Becoming Agile: A Grounded Theory of Agile Transitions in Practice

Rashina Hoda
Department of Electrical and Computer Engineering
The University of Auckland
Auckland, New Zealand
r.hoda@auckland.ac.nz

James Noble
School of Engineering and Computer Science
Victoria University of Wellington
Wellington, New Zealand
kjx@ecs.vuw.ac.nz

***Abstract*—Agile adoption is typically understood as a one-off organizational process involving a stage-wise selection of agile development practices. This view of agility fails to explain the differences in the pace and effectiveness of individual teams transitioning to agile development. Based on a Grounded Theory study of 31 agile practitioners drawn from 18 teams across five countries, we present a grounded theory of becoming agile as a network of on-going transitions across five dimensions: software development practices, team practices, management approach, reflective practices, and culture. The unique position of a software team through this network, and their pace of progress along the five dimensions, explains why individual agile teams present distinct manifestations of agility and unique transition experiences. The theory expands the current understanding of agility as a holistic and complex network of on-going multi-dimensional transitions, and will help software teams, their managers, and organizations better navigate their individual agile journeys.**

***Keywords*-agile software development; transition; self-organizing; teams; management; culture; theory; grounded theory;**

I. Introduction

Despite the wide-spread adoption and practice of agile methods [38], [42], becoming agile continues to be a daunting journey for many software teams [4], [31]. Adoption barriers and challenges typically involve organizational culture [24], [43], people [13], [15], [27], process [15], [43], and tools [41], [31]. Software teams struggle with their first steps [12], [14], [29]; trying to institute effective agile practices [6]; or simply feeling 'they are not there' yet in their agile journey [20]. Successfully adopting agile methods is acknowledged to be both demanding and time consuming [4], [31].

Researchers have proposed a number of structured agile adoption frameworks [2], [3], [4], [22] based on elaborate theoretical modeling and abstraction, but with little practical validation in industrial settings. Typically, these models prescribe a simultaneous staged adoption of low-level development practices and high-level agile values. While an incremental adoption approach is supported by evidence from industrial practice [1], there is no consensus on the proposed adoption stages or their sequence across the various theoretical models.

Furthermore, the current research focus on staged, structured adoption is often misaligned with the needs of the industry looking for practical guidance on agile [31]. For example, there is no clear explanation on why some teams seem to be more 'agile' than others, even though they all claim to follow agile development practices and have introduced them incrementally. This leaves several open questions: What aspects of software development are affected during industrial agile adoptions? How do these different aspects interact in practice? How can we explain the differences in the pace and effectiveness of different software teams attempting to become agile? To answer these questions, we need a holistic understanding of what all it takes to be become a self-organizing agile team — an understanding that is based on empirical evidence and grounded in practice.

In this paper, we present a theory of software development teams transitioning to agile development. The *Theory of Becoming Agile* is based on a Grounded Theory study of 31 practitioners drawn from 18 development teams across five countries. The theory explains agile *transitions* as an ongoing, continuous, long-term transformation, rather than clearly circumscribed stages of agile *adoption*. We find that agile transitions involve complex networks of changes across five dimensions: *software development practices* transitioning from traditional to agile practices; *team practices* changing from manager-driven to team-driven; the *management approach* transitioning from driving to empowering; the *reflective practices* changing from being limited to becoming embedded as a means of guiding continuous improvement; and *culture* changing from hierarchical to open. Critically, the theory explains that teams transition along these dimensions at different rates, and so can have different levels of agile practice in different dimensions. Grounded in practical evidence, the theory of becoming agile offers both a theoretical model for further research, and practical guidance for software teams undertaking transitions to agile development.

The rest of the paper is structured as follows: related work is summarized in section II. The Grounded Theory method and its application in this study are described with examples in section III. The results describing transitions across the five dimensions are presented in section IV. The inter-relationships between the dimensions of the theory, its application and verifiability, and comparison with related work are discussed in section V. The paper concludes in section VI.

II. Related Work

A. Adoption Challenges

A recent review of agile and scrum adoption challenges reported four major themes: people, organization, project and process [21]. The people-related adoption challenges included team size [27], lack of effective communication [15], lack of customer collaboration [13] and lack of experience with agile methods [13], [43] among others. The organization-related adoption challenges included cultural mismatch with agile methods [19], [13], [43] and lack of capacity to change the organizational culture [24], [43]. The project-related challenges included project size [24], [19] while the process-related challenges included agility degree [15] and anti-patterns [43].

Industrial experience reports and empirical studies of industrial practice continue to report challenges with initial adoption [12], [14], [29], struggles with regular practice [6], and failed attempts to adopt agile techniques [20]. Challenges of agile transitions, such as those related to organizational culture, management, people, and tools continue to be reported in industrial practice [12], [31], [41]. At the same time, a number of adoption frameworks have been proposed, summarized next.

B. Adoption Solutions

A systematic literature review of agile adoption [1] identified a number of structured agile adoption approaches [2], [4], [3], [22], most being agnostic to specific agile methods and focusing on high-level agile values and properties, for example, flexibility and responsiveness.

Guided by the organization's potential and readiness for agile adoption, Sidky and Arthur [2] proposed a four-stage process to systematically introduce the adoption of a predefined set of agile practices, which has been viewed as process-heavy and inflexible [17].

Qumer et al. [3] proposed an agile adoption and improvement model (AAIM) consisting of six stages within three blocks, and an agility measurement model. The levels of the AAIM model suggest specific agile properties to be introduced, e.g. speed, flexibility, and responsiveness in level one. In another study, Qumer and Henderson-Sellers [4] proposed an agile software solution framework (ASSF) consisting of an agile toolkit aimed at facilitating a customizable and combinatory agile adoption process.

A model presented by Sureshchandra and Shrinivasavadhani [22] focuses on distributed contexts. They describe a four-stage adoption process comprising of a first evaluation stage to assesses the suitability of the project to a distributed context; then two stages inception and transition, where concepts such as self-organization and customer collaboration are introduced; and finally a steady stage, which defines the completion of the transition where teams take full ownership of work and are completely self-organized.

Wufka and Ralph [30] proposed a preliminary process theory to explain how the interplay between stakeholders (developers, management and users) drives the need for changes; the iterations between recognition and response produce the changes; and the changes take place within the interactions between the team, the development process, and the software artifacts. How well this process theory of agility explains the practical experiences of industrial agile practitioners remains to be studied.

Arguably the only empirically generated agile adoption and transition framework was proposed by Gandomani and Nafchi [40] resulting from their Grounded Theory study based on our guidelines [33], [34]. It described the desired characteristics of an applicable transition framework to be iterative, continuous, gradual and business value based; and the key adoption and transition activities as: practice selection, adaptation, assessment, retrospective meeting and adjustment, modelling a plan-do-check-act (PDCA) approach [44] to agile transitions.

