Final version available at Elsevier : http://dx.doi.org/10.1016/j.jss.2016.01.004

The Daily Stand-Up Meeting: A Grounded Theory Study

VIKTORIA STRAY[1], DAG I. K. SJØBERG[1, 2], TORE DYBÅ[1, 2]

[1]*University of Oslo, Norway*

[2]*SINTEF, Norway*

stray@ifi.uio.no, dagsj@ifi.uio.no, tore.dyba@sintef.no

Abstract

Context: The daily stand-up meeting is one of the most used agile practices but has rarely been the subject of empirical research. **Aim**: The present study aims to identify how daily stand-up meetings are conducted and what the attitudes towards them are. **Method**: A grounded theory study of the daily stand-up meeting was conducted with twelve software teams in three companies in Malaysia, Norway, Poland and the United Kingdom. We interviewed 60 people, observed 79 daily stand-up meetings and collected supplementary data. **Results**: The factors that contributed the most to a positive attitude towards the daily stand-up meeting were information sharing with the team and the opportunity to discuss and solve problems. The factors that contributed the most to a negative attitude were status reporting to the manager and spending too much time; both the frequency and duration of the meeting were perceived to be too much. Based on our results, we developed a grounded theory of daily stand-up meetings and proposed evidence-based guidelines on how to organize them. **Conclusion**: Organizations should be aware of the factors that may affect attitude towards daily stand-up meetings and should consider our proposed guidelines to improve the way the meetings are conducted.

Keywords: Daily meeting, Stand-up meeting, Daily Scrum, Grounded theory, Agile software development, Qualitative research.

1 Introduction

Common to all agile methods is an emphasis on communication and the human side of software development (Merisalo-Rantanen et al., 2005). Conducting a daily stand-up meeting (DSM) is an important practice in the agile methods Scrum and Extreme Programming to improve communication in software projects. The software development industry has extensively adopted agile practices, many of which have been thoroughly investigated (Dingsøyr et al., 2012). However, the daily stand-up meeting (DSM), the most used agile practice (VersionOne, 2014), has rarely been the primary subject of research. According to a 2013 survey (VersionOne, 2014), DSMs are used by 85% of the organizations that employ agile development and it is a team practice that often distinguishes agile from non-agile teams (Murphy et al., 2013).

DSMs, like other types of meetings, are fitted into the rhythm of the organization and have their own place in clock time, duration and location. Even though the DSMs are expected to last for only 15 minutes, their total costs are significant. As an illustration, suppose half of the approximately one million software developers in the United States (Bureau of Labor Statistics, 2014) spent one hour a week in DSMs, the direct cost in time could reach approximately USD 1 billion annually in salaries. Consequently, to justify the cost of conducting DSMs, their benefits must be substantial.

In this article, we propose a theory of DSMs that includes propositions among DSM constructs, with explanations grounded in data. The data was generated from 79 observations of DSMs of eight software teams in three companies and 60 interviews with team members, Scrum Masters, Product Owners, and managers that worked in these teams and an additional set of four teams. Since meeting satisfaction is part of overall job satisfaction (Rogelberg et al., 2010), we also considered what factors positively and negatively affect attitudes towards DSMs.

Several studies have investigated the DSM as one of several agile practices. Pikkarainen et al. (2008) studied the impact of agile practices on communication and found that DSMs kept developers, project leaders and customers aware of the project status and helped the developers resolve design issues faster. Paasivaara et al. (2008) examined agile practices in global software development and found that DSMs help reveal problems early and improve transparency between sites. Moe et al. (2010) studied the nature of self-managing agile teams and found that DSMs were mostly used by a Scrum Master to obtain an overview of activity and progress in a project. McHugh et al. (2012) examined how agile practices impact trust and found that DSMs help a team function more cohesively. Dorairaj et al. (2012) studied dynamics in distributed teams and found that the practice promotes team interaction and builds a "one team" mindset.

The DSM was the primary study topic in some of our earlier research. In a longitudinal study, DSMs led to a greater commitment to a failing course of action (Stray et al., 2012b). In another study, we investigated the proportion of time spent on answering the three questions suggested in the Scrum literature (Sutherland and

Schwaber, 2011): “What has been accomplished since the last meeting? What will be done before the next meeting? What obstacles are in the way?” We found that these questions took less than a quarter of the meeting; more effort was spent on understanding problems and discussing solutions (Stray et al., 2012a). In yet another study, we identified thirteen obstacles to efficient DSMs and suggested ways to overcome them (Stray et al., 2013).

