Supporting Self-organizing Agile Teams
What's Senior Management Got to Do with It?

Rashina Hoda, James Noble, and Stuart Marshall

Victoria University of Wellington,
New Zealand
{rashina,kjx,stuart}@ecs.vuw.ac.nz
http://www.ecs.vuw.ac.nz

Abstract. Self-organizing Agile teams need a supportive environment to emerge and flourish. Through a Grounded Theory study of 58 Agile practitioners across 23 different software organizations in New Zealand and India, we found that senior management support is a critical environmental factor influencing self-organizing Agile teams. We describe the influence of senior management, and show how their support can create and sustain a supportive environment for self-organizing Agile teams.

Keywords: Agile Software Development, Self-Organizing Teams, Senior Management, Organizational Culture, Grounded Theory.

1 Introduction

The importance of senior management support in the adoption and use of Agile methods has been widely acknowlegded [5, 10, 14, 15, 22, 34]. Senior management has been found to influence organizational culture, which in turn influences the adoption of Agile methods in an organization [43, 46]. There is, however, little empirical evidence across multiple organizations, cultures, and countries to show how exactly senior managemment influences and supports self-organizing Agile teams in practice. Our research establishes senior management support as a critical environmental factor influencing self-organizing Agile teams and explains how senior management influences and supports such teams.

In this paper, we present the results of a Grounded Theory study involving 58 Agile practitioners from 23 different software development organizations in New Zealand and India conducted over a period of 4 years. Our study revealed that senior management support is a critical environmental factor influencing self-organizing Agile teams. We describe how senior management within organizations influences self-organizing Agile teams and how their support can enable self-organization in Agile teams. The rest of the paper is structured as follows: section 2 presents a brief background on self-organizing Agile teams and the role of senior management. Section 3 describes our research method. Section 4 presents the results of the study that describe the influence of senior management on self-organizing Agile teams. Section 5 discusses our findings in light of related work. Section 6 describes limitations of our study, followed by the conclusion in section 6.

2 Background

Self-organizing teams are at the heart of Agile software development [10, 12, 24, 32, 39, 41]. Self-organizing teams are considered the source of the best architecture, requirements, and design [24]. Self-organization is one of the principles behind the Agile Manifesto and has been identified as one of the critical success factors of Agile projects [4, 10, 24]. While Scrum specifically mentions self-organizing Agile teams, the concept of "*empowered*" teams has only recently been added to XP [48].

Agile teams are meant to be democratic teams—where all members are considered peers at the same level, without a strict hierarchy. Team members are empowered with collective decision making and cross-functional skills, which increases their ability to self-organize [34]. Self-organizing Agile teams are described as teams composed of "*individuals [that] manage their own workload, shift work among themselves based on need and best fit, and participate in team decision making*" [23]. Self-organizing teams must have a common focus, mutual trust, respect, and the ability to organize repeatedly to meet new challenges [12]. Sutherland, a co-creator of Scrum, explains that self-organizing teams consist of "*members with diverse backgrounds*" who are "*given a free hand*" by the top management [44].

Self-organizing Agile teams are not leaderless, uncontrolled teams [12, 45]. Leadership in self-organizing teams is meant to be light-touch and adaptive, providing feedback and subtle direction [3, 4, 9, 45]. Management in Agile teams is meant to be more facilitative and co-ordinating [34]. Leaders of Agile teams are responsible for setting direction, aligning people, obtaining resources, and motivating the teams [3].

Cockburn et al. point out that while the success of any process is largely dependent on the people, the ability of the people to achieve their goals is dependent on the level of support they receive from users, customers, and management [12]. They argue that Agile organizations practice "*leadership-collaboration*" instead of command and control style management, and that management in Agile organizations trust their teams to deliver to their best potential. They suggest that Agile teams function best in an organizational culture that supports people and collaborations.

In keeping with the Grounded Theory research method, related literature on the role of senior management is discussed in light of the results towards the end of the paper.

3 Research Method

Grounded Theory (GT) is the systematic generation of theory from data analyzed by a rigorous research method [17, 19]. GT was developed by sociologists Glaser and Strauss [20]. We chose GT as our research method for several reasons. Firstly, Agile methods focus on people and interactions and GT, used as a qualitative research method, allows us to study social interactions and behaviour. Secondly, GT is most suited to areas of research which have not been explored in great detail before, and the research literature on Agile team-customer relationships is scarce [21]. Finally, GT is being increasingly used to study Agile teams [11, 13, 31, 47]. Following Glaser's guidelines, we started out with a general area of interest — Agile project management — rather than beginning with a specific research problem [13].

3.1 Data Collection

We interviewed 58 Agile practitioners from 23 different software development organizations over 3 years from New Zealand and India. Figure 3 shows the participants and project details. In order to respect their confidentiality, we refer to the participants by numbers P1 to P58. All the teams were using Agile methods, primarily combinations of Scrum and eXtreme Programming (XP) — two of the most popular Agile methods today [5,36,40]. The teams practiced Agile practices such as iterative development, daily stand-ups, release and iteration planning, test driven-development (TDD), continuous integration and others. Participants' organizations offered products and services such as web-based applications, front and back-office applications, and local and off-shored software development services. Table 1 shows the participants and projects details.