Overall, prior research seems pre-dominantly concerned with: defining the adoption stages, with a focus on evaluation and introduction stages [2], [3], [4], [22]; the abstract, higher-order properties or principles of agility (e.g. responsiveness, adapting to change, speed) [3], [30]; and abstract theoretical modeling [30] in some cases leading to some practical validation [2], [3]. And yet, pre-defined, stage-based adoption frameworks may struggle to explain the acknowledged differences in the pace and effectiveness of individual agile transitions [4]. Overall, the dominant research focus on theoretical modeling seems to be mostly misaligned with the needs of the industry looking for practical guidance on holistic agile transitions [31].

III. Research Method

We adopted the Grounded Theory (GT) method [8], [9] and its various procedures for data collection, analysis, theory formulation and reporting. GT allows the researcher to uncover the primary concerns of the participants through identification of common patterns and themes that emerge from the constant comparison of data across participants at increasing levels of abstraction [23], [33], [34]. The distinguishing feature of GT is the absence of a clear research hypothesis upfront, rather the researcher attempts to uncover the main concern of the participants in the process. In this case, the focus was on the experience of transitioning to agile development and the associated challenges and strategies in real-world settings. GT was employed as the research method for several reasons:

(a) Agile methods emphasize people and interactions and GT is well suited to study human and social aspects;
(b) GT facilitates research on relatively less investigated areas and the topic of agile transitions requires more empirical evidence;
(c) GT is increasingly being employed to study agile software teams as it facilitates the investigation of social and human aspects [5], [16], [18], [34], [35]; and
(d) Being grounded in empirical evidence, GT allows the researcher to try and reconcile research rigor with industrial relevance, an issue highlighted as the 'grand challenge of agile research' [31], [38].

The following sub-sections present descriptions of the GT procedures applied in this study, including examples to elucidate their application.

A. Data Collection

We conducted semi-structured interviews with 31 agile practitioners from New Zealand, Australia, USA, India, and Portugal and observed development teams in situ at some participating companies in India. Participants with industrial agile experience were recruited through a general call for participation posted on popular agile user groups, social networking sites and local communities, such as the Agile Professions Network, Auckland and the Agile Software Community of India (ASCI). A variety of roles and designations were covered in order to achieve a rounded perspective.

Table 1 presents the participant demographics. The first column refers to participant numbers P1-P31 to maintain participant anonymity as per the human ethics guidelines. The second column lists their roles, e.g. developers, scrum masters, testers, business analysts etc. The third column refers to the project domain, e.g. healthcare, banking, shipping, marketing, and development; while the remaining columns list their years of total professional experience (TX), agile experience (AX), organizational sizes (OS), team sizes (TS), country of work (CN), and agile methods used (e.g. scrum, XP, combinations, under MD).

The interviews lasted approximately an hour on an average and were conducted at the participants' workplaces (e.g. in New Zealand and India) or through Skype video calls (e.g. for participants in the USA, Australia, and Portugal). A set of guiding interview questions were designed to cover three main areas: (a) professional background: e.g. *please tell me briefly about your professional background*; (b) questions related to self-organizing practices: e.g. *do you think your team is self-organizing and why/why not? Please give some examples*; and (c) questions related to work allocation: e.g. *how does task assignment work in your team?* However, the interviews were semi-structured to allow for the participants' main concerns around becoming agile to emerge.

B. Data Analysis

We employed Grounded Theory's data analysis and synthesis procedures i.e., open coding and constant comparison method to synthesize data from the interviews by identifying the patterns in the dataset. The word *open* here refers to keeping an open mind when analyzing the data i.e. not being biased by previous literature and/or personal researcher experiences; and *coding* refers to the task of data analysis [8], [9], [10]. Thus, open coding involves thoroughly analyzing the data to capture as many key points and concepts as possible. Details of open coding and other GT procedures can be found in a dedicated paper on the topic [33]. However, to explain these procedures as applied in this study, we present an example of working from the raw data of an interview transcript to the findings for one of the categories: *management approach.*

First, we obtained the *key points* from the interview transcripts. Key points are the summarized points from sections of the interview [37]. We then assigned a *code* to the key point. A code is a phrase that summarizes the key point in two or three words. One key point can lead to several codes.

TABLE I
PARTICIPANT DEMOGRAPHICS

P#	Role	Domain	TX	AX	OS	TS	CN	MD
P1	BA	Marketing	6	2	XL	6	USA	S
P2	PM	Business	10	6	M	8	NZ	S/K
P3	Dev	Health	2	2	XS	5	PGL	S/XP
P4	BA	Banking	9	3	L	10	USA	S/XP
P5	S. Dev	Business	4	3.5	M	8	NZ	S/K
P6	Dev	Health	3	1	L	6	IND	S
P7	QA	Banking	6	2	L	6	IND	S
P8	PM	Business	14	1	L	7	USA	S
P9	S. Dev	Banking	5	3	L	8	OZ	S/XP
P10	SM	Utilities	8	2.5	M	7	NZ	S/K
P11	PM	Utilities	10	2	M	7	NZ	S/K
P12	Dev	Freight	3	2	L	10	IND	S
P13	PM	Business	10	1.5	L	14	IND	S
P14	Dev	Finance	3	1.5	L	8	IND	S/L
P15	ERP	Finance	3	2.5	L	12	IND	S/L
P16	S. Dev	Marketing	10	5	XL	6	USA	S
P17	Dev	Finance	3	2	L	8	IND	S
P18	Dev	Finance	3	2	L	8	IND	S
P19	Dev	Freight	3	1.5	L	8	IND	S
P20	Dev	Marketing	3	2	XL	8	USA	S
P21	S. Dev	IT	5	4	XS	10	IND	M
P22	S. Dev	IT	5	4	XS	10	IND	M
P23	Dev	Telecom	2.5	1	XS	8-10	IND	S/XP
P24	Dev	Telecom	3	1.5	XS	8-10	IND	S/XP
P25	S. Dev	Telecom	6	1	XS	8-10	IND	S/XP
P26	Dev	Telecom	6	4	XS	8-10	IND	S/XP
P27	Dev	Telecom	1.5	0.3	XS	8-10	IND	S/XP
P28	S. Dev	Food	5.5	5.5	XS	5	IND	K
P29	S. Dev	Food	5	5	XS	5	IND	K
P30	S. Dev	Food	2.5	2.5	XS	5	IND	K
P31	Dev	Food	2.5	2.5	XS	5	IND	K

P#: participant number; Dev: developer, PM: project manager, SM: scrum master, QA: quality analyst, BA: business analyst, ERP: enterprise support; TX: total experience (in years); AX: agile experience (in years); OS: organizational size (XS<100; S<1K; M<5K; 100K<L<10K; XL>100K) TS: team size; Con.: country (PGL=Portugal; OZ= Australia; IND=India); MD: method (S=Scrum; K=Kanban; L=Lean, M=Mixed Methods.)