Much can be learned from case studies by doing a secondary grounded theory analysis (Glaser, 2001, p. 97). This study builds on our previous research. Among the 60 interviews of this study, 7 were reused from the study reported in (Stray et al., 2011), 17 were reused from the study reported in (Stray et al., 2012b) and 9 were reused from the study reported in (Stray et al., 2013). We reanalyzed the case study material and iteratively compared it with newly collected material.

This study also contributes to increasing the understanding of the costs and benefits of DSMs, which is important for improving agile software development. Finally, our work answers a call for more empirically based theories in software engineering (Herbsleb and Mockus, 2003; Hannay et al., 2007; Sjøberg et al., 2007).

The remainder of this paper is organized as follows. Section 2 outlines relevant background literature. Section 3 describes the research methods used. Section 4 reports our results. Section 5 discusses the results, limitations of the study and future work. Section 6 concludes.

2 Background

This section gives a brief introduction to the field of meetings in general and the DSM in agile development in particular.

2.1 Meetings in General

According to Boden (1994, p. 84), a meeting is “a planned gathering, whether internal or external to an organization, in which the participants have some perceived (if not guaranteed) role, have some forewarning (either longstanding or quite improvisatorial) of the event, which has itself some purpose or ‘reason,’ a time, [a] place, and, in some general sense, an organizational function.”

Employees spend a lot of time in meetings, and the amount has increased in the last decades (Rogelberg et al., 2006). A great portion of meeting time is perceived as ineffective, and over one third of the time is wasted, with annual losses up to USD 37 billion in the United States alone (Elsayed-Elkhouly et al., 1997). Furthermore, meeting demands also affect employee productivity beyond the meeting setting (Allen et al., 2012). For example, a meeting is a particular kind of interruption (Rogelberg et al., 2006), which may affect employees’ subsequent readiness to perform by influencing their psychological state. After an interruption, people have to scan and evaluate all new information that they have encountered; several short interruptions have a greater effect than one long interruption (Zijlstra et al., 1999). Parnin and Rugaber (2011) analyzed 10 000 programming sessions and found that developers returning to a task after an interruption, such as a meeting, needed 15 minutes or more to collect their thoughts and make the first edit for most tasks (57%).

Very few empirical research studies have specifically focused on team meetings; most studies use meetings only as a context for studying other variables of interest (O'Neill and Allen, 2012), although there are exceptions. Anderson et al. (2007) explored the nature of communication in virtual team meetings. They found that the communication was influenced by the way in which the technologies were used. For example, the person controlling the keyboard dominated cross-site communication even though the audio facility made contributions from any team member perfectly audible at either site. Sonnentag and Volmer (2009) studied how individuals in software design teams contributed to teamwork processes during team meetings. They found expertise to be a strong predictor of individuals' contributions. Team members with a high level of expertise were more involved in problem analysis and goal specification than those with less expertise. Kauffeld and Lehmann-Willenbrock (2012) analyzed videotaped team meetings and linked their observations with objective data on team productivity and organizational success. Their findings show that team meeting interaction processes affect meeting satisfaction, team productivity and organizational outcomes.

2.2 DSM in Agile Software Development

In software engineering, conducting DSMs in development teams became popular with the introduction of the agile methods XP and Scrum, in which the meetings are a mandatory practice. One of the main characteristics of agile teams is conducting DSMs. Murphy et al. (2013) found that agile teams were 48% more likely to use DSMs than non-agile teams. The DSM is supposed to be a brief gathering of team members and satisfies the definition of a meeting given in Section 2.1, because the event is planned and has a pre-arranged time and place, and a purpose. These characteristics distinguish the DSM from incidental social encounters at work. When the DSM was first introduced in Scrum, all of the team members were supposed to answer the following three questions in the meeting (Sutherland, 2004):

(1) What did you do yesterday?
(2) What will you do today?
(3) What obstacles got in your way?