The level of Agile experience varied across the different teams. While some teams had under a year of experience, others had been practicing Agile for over 5 years. The Indian teams were mostly catering to off-shored customers in Europe and USA, while

Table 1. Participants and Projects (P#: Participant Number, Position: Agile Coach (AC), Agile Trainer (AT), Developer (Dev), Customer Rep (Cust Rep), Business Analyst (BA), Senior Management (SM); *Organizational Size: XS < 50, S < 500, M < 5000, L < 50,000, XL > 100,000 employees)

P#	Positions	Method	Org. Size*	Location	Domain	Team Size	Project (months)	Iteration (weeks)
P1-P9	Dev x 3, BA, AC x 2, AT, Tester, Cust. Rep.	Scrum	M	NZ	Health	7	9	2
P10	AC	Scrum & XP	L	NZ	Social Services	4 to 10	3 to 12	2
P11-P18	Dev x 6, AC, SM	Scrum & XP	S	NZ	Environment	4 to 6	12	1
P19	SM	Scrum & XP	S	NZ	E-commerce	4	2	4
P20	AC	Scrum & XP	XL	NZ	Telecom & Transportation	6 to 15	12	4
P21	Cust. Rep.	Scrum	XS	NZ	Entertainment	6 to 8	9	4
P22	AC	Scrum & XP	S	NZ	Government Education	4 to 9	4	2
P23	AC	Scrum & XP	XS	NZ	Software Development	8	12	1
P24-P25	Dev x 2	Scrum	XS	NZ	Software Development	8 to 10	8	2
P26	AC	Scrum & XP	S	NZ	Farming	8	12	2
P27-P35	Dev x 4, AC, Tester, Sales Manager, SM x 2	Scrum & XP	S	India	Agile Software Development & Consultancy	5	6	2
P36-P39	AC x 4	Scrum & XP	M	India	Software Development	7 to 8	3 to 6	2
P40	SM	Scrum & XP	S	India	CRM and Finance	7 to 8	ongoing	3
P41	Designer	Scrum & XP	S	India	Web-based Services	5	1	2
P42	AC	Scrum & XP	L	India	Telecom	8 to 15	3	4
P43	AT	Scrum & XP	XS	India	Agile Training	7	8	2 to 4
P44-P45	Dev x 2	Scrum & XP	XS	India	Software Development	4	1	1
P46-P53	Dev, BA x 2, AT, AC, KS, HR, SM	Scrum & XP	M	India	Agile Software Products & Consultancy	15	12	1
P54	AC	Scrum & XP	M	India	Financial Services	8 to 11	36	2
P55	AC	RUP	XS	Canada	Telecom	10 to 15	10 to 15	2 to 4
P56	SM	Scrum	M	USA	Oil and Energy	5 to 8	12	2
P57	Cust. Rep.	Scrum & XP	M	USA	CRM and Cloud Computing	variable	variable	2 to 4
P58	AC	Scrum & XP	XS	USA	Health	variable	variable	2 to 4

most of the NZ teams were catering to in-house customers, some of whom were located in separate cities. We include more details of the context in sections below as necessary.

We collected data by conducting face-to-face, semi-structured interviews with Agile practitioners using open-ended questions. The interviews were approximately an hour long and focused on the participants' experiences of working with Agile methods, in particular the challenges faced in Agile projects and the strategies used to overcome them. We also observed several Agile practices such as daily stand-up meetings (co-located and distributed), release planning, iteration planning, and demonstrations. In order to get a rounded perspective, we interviewed practitioners in various roles: Developers, Agile Coach (Scrum Master or XP Coach), Agile Trainer, Customer, Business Analyst, Tester, and Senior Management. Senior Management was typically made up of chief executive officers, vice presidents, departmental heads, and senior managers. Data collection and analysis were iterative so that the constant comparison of data helped guide future interviews, and the analysis of interviews and observations fed back into the emerging results.

3.2 Data Analysis

We used open coding to analyze the interview transcripts in detail [16, 17]. We began by collating key points from each interview transcript. Then we assigned a *code*—a phrase that summaries the key point in 2 or 3 words—to each key point [16]. The codes arising out of each interview were constantly compared against the codes from the same interview, and those from other interviews and observations. This is GT's *constant comparison method* [18, 20] which was used again to group these codes to produce a higher level of abstraction, called *concepts* in GT.

The constant comparison method was repeated on the concepts to produce another level of abstraction called a *category*. As a result of this analysis, the concepts *Organizational Culture, Negotiating Contracts, Financial Sponsorship*, and *Resource Management* gave rise to the category *Senior Management Support*.