Raw data: *"I think it [self-organization] is fifty-fifty. So my personal default position with my team is to try and allow them to self-organize but there is a need for me right now to monitor those activities that they don't revert to practices that are compulsive. So my team is still transitioning..."*
Key Point: *encouraging empowerment but monitoring during transitioning*
Code: *encouraging empowerment (transitioning)*
Code: *monitoring teams (transitioning)*

In the above example, *encouraging empowerment* and *monitoring teams* were codes derived from the raw transcripts, where the former summarized the idea of encouraging autonomy in the team – a self-organizing trait; and the latter capturing the idea of monitoring the team activities to avoid

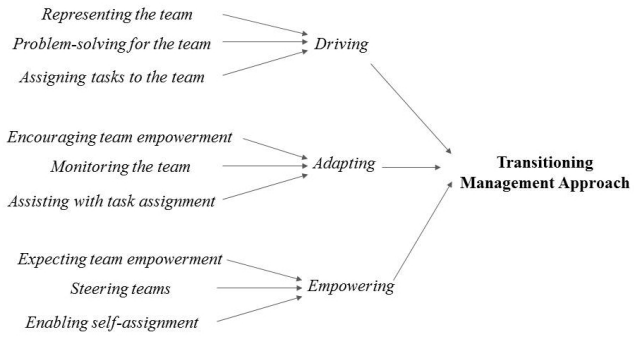

Fig. 1. Emergence of category *Transitioning Management Approach* from underlying codes and concepts.

reversion to traditional ways of working – a more traditional management approach. Particular aspects of a code can be captured in brackets as *properties*. 'Transitioning' was captured as a property of the codes. In other words, it reminds us that the original context of the raw data was referring to the fact that the teams were in the process of transitioning into agile methods and self-organization. The codes arising out of each interview were constantly compared against the codes from the same interview, and those from other interviews. This is GT's constant comparison method [8]. Using the constant comparison method, we grouped these codes to produce a higher level of abstraction called *concepts*. In this example, the concept that emerged was '*adapting*' which captured how managers monitored and assisted their team at times while at other times tried to encourage empowerment, adapting to the team's response to increasing levels of autonomy.

Concept: *adapting (to team's level of autonomy)*

Other concepts arising in a similar manner included *driving* referring to the management approach that encompassed the traditional traits of driving team decisions; and *empowering* referring to the managers encouraging and expecting high levels of team autonomy from more experienced teams.

The constant comparison method was repeated on the concepts to produce a third level of abstraction called *categories*. The *driving, adapting, and empowering* management approaches were seen to vary with the teams' relative level of agile experience such that managers on relatively new agile teams were seen to practice a driving approach; those with more autonomous teams were seen to practice an empowering approach; and those with teams becoming increasingly autonomous were seen to practice the adapting approach. Thus, the category derived to capture these transitions was *transitioning management approach*. Figure 1 depicts how the category *transitioning management approach* emerged from some of the underlying codes and concepts.

Category: *transitioning management approach*

Similarly, the other four categories were identified, i.e. *transitioning: software development practices, team practices, reflective practices and culture practices* as the dimensions of change in agile transitions.

The final step of GT's data analysis process is theoretical coding which involves conceptualizing how the categories relate to each other, and formulating a set of inter-related hypotheses to be represented as a theory [10], [11], [23]. Glaser [11] suggests several common structures of theories, referred to as theoretical coding families, such as: the six Cs (contexts, causes, consequences, contingencies, conditions, and covariance); degree; and process. Further details of open, selective, and theoretical coding and other GT steps such as memoing and sorting can be found in GT's seminal texts [10], a recent review [23] and a dedicated paper [33] on the application of GT in software engineering. The theoretical coding family best suited to our findings was the *process family* [11] which enables the findings to be represented a process; in this case, the process of becoming agile.

IV. Dimensions of Becoming Agile

In this section, we describe the five main categories identified in this study as the five dimensions, *development practices, team practices, management approach, reflective practice*, and *culture*, along with grounded examples and quotes from the underlying data. The hypotheses (i.e. inter-relationship between the categories) and the formulation of the overall theory is described and discussed in section V.

A. Transitioning Software Development Practices

Fig. 2. Transitioning Software Development Practices

Teams new to agile methods were getting acquainted with basic agile development practices such as the iterative and incremental delivery model. They often began by retaining many of their earlier ways of working. Often, individuals were perceived to be change resistant, although the resistance typically dampened over time:

"Obviously challenges are there and I can say people are resistant to changes...for example not everybody is willing to attend the meetings often...so it takes a while for them, for the rhythm to set...At the initial stages it is difficult but as time passes it gets easier." – P15, ERP, India

As they gained more experience, teams seemed to grasp the finer details of the agile development better, and moved away from their earlier practices and toward agile practices:

"Transition happens slowly. Initially they don't understand anything, sprint planning, demo, if demo [is] required? How would I point [estimate] the story? What will happen if I put 3 [points]? We need to explain [to] them. Initially learning, then we start applying." – P25, Senior Developer/Tech Lead, India

Teams that had progressed further along in their agile practice were seen to use agile artifacts and practices such as pair programming, user stories and tasks, testing, frequent releases, daily stand up meetings, and in some cases, retrospectives.

Overall, the software development practices transition from being mostly *traditional* to a *hybrid* of traditional and agile to finally, mostly *agile* practices as teams became accustomed

with agile methods. Significantly, these were not the only changes observed; transitions were identified in team practices, management approaches, reflective practices, and culture.

B. Transitioning Team Practices

Fig. 3. Transitioning Team Practices

In relatively new agile teams, individuals were seen to be risk-averse, preferring a *manager-driven* approach over a team-driven one. For example, team members did not seem to mind the delegation of tasks; rather they felt safer when management assigned tasks, as the team members did not have to volunteer or take decisions on their own.

"We really don't feel that we get delegated [tasks], but we feel one less job." – P13, Project Manager, India

Project and task management practices such as requirements specification, prioritization, and clarification were mostly driven by managers in collaboration with the customers and did not involve the team. In some cases, the managers and subject matter experts completed the effort estimations in collaboration with the customer while the development team focused on the task implementation.

This was in stark contrast to the treatment of the same activities in some other teams. As they gained experience, some teams displayed mixed traits such that project management practices were at times manager-driven and at other times incorporated the team to a greater extent.

"The estimation of tasks are done by [the] tech[incal] lead and sometimes I do it...we try to get the team to reach consensus on estimation, [this] is an area where we could be better..." – P2, Project Manager, New Zealand

Task assignment in such teams also took the shape of assisted assignment where team members were regularly assisted by their technical leads or project managers in assigning tasks with a view to improve individual autonomy over time.

