that agile teams tend to show various levels of autonomy [7, 16]. Based on strategies followed by the teams, it is noticed that different teams show different autonomy levels with respect to task allocation as elaborated in section 4.1.2.

5.2 Implications

The findings from this research can benefit software engineers, project managers and researchers in several ways. This research study can be helpful for the organizations in smooth transitioning from non-agile to agile software development. The workflow mechanisms and the task allocation strategies identified in this study can help them to design their new agile team structures based on these practices. Exploring the underlying workflow of agile teams has lead us to develop a road map for agile teams, from requirements elicitation to actual task allocation.

This research study can serve as a basis for understanding other task allocation strategies and internal workflow mechanisms of agile teams. Although the four workflow mechanisms identified in this study are likely not a closed set, but they will be very helpful in understanding ways the teams receive and allocate tasks in agile teams for commercial software development.

The findings also suggest that there are some challenges faced by the agile team members. The interview questions were able to highlight the areas where they were having issues in allocating tasks. This study can be used to suggest approaches and alternatives to the practices that are causing these issues.

6 Limitations and Future Work

This section discusses limitations of this study and areas for future research. The study involved only 12 interviews from the same organization which signifies a limited dataset and context. Due to this limitation we were not able to reach the saturation point for the four workflow mechanism identified in this study. Additionally, the study collected data from at least three participants per team except for Team T3, who had only one participant and so contributed as an individual perspective. T3 was the smallest of the teams and only one person was available for interview. Furthermore, the data collected primarily reflects developers' perspectives and so some concerns may not be represented in this study.
In continuation of this study in the future, the researchers may interview more teams, involved in developing various types of software and preferably from different backgrounds for a richer dataset. An extended survey-based investigation, targeting a broader and larger set of participants would strengthen the findings, providing valuable insights into more task allocation strategies. In future researchers can continue this research to understand which task allocation strategies work better for what type of workflow mechanisms. Another aspect of this study in future can include which type of strategies can affect the timely and accurate completion of various types of projects.

7 Conclusion

The study advances the understanding of task allocation process in agile software development and explains the strategies followed by the agile teams. Clients typically provide features or high-level requirements or user stories to the agile teams, which sometimes break them down into technical tasks or sub-tasks themselves or directly allocate to team members. The team members then select them individually or through discussions with the team as a whole. Allocation of tasks usually takes place during iteration or release planning.
The findings of this study demonstrate that there is no one way of tasks allocation. It varies team wise based on what suits the completion of the work in the best possible way. But a common mechanism found with the majority of the teams is that if the priority of the task is high, then the task is allocated to the most suitable person directly. Also on

average, the practice most commonly followed is that the team members collaborated with each other and with client/manager when assistance was needed. All the software practitioners mentioned that they engage with some level of negotiations during task allocations, but this is area needs further exploration in future studies.

Acknowledgement

We are grateful to all participants who contributed to this study by sharing experiences on task allocation process and practices followed in their teams.

References

1. Pinto JK, Slevin DP (1988) Critical success factors across the project life cycle. Project Management Journal, 19(3), pp. 67-75.
2. Hoda R, Murugesan LK (2016) Multi-Level Agile Project Management Challenges: A Self-Organizing Team Perspective, Journal of Systems and Software, Vol. 117, pp. 245-257.
3. Hoda R, Noble J ,Marshall S (2008) Agile Project Management In: Proceedings of the New Zealand Computer Science Research Student Conference, pp. 218-221.
4. Hoda R ,Noble J (2017) Becoming Agile: A Grounded Theory of Agile Transitions in Practice, IEEE International Conference on Software Engineering (ICSE2017).
5. Yin RK (2012) Applications of case study research. 3rd edn. London: SAGE Publications Ltd.
6. Clarke V, Braun V (2014) Thematic analysis. In: Encyclopedia of Critical Psychology. Springer, NY, pp. 1947–1952. doi:10.1007/978-1-4614-5583-7_311
7. Principles behind the Agile Manifesto. http://agilemanifesto.org
8. Hoda R, Noble J,Marshall S (2011) The Impact of Inadequate Customer Involvement on Self-Organizing Agile Teams, Journal of Information and Software Technology (IST).
9. Lamersdorf A, Jürgen M, Dieter R (2009) A survey on the state of the practice in distributed software development: criteria for task allocation, In : 4th IEEE international conference on global software engineering (ICGSE).
10. Simão Filho, M. Pinheiro, P.R., Albuquerque AB (2015) Task Allocation Approaches in Distributed Agile Software Development: A Quasi-systematic Review.In: Advances in Intelligent Systems and Computing, Vol. 349, pp. 243-252.
11. D. K. M. Mak, P. B. Krachten (2006) Task Coordination in an Agile Distributed Software Development Environment, In: IEEE Canadian Conference on Electrical and Computer Engineering - CCECE2006. Ottawa, Canada: IEEE.
12. Imtiaz S, Ikram N (2016) Dynamics of task allocation in global software development, Journal of Software: Evolution and Process. doi:10.1002/smr.1832
13. Crowston K, LI. Q., WEI, K., ESERYEL, U. Y.,HOWISON J (2007) Self-organization of teams in free/libre open source software development, Information and software technology, Vol. 49 No 6 , pp. 564-575.
14. Crowston K,Scozzi B (2009) Bug fixing practices within free/libre open source software development teams. Principle Advancements in Database Management Technologies: New Applications and Frameworks: New Applications and Frameworks, pp.51.
15. Seaman C (2008) Qualitative Methods. In Shull, F. et al.: Guide to Advanced Empirical Software Engineering. Springer.
16. Moe NB, Dingsøyr T, Dyba T (2008) Understanding self-organising teams in agile software development. In: 19th Australian Conference on Software Engineering, IEEE. Perth, pp.76–85. doi: 10.1109/ASWEC.2008.4483195
17. Highsmith J (2009) Agile project management: creating innovative products. Pearson Education.