We analyzed the observations and compared them to the concepts derived from the interviews. We found our observations did not contradict but rather supported the data provided in interviews, thereby strengthening the interview data. The use of *memoing*—theorizing write-up of ideas about codes and their relationships—was vital in recording the relationships between codes [17]. The conceptual *sorting* of memos was done to derive an outline of the emergent theory, showing relationships between concepts.

3.3 Generating a Theory

The final step of GT is generating a theory, also known as *theoretical coding*. Theoretical coding involves conceptualizing how the categories (and their properties) relate to each other as a hypotheses to be integrated into a theory [17]. Following Glaser's recommendation, we employed theoretical coding at the later stages of analysis [18], rather than a coding paradigm from the beginning as advocated by Strauss [42].

Our research has led to a grounded theory of self-organizing Agile teams [28,30]. The theory of self-organizing Agile teams explains how software development teams take on one or more informal, implicit, transient, and spontaneous *roles* and perform balanced *practices* while facing critical environmental *factors*. These roles include *Mentor*, *Co-ordinator*, *Translator*, *Champion*, *Promoter*, and *Terminator* [28]. The practices involve balancing freedom and responsibility, cross-functionality and specialization, and continuous learning and iteration pressure [27]. The factors are senior management support and level of customer involvement [25,26,29]. Different aspects of the theory including roles, practices, and level of customer involvement as an environmental factor has been described elsewhere [25, 26, 27, 28, 29].

In the following sections, we describe senior management support—a critical environmental factor influencing self-organizing Agile teams. We have selected quotations drawn from our interviews that shed particular light on the concepts. Due to space reasons we cannot describe *all* the underlying key points, codes, and concepts from our interviews and observation that further ground the discussion.

4 Senior Management Influence and Support

A majority of the participants described how self-organizing Agile teams are greatly influenced by the senior management at their own organizations [28].

> "*..the organizations I see getting the most benefit from Scrum, from Agile, are organizations where senior management really gets it! Where senior management has been has been through training...Senior management took the time to read, learn about Agile. The least successful Agile adoptions are ones where senior management has no interest in Agile, they have no interest in what Agile is.*" — P43, Scrum Trainer, India

Senior management influences organizational culture, the types of contracts governing projects, financial sponsorhip, and resource management. A senior management that does not support self-organizing Agile teams causes several challenges for the team in each of these areas.

4.1 Organizational Culture

Organizational culture has been defined as "*a standard set of basic suppositions invented, discovered or developed by the group when learning to face problems of external adaptation and internal integration*" [38]. Organizational culture has a strong influence on the ability of an Agile team to be self-organizing.

Traditional software development teams typically adopt strictly hierarchical organization structures. Self-organizing Agile teams on the other hand, require organization structures that are informal in practice, where the boundaries of hierarchy do not prohibit free flow of information and feedback. In an informal organizational structure, the senior management is directly accessible by all employees (maintaining an 'open-doors' policy), and accepts feedback—both positive and negative.

Agile organizations, where all the teams operate using Agile software development, are characterized by informal organizational structures. Informality in organizational structure promotes openness. Openness was one of the most common traits mentioned by participants that made the organizational culture condusive for Agile teams. In such organizations, team members are free to voice opinions, raise concerns, seek management support in resolving their concerns, make collaborative decisions, and adapt to changes in their environment. This freedom provided by senior management is crucial for the team to achieve and sustain autonomy [28].

> "*don't expect that you're going to be in any other traditional hierarchical company...no matter if its 4 years or three years [of experience], they [team] can walk up to [CEO's name] and say 'this what you did, is bullshit' (laughs) and [CEO's name] will say 'oh, ok fine, let's discuss what happened'. So people have that freedom to voice their opinion very clearly. At the same time people will [give] feedback to you.*" — P52, Human Resource Manager, India

Starting with an informal structure has a cascading effect. Informality in the organizational structure leads to openness marked by free-flow of communication and feedback, which in turn leads to an organizational culture of trust. An organizational culture where teams trust their senior management to support them, and when senior management trusts the teams to perform and display responsibility, makes fertile grounds for self-organization to emerge.

> "*one of the big things that's made a difference there, is they already had an* **environment of trust**. *There was no fear in the organisation. You often see a level of fearfulness in very bureaucratic organisations, people are not prepared to give people—to give bad news, you know, the automatic punishment for being the bearer of bad news. I didn't see any of that at [company name], the level of confidence, the level of trust between management and the people on the ground was quite high already. So I think the ground was fertile for Agile...And that was because of the management attitude and the supportive nature of the managers.*" — P26, Agile Coach, New Zealand

In contrast, an organization with a strict hierarchical structure is not condusive to self-organizing Agile teams. A common example is that of a government sector organization, with a strict hierarchical structure. The software development teams in such organizations form one of the lowest levels of hierarchy, topped by middle management, and then senior management. Such hierarchical structure is often coupled with heavy-weight processes requiring substantial documentaion, long change management processes, and slow software delivery and deployment processes. Such a culture retricts both team's ability to practice light-weight Agile methods, and their ability to self-organize.