A survey from 2009 reported that 69% of agile practitioners adhered to these three questions (VersionOne, 2009). In different methods and communities, DSMs have different names. We use the term DSM, which originates from XP (Wells, 2013). Other names are *Scrum meeting* (Rising and Janoff, 2000), *frequent, short meetings* (Rising, 2002), *morning roll call* (Anderson, 2003), *daily huddle meeting* (Paez et al., 2005), *daily meeting* (Pikkarainen, 2008) and *daily Scrum meeting* (Sutherland and Schwaber, 2013a).

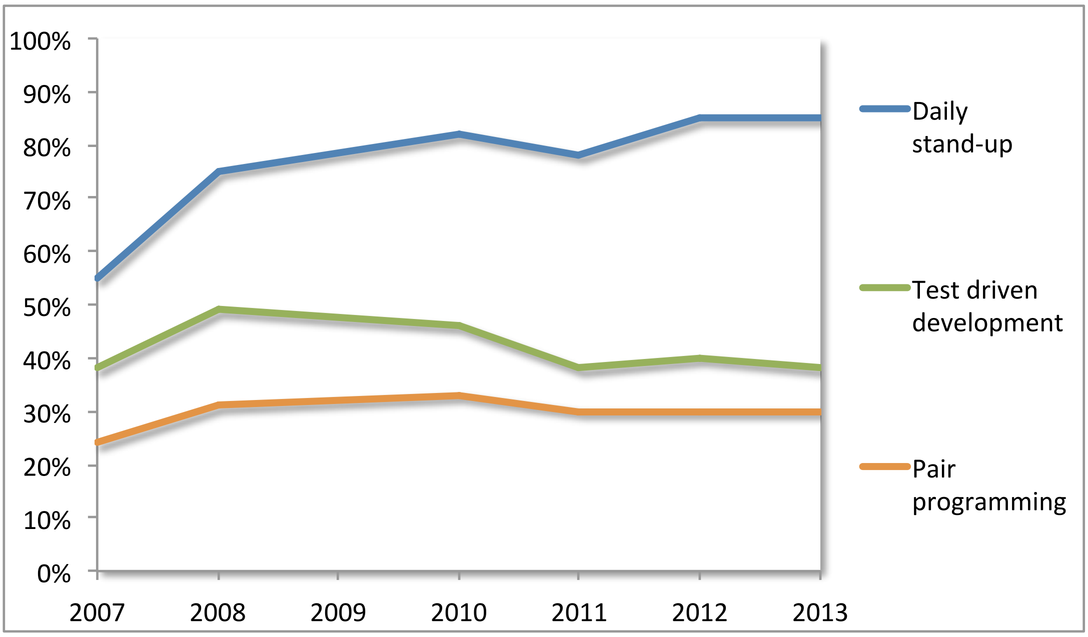


Figure 1: The percentage of agile practitioners employing the techniques DSM, test driven development and pair programming (VersionOne, 2007–2014)

Despite being the most commonly used agile practice (VersionOne, 2013), the DSM has been overlooked as a topic of study. The two most thoroughly investigated agile practices are pair programming and test-driven development (Dingsøyr et al., 2012). Figure 1 compares the adoption rate over time of the DSM, test-driven development and pair programming. The use of the DSM by agile practitioners increased from 55% in 2007 to 85% in 2013. Test-driven development had no increase (38% in both 2007 and 2013). Pair programming increased from 24% in 2007 to 30% in 2013.

The DSM is defined as a mandatory practice in Scrum. The originators of Scrum conceived the idea of a daily meeting from a paper (Coplien, 1994) that reported on the software project that developed Borland's Quattro Pro, in which architecture, design and interface issues were discussed in daily meetings (Schwaber and Beedle, 2002, p. 12). Being an apparently simple practice to implement, the DSM has garnered increasing interest in terms of adoption and diffusion. Recommendations such as "The best way to begin implementing Scrum is to establish daily Scrum status meetings" (Schwaber, 2003), is one possible explanation for the popularity of the DSM. The DSM was a way for software organizations to show that the organization had joined the agile movement and to be able to use new jargon such as "Daily Scrum."

The DSM is not a mandatory practice in Kanban, but many teams that practice Kanban nevertheless use DSMs; for example, the Kanban teams reported in Sjøberg et al. (2012) and the Kanban teams studied in this thesis. According to Kniberg and Skarin (2010), Kanban teams tend to use a more board-oriented format in which they focus on bottlenecks on the Kanban board instead of a format in which every person reports one by one.