"Here we get assistance from the scrum master on how to assign tasks like he discusses with the teams and explains on the effort, functions and time factors...Once we are confident we can self-assign ourselves in future." – P19, Developer, India

Sometimes the task assignment varied between self-assignment and manager-driven delegation, depending on how willing the individuals were to accept and assert autonomy.

"It [task assignment] is a bit of both. Like sometimes "Hey, [tech lead] could you look into that" and sometimes we go forward check it and decide how to split them up...People are starting to take their responsibilities..." – P5, Senior Developer, New Zealand

We found examples of project managers encouraging increased autonomy in task assignment. For example, P2 added elements of cross-functionality to the team's key performance indicators (KPI) in order to motivate the team members to attempt cross-functionality via task assignment.

"And I actually set a formal goal for the KPIs...You should be operating outside of your comfort zone and taking on big development even though you are a good database guy." – P2, Project Manager, New Zealand

Thus, these teams with increasing experience with autonomy were seen to be largely *manager-assisted*, with managers moving between driving the team practices on some occasions and following the team on others.

In contrast, the team practices in the teams most experienced with autonomy were largely *team-driven*, including self-assignment:

"Almost all tasks are exclusively self-assigned." – P10, Scrum Master, NZ

The importance attached to self-assignment was such that it was seen as a defining characteristic of autonomous, self-oganizing agile teams.

"Yes, I consider my team to be self-organizing...the scrum master doesn't force the member of the team saying you take this task and you take this task etc...Its open to the team that they come forward as they know their expertise so we take our own tasks..." – P7, Tester, India

Similarly, most of the project management practices such as requirements specification, clarification, prioritization, and estimation were also team-driven.

"It is just the developers...I do [estimate] occasionally depending on the project where I need to incorporate some decisions. I know my decisions are off...my team tends to estimate a little bit better." – P10, Scrum Master, NZ

"We don't wait for them [team leads]...We directly contact the client if they are online...I think we are now used to it." – P30, Senior Developer, India

Thus, we found that as the teams became more receptive to increasing autonomy the team practices, such as task assignment and estimation, changed from being *manager-driven* to becoming *manager-assisted* and eventually *team-driven* over time.

C. Transitioning Management Approach

Fig. 4. Transitioning Management Approach

Managers also must transition to agile development, both leading and responding to transitions in their teams. We use the term 'manager' here to refer to people in a variety of management roles including scrum masters, project managers, subject matter experts (SMEs) and team leads.

Initially, managers were driving requirements elicitation and clarification, task assignment, and problem solving on behalf of the team. They were often seen to be *driving* customer collaboration: eliciting requirements from customers and seeking clarifications on behalf of the team.

"He [manager] is the first point of the contact between product owner, dev[elopement] team, scrum master and the stake-holders. The client will be sharing anything only to the

manager [as] he may not know the development team." – P6, Software Engineer, India

Furthermore, the manager was looked up to as the problem-solver, motivating the team as they transitioned into agile methods:

"...we just leave the problem to the SMEs where he finds a way to alter the plans depending on the degree of the challenge...As we are the starters [pilot team] for agile [methods] we tend to get motivated a lot from the managers. That has driven us to greater extent." – P13, Project Manager, India

As teams gained experience with increasing levels of autonomy, the managers were seen to change their management approach to *adapting* between traditional monitoring activities and empowering traits of agile management.

"I think it [self-organization] is fifty-fifty. So my personal default position with my team is to try and allow them to self-organize but there is a need for me to right now monitor those activities that they don't revert to practices that are compulsive. So my team is still transitioning..." – P2, Project Manager, New Zealand

Managers acknowledged the changing needs of their teams and their tendency to revert to old ways of working and in practice, adapted their management style to the team's changing conditions.

"My team is very easily distracted by what is going on outside the project...that is where unfortunately I have to sometimes revert to classic style management...command and control mode" – P2, Project Manager, New Zealand

In teams that were more accepting of autonomy, managers were seen to practice an *empowering* style of management.

"Our project manager enables and empowers the team to do their tasks. She is the one who clears operational or process related blocks that hinder [the] completion of a task." – P16, Senior Developer, USA

The managers perceived their role to be one of a supporter, expecting autonomy from their team, and steering the team in the right direction.

"Well just the usual kind of coffee maker you know. I mean effectively in a scrum team that's what you do, you support people and you encourage them to take responsibilities...you can't neglect or abandon [them] but you need to push them in the right way." – P10, Scrum Master, NZ

They often perceived themselves to be the unseen force that guided the team but whose absence did not stall things as the team drove the decision making.

"For me self-organizing team is something that wouldn't notice if I disappeared (laughs)." – P10, Scrum Master, NZ

Since the manager was no longer driving the team practices, the empowering management approach encompassed other aspects such as encouraging teams, removing external impediments, keeping the team abreast of progress and changes, and helping the team focus by screening them from disturbances.

"Even if you are self-organizing team you need someone who cuts the interferences from the outside." – P9, Senior Developer, Australia

An empowering management approach also meant that the managers practiced subtle forms of authority such as making their presence known to instill a sense of responsibility among the team members.

"I think authority [is] maybe required but that's for isolated circumstances...stuff like testing a task is tedious for the people as they lack experience in it...I have a board near my desk so when people come up to the board looking for a task so I can let them know that I actually watch them when they pick up a testable task." – P10, Scrum Master, NZ

In stark contrast to the *driving* management approach where managers represented their relatively new agile teams to the customers, managers in self-organizing teams expected direct team-customer collaborations, stepping away from being a mediator and giving over control to the team.

"I do not interfere in the relationship between my team members and the clients as I don't want to jeopardize the communication between them." – P10, Scrum Master, NZ

The transitioning management approach corresponding to the level of team autonomy was well summarized in this quote by one of the participants:

"I think if the team is experienced in agile, then he [the manager] has to enable team members to self-assign tasks. But if team is pretty new to agile, he has to guide them in following the agile methodology. He should try to remove blocks whenever possible. He should motivate more and appreciate when things are done in the right way." – P20, UI Developer, USA

Thus, we found that as the teams gain more experience in practicing agile and become more receptive to autonomy, the managers changed their management approach from *driving* the team practices to *adapting* to dynamic team conditions and then to *empowering* the teams.

D. Transitioning Reflective Practices

Fig. 5. Transitioning Reflective Practices

Reflection and learning are two key principles guiding agile teams and are manifested in practices such as retrospectives and training as well as in informal or personal endeavours. We discovered a marked difference in these practices and the attitude of individuals in relatively new agile teams versus those with more experience.

In particular, new agile teams were *limited* in their reflective practices. New teams were strongly implementation focused and while learning opportunities through trainings were available to new teams, such learning-focused tasks were often seen as time consuming activities, intruding on the software implementation time.