A strict hierarchical structure also has a cascading effect. The hierarchy in such an organization enforces a lack of openness marked by restricted and indirect lines of communication and feedback, which in turn leads to an environment of fear. Teams are afraid of voicing opinions, raising concerns, making collaborative decisions, and adapting to changes in their environment:

> "*...government business drivers are not 'time to market' or producing anything useful...the documentation is definitely more important than actual working software. They are not impressed at all by demos and working software—they almost didn't care! 'Why don't they have a big upfront design document?' It*

basically took me ages to basically force them to accept vertical slicing of that. I think its a fear of giving up control. Control doesn't exist, but they are afraid to give it up ... I was the PM on that project, they are still working on it, I went away screaming!" — P23, Agile Coach, NZ

On the other hand, some government sector organizations find that their culture, while seemingly different, can be receptive to changes brought on by Agile methods.

"*It's interesting because it's [Agile] probably a much better fit [to our culture] than you might think. On one hand [in] our organization...part of the culture is that people do tend to work in isolation...But because it's very scientifically oriented there's quite an openness to sharing ideas and information as well...once they [in-house customers] were exposed to the Agile development group and they were sitting in the room with them and the whiteboard and things, they became very open and very communicative. They would have never have volunteered that or expected that, but once they had people around them that were used to operating that way they were very open to that. So it fit quite well is what I'm saying, it fit pretty well.*" — P18, Senior Management, NZ

Senior management support, in terms of providing freedom and establishing an organizational culture of trust, is therefore extremely important for self-organizing Agile to establish and flourish. A senior management that supports self-organizing Agile teams will (a) maintain an informal structure, (b) provide freedom for teams to provide feedback, and (c) create an organizational culture of trust.

4.2 Negotiating Contracts

Self-organizing Agile teams are influenced by the type of contracts that govern their projects. Senior management—either directly in smaller organizations, or through their sales department in larger organizations—is responsible for negotiating contracts with customers. A customer can demand a fixed-bid contract where the cost, time, and scope of the project are fixed up-front. If senior management accepts the customer's demand for a fixed-bid contract, it has far-reaching consequences for the self-organizing Agile team. Teams find that "*fixed price doesn't work well with Agile*" because "*Agile talks about embracing change [and] can't do fixed price projects with changes coming in*" (P42, P27).

The process of fixing the cost, time, and scope of the project in a fixed-bid contract involves estimating the project. Senior management that does not support self-organizing Agile teams, fixes the cost, time, and scope based estimates provided by managers, rather than the teams. As a result the team is placed under pressure to deliver to often unrealistic estimates. The negative consequences of a fixed-bid contract in an Agile project are captured in the following comment by an Agile trainer and coach who worked several with Indian organizations:

"*The whole premise of the fixed-bid contract is that requirements will be fixed. The nature of software development is that requirements are inherently unstable and so when you are entering into contract negotiation, you are dealing with the recognition that the requirements will be unstable... Biggest source* *of dysfunction is not actually from the customer—the greater source of dysfunction comes from within the organization where the contract—fixed bid contract—is negotiated by the sales team, it is negotiated for the smallest amount of money possible. And so the team from day one is under pressure to over-commit and under-deliver and that I see again and again and again!*" — P43, Agile Trainer, India

In contrast, senior management that is aware of the negative consequences of fixed-bid contracts on the teams better supports self-organizing Agile teams for example, they provide customers with options such as offering an iteration on a trial basis, swapping features, the flexibility to buy more iterations or terminate the contract with an iteration's notice. For example, an Indian senior manager encouraged customers to buy a few iterations, instead of signing one contract for a large project:

"*Most of the time...[we] sell a certain number of iterations.*" — P34, Senior Management, India

By allowing the customers to use Agile development on a trial basis, Agile practitioners are able to build confidence among customers and provide them with risk coverage. Once the customers have tried a few iterations, then they are offered the option to buy more iterations or features as needed:

"*One thing we [development firm] used to do and worked very well—we used to tell the customers you don't have any risks...in case of Agile we enter into a contract with the client—OK we'll show you working software every fifteen days, you'll have the option of ending the project within one sprint's notice. Maximum they can lose is one sprint. Advantage we show to client they don't have to make up their entire mind... [they] can include changes in sprints -they see it as a huge benefit to them.*" — P27, Developer, India

Some Agile practitioners allow the customers to swap features. The project is delivered at the same time and price as initially specified in the contract, but the customer can remove product features that they no longer require and replace them with new ones (requiring approximately equivalent effort) that are of more business value to them:

"*... customer after seeing demo after fourth iteration realizes the features built, say the thirteenth feature, is not required and he needs something else... he can swap the two.*" — P27, Developer, India

By providing the customers with the option to quit the project in the worst case scenario, their financial risks are covered. If the customers are unhappy with the results, they could always quit the project.