Table 1 gives an overview of the current guidelines of DSM that most people use today. Based on Boden's (1994, p. 86) division of formal and informal meetings, DSMs may be characterized as informal because they are task and decision oriented, have casual conversation styles and are generally unrecorded, and members are gathered for a narrow organizational goal.

Table 1: Current guidelines of DSM

Characteristic	Daily Stand Up Meeting in XP (Wells, 2013)	Daily Scrum (Sutherland and Schwaber, 2013a)
Purpose	Communication among the entire team	Synchronize activities and create a plan for the next 24 hours
Potential benefits	Communicate problems and solutions and promote team focus. Replace many other meetings, giving a net savings several times its own length	Optimize the probability that the development team will meet the sprint goal. Improve communications, eliminate other meetings, identify impediments to development for removal, highlight and promote quick decision-making, improve the development team's level of knowledge
Potential pitfalls	Not mentioned	Consider the meeting as a status event (Sutherland and Schwaber, 2013b)
DSM questions	Developers report at least three things: 1. What was accomplished yesterday? 2. What will be attempted today? 3. What problems are causing delays?	The development team members explain the following: 1. What did I do yesterday that helped the Development Team meet the Sprint Goal? 2. What will I do today to help the Development Team meet the Sprint Goal? 3. Do I see any impediment that prevents me or the Development Team from meeting the Sprint Goal?
Format	Meeting participants stand up in a circle	Not mentioned
Turn-taking	Not mentioned	Not mentioned
Frequency	Every day	Every day
Time of day	Morning	Not mentioned
Duration	Not mentioned	15 minute time-boxed

3 Research Method

The motivation for our research was to increase the understanding of which factors contribute to effective teamwork because teams are the fundamental organizational unit through which agile software projects are executed. We chose grounded theory as our research method because it is considered suitable for pursuing a general understanding of a phenomenon; that is, when a researcher asks "What is going on here?" (Glaser, 1978).

3.1 Research sites

Understanding the context is important for the interpretation of the results of any empirical study (Dybå et al., 2012). Here we describe the three companies that participated in this field study. We had access to these companies through an industry-managed research project on teamwork in agile software teams. We investigated a Norwegian consulting company, an International telecommunications company and an International software company, which we denote, respectively, Alpha, Beta and Gamma.

Alpha is a company with 350 employees. We studied two distributed, closely collaborating Kanban teams who worked on the same project. One team was located in Norway (Alpha 1) and one in Poland (Alpha 2). The team members located in Poland were employees of the project's client, a Fortune 500 industrial company. The project members worked on maintenance and extensions of a content management system.

Beta has 700 employees in 20 countries. We studied two Scrum teams in Norway and five Scrum teams in Malaysia. The teams were working on different projects, which had iterations lasting from two to four weeks.

Gamma has 150 employees in four organizational units. We observed one team in Norway and two in the United Kingdom. The goal of the Scrum project was to develop an engineering software product for the oil and gas industry and the length of the iterations was usually three weeks.

Table 2 shows an overview of the investigated teams. In the time period of interviews in each of the teams the teams were static (one person we interviewed had handed in his resignation).

Table 2: The investigated teams

Comp-any	**Team**	**Location**	**Team members**	**Distribution**	**Inter-viewed**	**Obser-ved**
Alpha	1	Norway	10	Co-located	✓	✓
	2	Poland	9	Co-located	✓	✓
Beta*	1	Norway	9.5	Distributed	✓	✓
	2	Norway	8.5	Co-located	✓	✓
	3	Malaysia	10	Co-located	✓	✓
	4	Malaysia	3	Co-located	✓	
	5	Malaysia	8	Co-located	✓	
	6	Malaysia	6	Co-located	✓	
	7	Malaysia	9	Co-located	✓	
Gamma	1	Norway	10	Co-located	✓	✓
	2	UK	9	Distributed	✓	✓
	3	UK	8	Distributed	✓	✓

*Teams 1 and 2 in Beta shared one team member.