"We also had times where a whole team lacks in one specific area of the project. It is not a big deal as we had to go through training sessions to understand and work but the problem is time consumption." – P13, Project Manager, India

As teams gained more experience, the learning and reflection became more *focused* on improving agile practices and working toward becoming more self-organizing.

"We are now working on process improvements. Previously we did not have planning poker, but we introduced it." – P25, Senior Developer, India

In teams with more agile experience, learning and reflective practices were more commonly prevalent and strongly *embedded* in regular practice. For example, while retrospective are a standard scrum practice, we noted that very few new teams performed retrospectives, while nearly all the experienced teams regularly held retrospectives to drive continuous improvement.

"So after completing the tasks we have a retrospective meeting on what has gone wrong and right and it serves as inputs for our next iteration." – P9, Senior Developer, Australia

In some cases, we also found evidence of reflective practice on an organizational level, intentionally customizing the agile method to their specific contexts and constraints, exhibiting a wider reflective mindset:

"We have hired an experienced scrum expert to overlook the process. After every release, we try to see what went wrong and what went right and make changes in the next release to avoid issues." – P16, Senior Developer, USA

In other cases, learning opportunities were created in the form of organization-wide initiatives. How well such opportunities were harnessed by the teams and individuals varied.

"There are many initiatives from organisation as well...Somebody who just learned new technology, he just speaks on Saturdays once a month. First half is theory and second half is hands-on...everybody would come and learn." – P22, Senior Developer, India

Overall, new agile teams tended to be highly *limited* in their learning and reflective practice, focusing more on implementation. With experience, the reflective practices became *focused* on driving agile practice improvements and in some cases, became effectively *embedded* in the team's and individuals' regular practice and in the wider organizational mindsets.

E. Transitioning Culture

Fig. 6. Transitioning Culture

Organizational culture is often cited as one of the most challenging factors in agile adoption [19], [13], [24], [43]. We found evidence to suggest that it is not simply the organizational culture at large, but also the team culture and individual traits, that dictate the work culture of a given team. In other words, different teams within the same organization, i.e. with similar organizational culture, could have significant differences in their levels of team and individual autonomy and in the other dimensions (e.g. reflective practices). We define this combined effect of organizational, team and individual culture as the operational culture or simply, *culture*.

Some teams worked within *hierarchical* organizational cultures, where information flow and operational decision making was passed down from senior management and customers to individual team members, typically via subject matter experts or tech leads. Similarly, issues arising, such as missed deadlines, were relayed from the team to the managers and then to the product owner who in turn informed the clients.

"...in case if we are unable to deliver [on time] we will document the reason (software failure, resources etc.) ... After getting the approval from the managers we will follow up the same to the product owner who in turn provides necessary explanation to the clients." – P6, Software Engineer, India

Other teams were seen to exist in organizational cultures that were *evolving* away from a traditional, hierarchical culture and towards a more open and inclusive culture. Such teams were mostly able to communicate and collaborate directly with their customers, however, a main point of contact or a customer representative was usually also present. Some other teams had highly *open* organizational cultures marked by direct lines of communication and a clear absence of hierarchy.

"Also there is no defined communications hierarchy or process defined." – P16, Senior Developer, USA

How effectively the teams harnessed this freedom depended on the individual teams and the individuals within a team, however.

"...we can organize meeting with the clients ourselves. It is purely upon the individual to get the work done on time." – P1, Business Analyst, USA

An absence of hierarchy in decision-making was similarly present in open organisation cultures. More autonomous teams were able to contribute to decision making, controlling the amount and type of work they committed to within an iteration; and deciding whether or not to take on any additional work requested by the customers without involving the management:

"So we can decide whether we can make this addition or to reject it...This is not done by any external influence but it is purely us." – P7, Tester, India

Thus, the operational culture of the teams was found to be largely *hierarchical* for some teams and *evolving* towards becoming more *open* in others, with increasing levels of autonomy afforded by the organizational culture; and accepted and asserted by the team and individual cultures.

V. Hypotheses and Application

A grounded theory is more that just a set of descriptive categories: it should also describe the key relationships between those categories, i.e. a set of inter-related hypotheses [11], [23]. We have presented the main categories of the *grounded theory of becoming agile* as a network of on-going transitions across the five dimensions of: *development practices, team practices, management approach, reflective practices*, and *culture* over time. In this section we describe the relationships between those categories, i.e. the hypotheses, and analyse how changes in one dimension influences changes in other dimensions. By considering progression along each dimension,

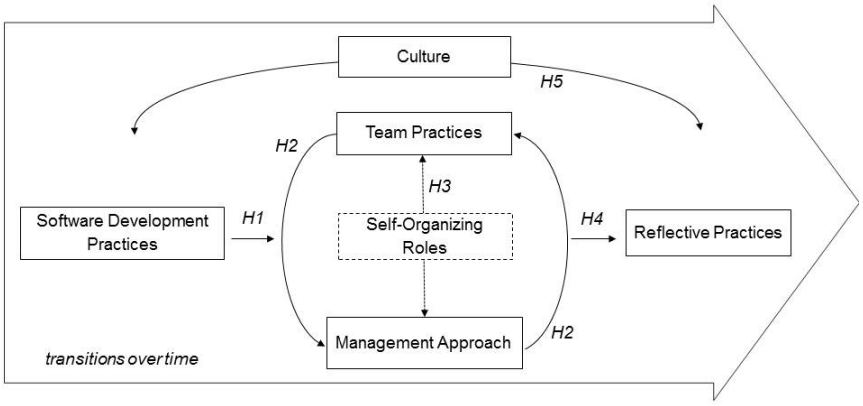

Fig. 7. A Grounded Theory of Agile Transitions as a Network of on-going Transitions across Five Dimensions: Software Development Practices, Team Practices (TP), Management Approach (MA), Reflective Practices, and Culture; with self-organizing roles [34] enabling the TP and MA transitions.

we demonstrate how the theory can be used to identify the strengths and weaknesses of a team's agile practice, and explain how the theory can be applied by actual teams. Finally we outline the threats to validity and verifiability.

A. Relationships Between Dimensions

Figure 7 shows the dimensions in the theory (within solid boxes) as described in the previous section and their inter-relationships between dimensions (hypotheses H1-H4, represented by arrows). We now discuss these hypotheses in more detail:

H1: *The transition of a team's software development practices from traditional towards agile is necessary (though not sufficient) for the changes in the team practices and the management approach to occur.* A team's transition to agile development typically begins with the core agile software development practices: pair programming, unit testing, etc. Changes in other dimensions such as the team and management practices follow changes in development practices. This is supported by the fact that all the changes in the team practices and management approach were observed in the context of teams who were further along in their transition to the agile development practices.