If a customer is still insistent on a fixed-bid contract, senior management can support a self-organizing Agile team by inviting the team to estimate their projects. Based on the rate of development per iteration—the team velocity—the team can estimate the time required for developing a particular set of requirements in a given domain. Then some amount of extra time could be added to the estimated time as a buffer. The contract is then drawn on this estimated time (including buffer) for a fixed price and scope.

> "*Agile will not ask you in how much time will you [need to] complete the project...but [the customer will]. Sometimes you've got to map internal Agile practices to customer practices....Actually it comes from a lot of experience on Agile. When you know that okay this is generally the velocity of the team that the team is able to do within the given domain, the given complexity and then you make some rough estimates, including some buffer. [Customer says] 'okay I want these features, tell me the time'. so then we'll make prediction based on Agile data that this is the team size, this is the velocity, we assume the team won't change then the Agile burndown chart will say let's say 2 weeks so we'll say okay another 2 days of buffer, so 2 weeks ands 2 days, something like that.*" — P28, Developer, India

A small amount of buffer time was important to allow the customer the possibility of introducing changes in requirements along the way, while giving the development team time to respond to those changes. Buffering was a practical strategy of working with a fixed-bid contract while using Agile methods.

Finally, senior management in Agile organizations are very careful about negotiating contracts that are "*Agile-friendly*". They frequently have a specialized sales team that understand Agile methods and the consequences of the contract on the self-organizing Agile teams.

> "*In the sales room, even the way we work is Agile. We have two groups, one for marketing, one for sales. We have stages for each teams—we use kind of post-its and put them up. So even our sales is Agile.*" — P33, Sales Manager, India

Senior management that supports self-organizing Agile teams will (a) try to convince customers to try flexible contract options, (b) engage the team in providing estimates for the fixed-bid contract, along with adding a contingency buffer, or (c) negotiate "*Agile friendly*" contracts.

4.3 Financial Sponsorship

Self-organizing Agile teams need financial sponsorship from their senior management in the form Agile training and an infrastructure that's supportive of self-organizing practices. The team needs senior management support in order to benefit from the presence of a *Mentor* in the form of an Agile Coach [28]. The Agile Coach is often a contracting consultant, hired specifically to train a new team on Agile principles, values, and practices. In other cases, an existing project manager in the organization may take up the *Mentor* role. The senior management provides financial support by either hiring contracting Agile Coaches or sponsoring these managers, and occasionally other team members, to receive Agile training (e.g. a Scrum Master Certification).

Financial support is also required in the form of infrastructure support, such as setting up an open-plan workplace and tools for electronic communication and collaboration with distant customers. A supportive senior management champions the cause of self-organizing Agile teams and provides financial support for such an infrastructure.

"*In most organizations I'd say skype would be blocked. They [senior management in non-Agile organizations] say we do chat or call their friends abroad and waste time but here in [this organization], skype is there on every machine because the management knows that it is an important communication tool...So yeah definitely the change in the mindset of the organization has to be there. For example, they [senior management] have provided LCD TVs within the rooms and there are a lot of skype meeting rooms which have LCD TVs, camera, and you have skype installed. If I stand up, you actually go through those moves and you can see the customer and they can see us, so like that. Again there is that initiative from the senior management because they might as well say that 'okay do it on your own machine or we cannot provide LCD TVs for every team!' So that drive has to come from them definitely.*" — P29, Developer, India

"*...level of sponsorship means...the senior manager...say 'This is the methodology we are adopting. I expect you to change your practices and techniques to support that, and here's some money to do so...here's some time, here's some resources.*" — P7, Agile Coach, NZ

Senior management that supports self-organizing Agile teams is willing to make such financial investments as (a) hiring a *Mentor* for new teams or providing existing Project Managers with Agile training and (b) providing the infrastructure necessary for effective functioning of the self-organizing Agile teams.

4.4 Human Resource Management

An important role of senior management is the way they manage human resources. For self-organizing Agile teams, dedicated teams are highly desired. When team members are allocated to multiple projects, it has a negative influence on the teams' ability to perform and self-organize. One of the main characteristics of self-organizing Agile teams is high levels of cohesion and collaboration within the team. The team's ability to self-organize is dependent on understanding each others' strengths and weaknesses and forming a team culture of openness and respect. It takes time for a team to learn about each other and self-organize based on members' individual abilities.