3.2 Choice of research method: Grounded theory

Grounded theory is defined as “a general methodology of analysis linked with data collection that uses a systematically applied set of methods to generate an inductive theory about a substantive area” (Glaser, 1992, p. 16). This methodology aims to develop a theory rather than extend or verify existing theories.

Grounded theory was introduced in 1967 with the publication of Glaser and Strauss’s (1967) book, *The Discovery of Grounded Theory*. The method later evolved into two versions with separate terminology and processes as Glaser and Strauss developed different perspectives (Goulding, 1998). These two versions are known as the Glaserian and the Straussian methods of grounded theory, respectively. Among other factors, they differ with respect to the formulation of research problems, data analysis and coding techniques. According to Strauss and Corbin (1990, p. 34), a researcher should begin a grounded theory study by defining a research problem. In contrast, Glaser advises moving into an area of interest without establishing such a definition (Glaser, 2001, p. 21). Furthermore, Glaser describes selective coding as occurring early in the analysis, when the core category has been identified, while Strauss and Corbin describe selective coding as occurring towards the end of the analysis, with the purpose of selecting the core category (Glaser 1992, p. 75). Glaser argues that the theory should only explain the phenomenon under study, while Strauss and Corbin describes coding matrixes as explaining the phenomenon beyond the immediate field of study (Goulding, 1998). Glaser (1992, p. 62) criticizes the use of these coding matrices, claiming that they lead to theories based on preconceptions because data is “forced” into categories.

Because Glaserian and Straussian grounded theories have substantial differences, it is important that researchers explicitly state which method they use. We decided to follow Glaser’s method since we wanted to approach the field with no research questions but rather a general interest in it; we wanted to let the concepts and categories emerge from the data. Furthermore, we found the Glaserian grounded theory more flexible because it is less prescriptive than the Straussian method.

However, throughout the study, data collection and analysis occurred within an iterative process. Figure 2 depicts our research approach.

3.3 Data Collection Techniques

We used three different data sources; Table 3 shows the data techniques we used.

Table 3: Data collection

Technique	**Number**	**Description**
Interviews	8 in Phase 1, 52 in Phase 2	We conducted semi-structured interviews with open-ended questions.
Observations of DSMs	9 in Phase 1, 70 in Phase 2	We made notes from all DSMs we observed. Thirteen of these observations were recorded and transcribed word by word.
Questionnaires	19 in Phase 2	Project members in Alpha answered a questionnaire anonymously.

Table 4: Interviewees

Comp-any	**Team**	**Inter-viewees (*N*)**	**Roles of interviewees*(*N*)**	**Age, mean (median)**	**Years in company, mean (median)**
Alpha	1	9	A (3), D (4), PM (1), T (1)	34.7 (35.0)	6.2 (5.5)
	2	6	A (1), D (2), PM (1), T (1), TL (1)	31.5 (31.0)	3.1 (3.3)
Beta	1	4.5	A (1), D (1), SM (1), T (1), TW (0.5)	44.9 (45.0)	10.7 (10.0)
	2	5.5	D (3), SM (2), TW (0.5)	41.5 (40.5)	10.9 (10.5)
	3	9	D (5), PM (1), PO (1), SM (1), T (1)	32.9 (32.0)	2.3 (2.5)
	4	3	D (1), SM (1), TL (1)	32.3 (31.0)	2.3 (1.0)
	5	4	D (1), PM (1), SM (1), T (1)	32.3 (30.5)	5.6 (2.8)
	6	3	A (1), D (1), TL (1)	32.7 (32.0)	1.0 (1.0)
	7	4	D (2), TL (1), PO (1)	35.8 (37.5)	4.4 (3.0)
Gamma	1	9	A (2), D (6), PM (1)	42.0 (38.0)	10.5 (10.0)
	2	2	D (1), PO (1)	44.0 (44.0)	14,5 (14,5)
	3	1	D (1)	59.0 (59.0)	14.0 (14.0)
			Sum: A (8), D (28), PM (5), PO (3), SM (6), T (5), TL (4), TW (1)	**Avg: 37.0 (35.0)**	**Avg: 6.5 (4.5)**

*A = Architects; D = Developers; PM = Project Managers; PO = Product Owners; SM = Scrum Masters; T = Testers; TL = Team Leaders; TW = Technical Writers

3.3.1 Interviews

This study involved 60 interviews across 12 teams profiled in Table 4. Eight of the participants in Alpha 1 were interviewed in Phase 1; all other interviews were conducted in Phase 2. The teams ranged in size from three to eleven persons. Nine different roles were interviewed in total. If interviewees had more than one role, the table shows their main role.