H2: *The transitions in the team practices and the management approach tend to reflect and adapt to each other.* The success of a particular management approach, e.g. an *empowering* approach, is incumbent upon the team practices transitioning in the same direction, e.g. becoming *team-driven*. The different team practices and management approaches continue to oppose or support each other until an equilibrium of sorts can be maintained such that a majority of the team practices support the management approach and vice-versa; and together they transition towards agile development within autonomous self-organizing teams. A self-organizing agile team is supported and maintained by a number of particular team roles [34]. Teams start off with managers playing many of these roles, e.g. mentor, coordinator, translator. Over time these are passed over to, and taken up by, the team as the management style transitions from *driving* to *empowering* and the team practices change from being *manager-driven* to *team-driven*. Thus, the self-organizing roles are the mechanisms by which the team practices and the management approach transition towards self-organization.

H3: *Transitions in team and management practices are necessary (though not sufficient) for changes in the team's reflective practices.* Reflective practices were effectively embedded only in teams whose development practices, team practices, and management practices had already transitioned significantly. Effective and regular reflective practice at a team and individual level can be considered a higher-order practice attained by highly autonomous, self-organizing teams.

H4: *All changes are influenced by a combination of the organizational, team and individual culture.* A shift away from a hierarchical culture toward an open culture supports the other four dimensions. Since a change in the organizational culture involves multiple stakeholders across the organization, not all of whom may perceive benefits in adopting agile or even be directly related to software development, this dimension is arguably the most challenging to change. On the other hand, an initial organizational culture that is itself more agile will effect the individual and team cultures and their transitions.

The theory depicted in Figure 7 is not a linear progression of events for software teams to become self-organizing agile teams. Rather, the theory describes a complex network of multi-dimensional changes that occur over time as a team transitions. A new team embarking on their agile journey can expect these transitions: we hypothesize that teams could choose to be guided by the theory to transition more efficiently and effectively. Next, we present an example of its application.

B. Applying the Theory to Explain the Uniqueness of Agile Transitions

Agile transitions are a set of relevant changes across a multi-dimensional network. The unique configuration of any software team on this network and their progress along the five dimensions can explain why individual agile teams present distinct manifestations of agility and unique transition experiences.

To elucidate the uniqueness of agile transitions using our theory, we present an example of applying of our theory to

map two specific teams that participated in this study: Team T1 represented by P23-P27 and Team T2 represented by P21-22, both from the same organization. Figure 8 visualises the unique position of these two teams along the five dimensions in a radar diagram. (A similar mapping approach was recommended by Boehm and Turner, although mapping against a very different model [7].) The innermost level represents *low* (L), the central level represents *medium* (M), and the outermost level represents *high* (H) levels of progress along each dimension, based on the descriptions of each dimension in Section IV.

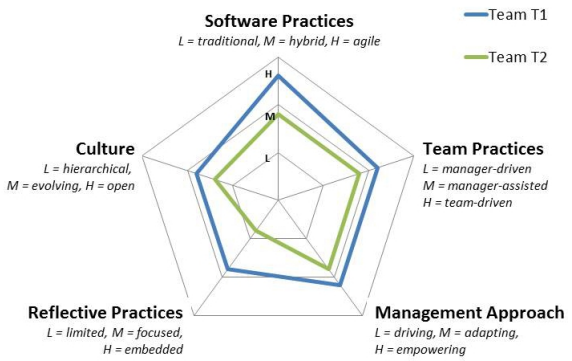

Fig. 8. An Example of Applying the Theory of Becoming Agile to Map the Uniqueness of Agile Transitions across Five Dimensions using samples teams T1 and T2.

For example, the 'low' for team practices refers to manager-driven practices, 'medium' refers to manager-assisted practices, and 'high' refers to team-driven practices (as shown in Figure 3 and described in Section IV.B). The other dimensions can be evaluated similarly against their respective definitions. In this exercise, we mapped the agility of T1 and T2 against the theory's dimensions, by observing and discussing their practices with the team, and then comparing them with the descriptions of each dimension in Section IV.

Team T1 Transitions*:* The *software development practices* of this team rated highly according to that dimension. The development practices were predominantly *agile*: two-weekly sprints, daily stand-ups, sprint planning meetings, and the use of planning poker for team-based estimations were reported by multiple team members. The *team practices* dimension is classified as medium, that is *manager-assisted*, because user stories were assigned by managers to the team lead, but the individual team members were free to select from the pool of technical tasks associated with the user stories, using JIRA as a tool. Similarly, the *management approach* was also medium, working in tandem with the team practices, with the manager *adapting* to the evolving needs of the team: encouraging autonomy while maintaining a level of control through direct code reviews and quality checks.

Seen across the *reflective practice* dimension, this team had a positive attitude to learning and *focused* on using retrospectives to some extent. Finally, the overall *culture* was also medium, i.e. *evolving*, reflecting the combined influence of a reasonably high individual and team culture of openness with an organizational culture that was evolving.

Team T2 Transitions*:* The *software development practices* of Team T2 were a *hybrid* of agile and their previous practices (i.e. medium), as the team was in the process of fixing its iteration length between two and three weeks, conducting peer-reviews, and only performed pair programming when they considered it was especially necessary. The *team practices* dimension were also medium, as members asserted autonomy in certain areas (e.g. estimations) but were largely *manager-assisted* in others. For example, self-assignment was limited to exchanging tasks and helping peers complete their tasks, as compared to Team T1 who generally self-assigned tasks.

"Yes we can [self-assign]. If developer have some tasks and he is stuck and he couldn't finish and at that time, we could pick that task." - P22, Senior Developer, India

The *management approach* was *adapting*, involving some level of monitoring as members were asked to justify the time spent on some occasions, similar to Team T1's management approach. Team T2 was mostly implementation-focused, however, and their *reflective practices* were rather *limited*.

It is interesting to note that the *culture* of the two teams was different, even though both are part of the same organization. This is supported by the empirical evidence from the two teams which suggested that the actual influence of the organizational culture varied with the level of autonomy asserted by different teams and individuals in practice. In other words, while the external environment was largely similar for both the teams, how much the organizational culture influenced them depended on their respective team and individual cultures. Thus, Team T1 which exhibited higher individual and team autonomy was able to enjoy a somewhat more open culture than Team T2.

Software teams, managers and agile coaches can use the theory of becoming agile to assess their position along the five dimensions as described in the above two team examples and illustrated in Figure 8, and so track their transitions over time in order to motivate continuous improvement.