"*What I think affected our project...[the developer] was working on another project, he didn't have enough time, so he didn't have the space to chat with anybody, to discuss ideas with anybody, to work with anybody, so he was really just on his own, and I think that really impacted a lot of the work he did in the last few months ... When you're working in a team like this [Agile team] and you've got to work quite closely, the individuals in the team matter.*" — P21, Customer Rep, NZ

If the members are split across multiple projects, it affects their ability to perform group programming that enables self-organization. Senior management that does not realize the implications of their resource management can have a negative influence on the team:

"*[explaining how resource management works]...resource-assignment, right...If I am VP (vice president)...for me, resource is a pure mathematical figure. 0.25 is*

> *2 hours. if I divide, make the equation work, I'll be happy! Ground reality is different. People can't work 0.25! One side as a VP I want to get business, I have to do equations: 0.5 from here, 0.5 from here etc and make it 3...pure mathematics...not feasible in ground reality...People have to be mature enough...[its] just a matter of understanding the ground reality: if they [senior management] are a developer how would they react to the situation?*" — P39, Agile Coach, India

On the other hand, supportive senior management values their teams and respects their human side as much, if not more, than their technical skills:

> "*...I personally feel it's one of those companies where does a lot for the people. They [senior management] definitely understand people, values, and you know, they understand their emotions...so we do respect people and you know if they [team] have any concerns or worries we [company] will try to understand it.*"
> — P52, Human Resource Manager, India

Resource management in terms of the hiring process and removal of individuals from teams is also influenced by senior management. In Agile organizations where senior management supports self-organizing Agile teams, their Human Resources departments are set up specifically to hire people that are likely to fit into Agile teams.

Sometimes, team members need to be removed from an Agile team because of their inability to fit into the culture. One of the team members typically takes on a *Terminator* role and seeks senior management support in removing such individuals [28].

Senior management supports self-organizing Agile teams through managing resources by (a) providing dedicated resources to projects, (b) hiring individuals to fit into an Agile culture, and (c) removing individuals who threaten self-organizing teams with the help of a *Terminator*.

5 Discussion and Related Work

Senior management influences the organizational structure and culture in an organization [34]. The importance of senior management support in the form of a condusive organizational culture has been widely acknowledged [5,22,14,15,34,10,43,46]. Agile methods challenge conventional management ideas, and require changes in organization structure, culture, and management practices in traditional software development organizations [15,34]. Changing mindsets and cultures, however, is no trivial task [8].

Beck highlights the influence of organizational culture on the use of Agile methods and argues that an environment of isolation, timidity, and secrecy will cause challenges [6]. Our research supports the claim that an environment of openness, communication, and trust is imperative for self-organizing Agile teams. The influence of senior management in creating and maintaining such an environment is extremely important.

A study of the influence of organizational culture on Agile method use found correlations between certain aspects of organizational culture and the use of Agile practices [43]. In particular, the study found that organizations that value collaboration, feedback, learning, and empowerment of people are better suited to support Agile methods. Our findings support these claims, as well as the conclusion that hierarchically structured organizations are not well suited to Agile methods. Management in Agile teams is meant to be facilitative and collaborative [34,43]. Empowerment and collective decision making in Agile teams are seen to increase their ability to self-organize [34]. Similarly, our research shows that these aspects of organizational culture have a strong influence on the self-organizing ability of Agile teams.

Tolfo and Wazlawick studied the influence of organizational culture on the adoption of XP [46]. Their study concludes that while XP generally assumes the existence of a condusive environment for XP teams, such an organizational culture is not always present in software organizations. In particular, the level of autonomy an organization provides to its members was found to be an important ingredient of a condusive organizational culture. Our findings supports this claim and link senior management support to self-organizing teams.

Most studies that have explored the influence of senior management support and organizational culture have focussed on XP teams [37,46]. Studies exploring the influence of organizational culture on Scrum teams, however, are limited. In a Scrum-based study, Moe et al. found that the management did not provide an environment conducive to self-organization that led to reduced external autonomy [33]. Our research found that self-organizing Agile teams (practicing Scrum or combinations of Scrum and XP) require a condusive organizational culture marked by freedom, openness, trust, and an informal organizational structure. In contrast, an organization with a hierarchical organizational structure and an environment of restricted, formal, and indirect communication restricts the teams' ability to self-organize.

In a paper on introducing lean principles with Agile practices in a Fortune 500 company, Parnell-Klabo described various difficulties in securing a buy-in for a pilot project [35]. Some of these included obtaining facility space for collocation, gaining executive support, and influencing the change curve.

Several attributes of Agile methods are well aligned with senior management's business drivers discussed in this chapter. For example, fast delivery and rapid response to changes in business and technology are key attributes of Agile methods [1,5,7,24]. It would appear then that convincing senior management to support self-organizing Agile teams would be an easy task. However, this is not always the case. Organizations don't change for the sake of change, they change when they see benefit from it.

A single case-study of adopting XP at a diverse, multidisciplinary web-development environment at IBM highlights the existence of skepticism amongst senior management regarding Agile nomenclature [22]. For example, the use of the XP term "*planning game*" was not well received by senior executives who preferred more formal-sounding terms like "*planning process*". Our participants shared experiences of facing skepticism when trying to secure senior management support. Our findings suggest that convincing senior management not only requires that a team member takes on the role of a *Champion*, but also that they understand senior management's business drivers [28]. These business drivers include applicability of Agile methods to project context, time-to-market, customer demands, and process improvement. In other words, senior management does not undertake drastic changes in their organizations without a strong incentive. Understanding the business drivers particular to different organizations and their senior management is critical for a *Champion* advocating the introduction and continued support for self-organizing Agile teams.