C. Comparing to Related Work

Prior related work pre-dominantly comprises of structured, staged frameworks for agile adoption [2], [3], [4], [22]. Some of these, for example [2], recommend a linear progression across set levels of agility and have faced criticisms on account of high complexity, overheads, and low flexibility [17]. Others, for example [3] and [4], have been seen as mostly abstract, lacking concrete details of potential industrial application [40]. The framework proposed by Sureshchandra and Shrinivasavadhani [22] is exclusively focused on the specific context of distributed teams and is therefore limited in its application. The Grounded Theory based framework proposed by [40] adopts a Scrum approach to generic *'agile practices'* adoption, working through a handful of practices iteratively. Based on data collected mostly from managers and administrators, it can be seen as an abstract organizational framework to orchestrate planned agile transformations. Our theory, on the other hand, differs from these approaches in that it:

- explains agile *transitions* as an ongoing, continuous, long-term transformation, rather than circumscribed stages of agile *adoption* as presented in [2], [3], [4];
- explains the multi-dimensional nature of this complex journey across five different dimensions instead of along a single, linear axis of a set of generic *agile practices* [40]; And unlike any other framework:
- emphasizes a key distinction between the efforts to encourage autonomy (via the *management approach*) and the actual acceptance and assertion of autonomy (via the *team practices*) and describes the self-organizing roles [34] as a means to achieve the desired dual shift in perspectives.
- shows that reflective practices are higher-order practices, and so the last to transition.
- suggests that the influence of *culture* on agile transitions is a combination of the organizational, team and individual cultures.
- explains that different teams progress along the five dimensions at different rates, thereby explaining their distinct manifestation of agility and unique transition experiences.
- provides a theoretical model for future research, and,
- offers practical guidance to agile teams to self-assess their agility and track their transition.

D. Verifiability and Threats to Validity

Any grounded theory is limited in that it will be a mid-ranged substantive theory, applicable to the contexts studied [9], [10], which in turn are dictated by access to research participants. In order to obtain willing participants, we found it essential to guarantee that all identifying details — not just of the individual participants, but also their companies, and third-party clients — would be kept confidential to the researchers, under human ethics guidelines governing this study.

While the aim of a GT study is to generate new theory and not to test existing ones [9], [23], [25], [33], the verifiability of the theory can be inferred from the soundness of the research method, and from evidence that the theory is derived from the underlying data by means of that method [33]. This is why we have explained our study procedures in depth in Section III, and why the presentation of the theory, particularly the core categories in Section IV, contains quotations from our participants (albeit excerpted) and in some cases, descriptions of our observations of development projects. These details make evident how our theory fulfills the standard GT evaluation criteria: the generated categories *fit* the underlying data (see example in Figure 1); the theory is able to *work* (i.e. explain the participant's main concerns, see application in Figure 8); the theory has *relevance* to the domain, i.e. agile software development; and is *modifiable* with new data [10], [23], [33]. In particular, while GT studies typically do not claim generalization, the resulting theory should be *modifiable* in other contexts, making the GT method one of the most agile research methods available [33]. What this means is that we do not claim this theory to be absolute or final. We welcome extensions to the theory based on unseen aspects or finer details of the present dimensions or potential discovery of new dimensions from future studies.

VI. CONCLUSION

In this paper, we present the Theory of Becoming Agile, which explains how development teams transition to agile practices. The theory is the result of a Grounded Theory study involving 31 agile practitioners from 18 teams across five countries. Rather than a single change, or even a staged progression along a linear axis, our Theory of Becoming Agile considers an agile transition to take place within a multi-dimensional network of on-going changes in different areas of practice. We identified five dimensions of transitions as follows:

- As teams gain experience in agile software development, their *software development practices* transition from *traditional* → *hybrid* → *agile* practices.
- As teams gain experience with accepting and asserting autonomy, their *team practices* transition from *manager-driven* → *manager-assisted* → *team-driven*.
- As managers gain more experience with agile development, their *management approach* transitions from *driving* → *adapting* → *empowering*.
- As the teams gain experience across the agile software practices, team practices, and management approach, their *reflective practices* transition from being *limited* → *focused* → *embedded*.
- As the organization offers increasing autonomy (and the team and individuals accept that autonomy) their *culture* transitions from being *hierarchical* → *evolving* → *open*.

We also explained the inter-relationships between these dimensions:

- A transition in *software development practices* from traditional to agile cascades to the other transitions.
- Transitions in the *team practices* and *management approach* tend to reflect and adapt to each other, moving towards self-organization.
- The above transitions are necessary though not sufficient for a transition in the *reflective practices*.
- All transitions are influenced by a combination of organizational, team and individual *culture*.

Crucially, our theory explains that teams do not progress along these dimensions at the same pace. This theory explains why some teams are more 'agile' than others, even when both teams are practicing the same *software development* practices, because these practices are just one of the dimensions in the model. We hypothesize that teams will be able to use the model to assess their progress across the five dimensions, and thus to guide and monitor their ongoing agile transitions.

ACKNOWLEDGEMENT

Our sincere gratitude to all the agile practitioners who participated in this research. This study was conducted as per the guidelines of the University of Auckland Human Participants Ethics Committee (UAHPEC), Application Reference #7867.

REFERENCES

[1] A. Rohunen, P. Rodriguez, P. Kuvaja, L. Krzanik, & J. Markkula, *"Approaches to agile adoption in large settings: a comparison of the results from a literature analysis and an industrial inventory"*, In International Conference on Product Focused Software Process Improvement, Springer Berlin Heidelberg, pp. 77-91. 2010.

[2] A. Sidky, J., Arthur, *"A Disciplined Approach to Adopting Agile Practices: The Agile Adoption Framework"*. Innovations in Systems and Software Engineering, vol 3, pp.203–216, 2007.

[3] A. Qumer, B. Henderson-Sellers, T. McBride, *"Agile Adoption and Improvement Model"*, In: Rodenes, M. (ed.) Proceedings of European and Mediterranean Conference on Information Systems (EMCIS), 2007.

[4] A. Qumer, B. Henderson-Sellers, *"A Framework to Support the Evaluation, Adoption and Improvement of Agile Methods in Practice."* Journal of Systems and Software, vol. 81, pp. 1899–1919, 2008.

[5] A. Martin, R. Biddle, J. Noble, *"The XP customer role: a grounded theory"*, In: Agile 2009, IEEE Computer Society, Chicago, 2009.

[6] A. Marchenko & P. Abrahamsson, *"Scrum in a multiproject environment: An ethnographically-inspired case study on the adoption challenges"*, In Agile, 2008 Conference, pp. 15–26, IEEE, 2008.

[7] B. Boehm, R. Turner, *"Rebalancing your Organization's Agility and Discipline"*, Extreme Programming and Agile Methods-XP/Agile Universe, Springer, Berlin, Germany, pp.1–8, 2003.

[8] B. Glaser, Basics of grounded theory analysis: emergence vs forcing. Sociology Press, Mill Valley, CA, 1992.

[9] B. Glaser, A.L. Strauss, The discovery of grounded theory. Aldine, Chicago, 1967.

[10] B. Glaser, Theoretical Sensitivity: Advances in the Methodology of Grounded Theory. Sociology Press, 1978.

[11] Glaser, B. The Grounded Theory Perspective III: Theoretical Coding. Sociology Press, Mill Valley, CA, 2005.