Most of the above studies have explored the influence of management support on the adoption and use of Agile methods. Our findings show the influence of senior management support on self-organizing Agile teams in particular, and highlight strategies used by teams to secure such support, in an effort to achieve and sustain self-organization.

6 Limitations

Since the codes, concepts, and category emerged directly from the data, which in turn was collected directly from real world, the results are grounded in the context of the data [2]. We do not claim the results to be universally applicable: rather, they accurately characterize the context studied [2]. Our choice of research destinations and participants were limited in some ways by our access to them.

7 Conclusion

We conducted a Grounded Theory study, involving 58 Agile practitioners from 23 different software development organizations in New Zealand and India, over a period of 4 years. The results establish senior management support as a critical environmental factor influencing self-organizing Agile teams.

In this paper, we have described the importance of senior management in supporting self-organizing Agile teams. Senior management influences self-organizing teams through organizational culture, negotiating contracts, financial sponsorship, and resource management. Senior management supports self-organizing Agile teams by creating and maintaining an open and informal organizational culture, negotiating "*Agile-friendly*" contracts, providing financial sponsorship, and managing human resources in a way that supports self-organization. In contrast, senior management that does not manage these factors effectively causes challenges for a self-organizing teams at best and disables self-organization in Agile teams at worst. Future studies could explore other ways in which senior management may influence and support self-organizing Agile teams in other cultures.

Acknowledgments. Our thanks to all the participants. This research is supported by a BuildIT PhD scholarship (NZ), an Agile Alliance academic grant (USA).

References

1. Abrahamsson, P., Warsta, J., Siponen, M.T., Ronkainen, J.: New directions on agile methods: a comparative analysis. In: Proceedings of 25th International Conference on Software Engineering, pp. 244–254 (2003)
2. Adolph, S., Hall, W., Kruchten, P.: A methodological leg to stand on: lessons learned using grounded theory to study software development. In: CASCON 2008, pp. 166–178. ACM, New York (2008)
3. Anderson, L., Alleman, G.B., Beck, K., Blotner, J., Cunningham, W., Poppendieck, M., Wirfs-Brock, R.: Agile management - an oxymoron?: who needs managers anyway? In: OOPSLA 2003, pp. 275–277. ACM, New York (2003)

4. Augustine, S.: Managing Agile Projects. Prentice Hall PTR, Upper Saddle River (2005)
5. Beck, K.: Extreme Programming Explained: Embrace Change, 1st edn. Addison-Wesley Professional, Reading (1999)
6. Beck, K., Andres, C.: Extreme Programming Explained: Embrace Change, 2nd edn. Addison-Wesley Professional, Reading (2004)
7. Boehm, B.: Get ready for agile methods, with care. Computer 35(1), 64–69 (2002)
8. Boehm, B.W., Turner, R.: Rebalancing your organization's agility and discipline. In: XP/Agile Universe, pp. 1–8 (2003)
9. Chau, T., Maurer, F.: Knowledge sharing in agile software teams. In: Lenski, W. (ed.) Logic Versus Approximation. LNCS, vol. 3075, pp. 173–183. Springer, Heidelberg (2004)
10. Chow, T., Cao, D.: A survey study of critical success factors in agile software projects. J. Syst. Softw. 81(6), 961–971 (2008)
11. Cockburn, A.: People and Methodologies in Software Development. PhD thesis, University of Oslo, Norway (2003)
12. Cockburn, A., Highsmith, J.: Agile software development: The people factor. Computer 34(11), 131–133 (2001)
13. Coleman, G., O'Connor, R.: Using grounded theory to understand software process improvement: A study of Irish software product companies. Inf. Softw. Technol. 49(6), 654–667 (2007)
14. Coram, M., Bohner, S.: The impact of agile methods on software project management, pp. 363–370 (2005)
15. Dybå, T., Dingsoyr, T.: Empirical studies of Agile software development: A systematic review. Inf. Softw. Technol. 50(9-10), 833–859 (2008)
16. Georgieva, S., Allan, G.: Best practices in project management through a grounded theory lens. Electronic Journal of Business Research Methods (2008)
17. Glaser, B.: Theoretical Sensitivity: Advances in the Methodology of Grounded Theory. Sociology Press, Mill Valley (1978)
18. Glaser, B.: Basics of Grounded Theory Analysis: Emergence vs Forcing. Sociology Press, Mill Valley (1992)
19. Glaser, B.: The Grounded Theory Perspective III: Theoretical Coding. Sociology Press, Mill Valley (2005)
20. Glaser, B., Strauss, A.L.: The Discovery of Grounded Theory. Aldine, Chicago (1967)
21. Grisham, P.S., Perry, D.E.: Customer relationships and extreme programming. In: HSSE 2005: Proceedings of the 2005 Workshop on Human and Social Factors of Software Engineering, pp. 1–6. ACM, New York (2005)
22. Grossman, F., Bergin, J., Leip, D., Merritt, S., Gotel, O.: One XP experience: introducing agile (XP) software development into a culture that is willing but not ready. In: CASCON 2004: Proceedings of the 2004 Conference of the Centre for Advanced Studies on Collaborative Research, pp. 242–254. IBM Press (2004)
23. Highsmith, J.: Agile Project Management: Creating Innovative Products. Addison-Weasley, USA (2004)
24. Highsmith, J., Fowler, M.: The Agile Manifesto. Software Development Magazine 9(8), 29–30 (2001)
25. Hoda, R., Noble, J., Marshall, S.: Negotiating Contracts for Agile Projects: A Practical Perspective. In: Abrahamsson, P., Marchesi, M., Maurer, F. (eds.) XP 2009. LNBIP, vol. 31, pp. 186–191. Springer, Heidelberg (2009)
26. Hoda, R., Noble, J., Marshall, S.: Agile Undercover: When Customers Don't Collaborate. In: Sillitti, A., Martin, A., Wang, X., Whitworth, E. (eds.) XP 2010. LNBIP, vol. 48, pp. 73–87. Springer, Heidelberg (2010)