[12] B. Moravcová, & F. Legény, *""Agile Adoption" in IT Companies-Building a Change Capability by Qualitative Description of Agile Implementation in Different Companies"*, In International Conference on Exploring Services Science, pp. 251–262, Springer International Publishing, 2016.

[13] C. de O. Melo, V. Santos, E. Katayama, H. Corbucci, R. Prikladnicki, A. Goldman, and F. Kon, *"The evolution of agile software development in Brazil"*, Journal of the Brazilian Computer Society, vol. 19, no. 4, pp. 523–552, 2013.

[14] D. Goodman, & M. Elbaz, M, *"It's Not the Pants, it's the People in the Pants"*, Learnings from the Gap Agile Transformation: What Worked, How We Did it, and What Still Puzzles Us. AGILE'08 Conference, pp. 112–115, IEEE, 2008.

[15] E. V. Kelle, J. Visser, A. Plaat, and P. V. D. Wijst, *"An Empirical Study into Social Success Factors for Agile Software Development"*, 2015 IEEE/ACM 8th International Workshop on Cooperative and Human Aspects of Software Engineering, pp. 77–80, 2015.

[16] E. Whitworth and R. Biddle, *"The social nature of agile teams"*, In: Agile 2007, IEEE Computer Society, USA, 2007.

[17] H.C. Esfahani, *"Transitioning to Agile: A Framework for Pre-Adoption Analysis using Empirical Knowledge and Strategic Modeling*, PhD Dissertation, Graduate Department of Computer Science. University of Toronto, Canada 2012.

[18] G. Coleman, R. O'Connor, *"Using grounded theory to understand software process improvement: a study of Irish software product companies"*, Inf Softw Technol 49(6), pp. 654–667, 2007.

[19] G. M. Kapitsaki and M. Christou, *"Where Is Scrum in the Current Agile World?"* Proceedings of the 9th International Conference on Evaluation of Novel Approaches to Software Engineering, pp. 101–108, 2014

[20] J. McAvoy & T. Butler, *"A failure to learn in a software development team: the unsuccessful introduction of an agile method"*, In Information Systems Development, pp. 1–13, Springer US, 2009.

[21] J. Lopez-Martinez, R. Juárez-Ramírez, C. Huertas, S. Jim & C. Guerra-Garc, *"Problems in the Adoption of Agile-Scrum Methodologies: A Systematic Literature Review"*. In 2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT), pp. 141–148, IEEE, 2016.

[22] K. Sureshchandra, J. Shrinivasavadhani, *"Adopting Agile in Distributed Development"* In: Proceedings of the 2008 IEEE International Conference on Global Software Engineering, IEEE Computer Society, Washington, pp. 217–221, 2008.

[23] K. J. Stol, P. Ralph, B. Fitzgerald, *"Grounded theory in software engineering research: a critical review and guidelines"*, In Proceedings of the 38th International Conference on Software Engineering, pp. 120-131, ACM, 2016.

[24] L. Kompella, *"Agile methods, organizational culture and agility: some insights"*, Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering -CHASE 2014, pp. 40–47, 2014.

[25] M. D. Myers, *"Qualitative research in information systems"*, MIS Quaterly 21, 2, pp.241–242, 1997.

[26] M. Fowler and J. Highsmith, The agile manifesto. Softw. Dev. 9 (8), pp. 28–35, 1991.

[27] M. Hummel, C. Rosenkranz, and R. Holten, *"The Role of Communication in Agile Systems Development: An Analysis of the State of the Art"*, Business and Information Systems Engineering, vol. 5, no. 5, pp. 343–355, 2012.

[28] M. L. Markus and D. Robey, *"Information Technology and Organizational Change: Causal Structure in Theory and Research"*, Manage. Sci., vol. 34, no. 5, pp. 583–598, 1988.

[29] M. Pikkarainen, O. Salo, R. Kuusela, & P. Abrahamsson, *"Strengths and barriers behind the successful agile deployment—insights from the three software intensive companies in Finland"*, Empirical software engineering, 17(6), 675–702, 2012.

[30] M. Wufka and P. Ralph, *"Explaining Agility with a Process Theory of Change"*, In Agile Conference (AGILE), pp. 60–64, 2015.

[31] P. Gregory, L. Barroca, H. Sharp, A. Deshpande, & K. Taylor, *"The challenges that challenge: Engaging with agile practitioners' concerns"*, Information and Software Technology, 77, pp. 92–104, 2016.

[32] P. Ralph, *"Software engineering process theory: A multi-method comparison of Sensemaking-Coevolution-Implementation Theory and function-behavior-structure theory"*, Inf. Softw. Technol., vol. 70, pp. 232–250, 2016.

[33] R. Hoda, J. Noble, S. Marshall, *"Developing a grounded theory to explain the practices of self-organizing Agile teams"*, Empirical Software Engineering, vol. 17(6), pp. 609–639, 2012.

[34] R. Hoda, J. Noble, S. Marshall, *"Self-organizing roles on agile software development teams"* IEEE Transactions on Software Engineering, vol. 39(3), pp. 422–444, 2013.

[35] R. Hoda, L.K. Murugesan, *"Multi-level agile project management challenges: A self-organizing team perspective"*, Journal of Systems and Software, vol. 117, pp. 245–257, 2016.

[36] S. Augustine, Managing Agile Projects. Prentice Hall PTR , 2005.

[37] S. Georgieva, and G. Allan, *"Best practices in project management through a grounded theory lens"*, Electronic Journal of Business Research Methods 6, 1, pp. 43–52, 2008.

[38] S. Stavru, *"A critical examination of recent industrial surveys on agile method usage"*, The Journal of Systems and Software, 94, pp. 87–97, 2014.

[39] T. Dingsøyr, S. Nerur, V. Balijepally, N.B. Moe, *"A decade of agile method- ologies: Towards explaining agile software development"*, J. Syst. Softw. 85 (6), pp. 1213–1221, 2012.

[40] T. J. Gandomani & M. Z. Nafchi,*"An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach"*, Journal of Systems and Software, vol. 107, pp. 204–219, 2015.

[41] T. J. Gandomani, H. Zulzalil, A.A.A. Ghani, A.B.M. Sultan, & M. Z. Nafchi, *"Obstacles in moving to agile software development methods; at a glance"* Journal of Computer Science, 9(5), 620, 2013.

[42] VersionOne, 10th Annual state of agile software development, http://stateofagile.versionone.com/ (last accessed 10th August 2016

[43] V. Eloranta, K. Koskimies, T. Mikkonen, and J. Vuorinen, *"Scrum Anti-Patterns—An Empirical Study"* 2013 20th Asia- Pacific Software Engineering Conference (APSEC), vol. 1, pp. 503–510, 2013.

[44] W. E. Deming, *"Out of the Crisis"*, The MIT Press, Cambridge, Massachusetts, 2000.