27. Hoda, R., Noble, J., Marshall, S.: Balancing Acts: Walking the Agile Tightrope. In: Co-operative and Human Aspects of Software Engineering workshop at ICSE 2010, South Africa. ACM, New York (2010)
28. Hoda, R., Noble, J., Marshall, S.: Organizing Self-Organizing Teams. In: ICSE 2010: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, South Africa, pp. 285–294. ACM, New York (2010)
29. Hoda, R., Noble, J., Marshall, S.: Impact of Inadequate Customer Involvement on Self-Organizing Agile Teams. Journal of Information and Software Technology (2010) (in press)
30. Hoda, R.: Self-Organizing Agile Teams: A Grounded Theory. PhD thesis, Victoria University of Wellington, Wellington, New Zealand (2010) (submitted)
31. Martin, A., Biddle, R., Noble, J.: The XP customer role: A grounded theory. In: AGILE 2009, Chicago. IEEE Computer Society, Los Alamitos (2009)
32. Martin, R.: Agile Software Development: principles, patterns, and practices. Pearson Education, NJ (2002)
33. Moe, N.B., Dingsoyr, T., Dybå, T.: Understanding self-organizing teams in agile software development. In: ASWEC 2008, Washington, pp. 76–85. IEEE, Los Alamitos (2008)
34. Nerur, S., et al.: Challenges of migrating to Agile methodologies. Commun. ACM 48(5), 72–78 (2005)
35. Parnell-Klabo, E.: Introducing Lean principles with Agile practices at a Fortune 500 company. In: AGILE 2006: Proceedings of the Conference on AGILE 2006, Washington, DC, USA, pp. 232–242. IEEE Computer Society, Los Alamitos (2006)
36. Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., Still, J.: The impact of agile practices on communication in software development. Empirical Softw. Engg. 13(3), 303–337 (2008)
37. Robinson, H., Sharp, H.: Organisational culture and XP: three case studies. In: Agile Development Conference/Australasian Database Conference, pp. 49–58 (2005)
38. Schein, E.H.: Organizational Culture and Leadership, 1st edn. Jossey-Bass Publishers, San Francisco (1985)
39. Schwaber, K.: Scrum guide. Scrum Alliance Resources, `http://www.scrum.org/storage/scrumguides/Scrum%20Guide.pdf` (last accessed on November 9, 2010)
40. Schwaber, K., Beedle, M.: Agile Software Development with SCRUM. Prentice-Hall, Englewood Cliffs (2002)
41. Sharp, H., Robinson, H.: An ethnographic study of XP practice. Empirical Softw. Engg. 9(4), 353–375 (2004)
42. Strauss, A., Corbin, J.: Basics of qualitative research: grounded theory procedures and techniques. Sage Publications, Newbury Park (1990)
43. Strode, D.E., Huff, S.L., Tretiakov, A.: The Impact of Organizational Culture on Agile Method Use. In: Proceedings of the 42nd Hawaii International Conference on System Sciences, Washington, DC, USA, pp. 1–9. IEEE Computer Society, Los Alamitos (2009)
44. Sutherland, J.: Roots of Scrum: Takeuchi and Self-Organizing Teams. World Wide Web electronic publication, `http://jeffsutherland.com` (last accessed on March 31, 2010)
45. Takeuchi, H., Nonaka, I.: The new new product development game. Hardvard Business Review 64(1), 137–146 (1986)
46. Tolfo, C., Wazlawick, R.S.: The influence of organizational culture on the adoption of extreme programming. Journal of Systems and Software 81(11), 1955–1967 (2008)
47. Whitworth, E., Biddle, R.: The social nature of Agile teams. In: Agile 2007, USA, pp. 26–36. IEEE Computer Society, Los Alamitos (2007)
48. XP: Extreme programming: A gentle introduction. World Wide Web electronic publication, `http://www.extremeprogramming.org/` (last accessed on October 23, 2010)