The participants gave their consent for the interviews to be recorded and agreed to the publication of the results subject to anonymity. The interviews varied between 25 and 96 minutes. In most of the interviews, at least two researchers participated. One asked questions, and one or more others took notes and asked additional questions at the end.

The interview guide used in our semi-structured interviews consisted of four parts. In the first part, we introduced ourselves and assured the interviewee of confidentiality. The topic of investigation presented to the interviewees was “teamwork in agile projects”. The second part comprised “warm-up” questions regarding the interviewee’s background, experience and current activities. The third part involved the main interview. This part was modified as the research progressed in Phase 2 to comply with the theoretical saturation of the core category (DSM), which is normal when using grounded theory. The fourth part included closing questions and provided an opportunity for the interviewee to ask questions and make additional comments. Appendix A shows how the interview guide looked in the beginning of the data collection in Phase 2. All the interviews were transcribed word by word, mostly by the first author, partly by an MSc student, whose transcripts were validated by the first author.

3.3.2 Participant Observation

Our participant observation was guided by a protocol based on Spradley (1980) that contained questions to be answered by the researcher; see Appendix B. The protocol was initially proposed by the first author and reviewed by the two other authors. Information recorded while observing meetings included names and roles of the attendees, time of the day, promptness in starting, duration, type of discussions, leadership and facilitation, format (who was sitting or standing), communication channel (phone or video), the number of participants present in each location, language and atmosphere. The observer wrote notes either during the meeting or immediately after. On many occasions, two observers were present to ensure the reliability of the information captured. We sometimes took pictures to document the meeting setting. When we started to observe meetings in Phase 2 in Company Beta, we decided to audiotape the meetings. However, for capacity reasons, we only managed to audiotape and transcribe the first thirteen meetings. Note that transcribing a meeting is more time consuming than transcribing an interview, particularly because keeping track of who is speaking is hard.

3.3.3 Questionnaire

A questionnaire with statements about DSMs (Table 5) was distributed to Alpha 1 in September 2010 and to Alpha 2 in October 2010. The statements were scored on a 5-point Likert scale from strongly disagree to strongly agree, except for the last two questions, which had three blank spaces each. Similarly to the development of the interview guide and observation protocol, the questionnaire was proposed by the first author and reviewed by the two other authors.

Table 5: Questionnaire

Daily stand-up meetings
• I am satisfied with the outcomes of our stand-up meetings.
• I look forward to our stand-up meetings.
• I feel energized and ready to get down to work after a stand-up meeting.
• I think our stand-up meetings improve our development process.
• I feel that our stand-up meetings contribute to better teamwork.
• I feel that it is worth spending time on stand-up meetings given the results we get from them.
• List three positive things about the stand-up meetings:
• List three negative things about the stand-up meetings:

3.4 Data Analysis Techniques

3.4.1 Coding

We followed Glaser's two sequential stages of substantive coding: *open* and *selective.* Open coding sets the direction of the research by identifying a core category and serves as the initial step of the theoretical analysis in grounded theory (Glaser, 1992, p. 39). Selective coding continues the process and is limited only to categories related to the core category.

Glaser (2011, p. 3) uses the terms *code*, *concept* and *category* synonymously and explains that they all "refer to conceptualizing an emergent pattern." For the sake of clarity, we refer to three different levels of data abstraction: *Codes* are assigned to statements at the first level of abstraction; groups of codes are *concepts* at the second level; and groups of concepts are *categories* at the third level.

Theoretical coding involves detecting relationships between codes, concepts and categories and may occur throughout the whole study (Hernandez, 2009). Glaser (1978, 1998, 2005) has identified 50 different theoretical coding families to assist researchers in conceptualizing how categories and their properties may relate to each other. The use of these coding families increases completeness and relevance for the grounded theory (Glaser 2005, p 70).

3.4.2 Constant Comparative Method

The constant comparative method is a key part of grounded theory and involves comparing codes and concepts to produce a higher level of abstraction (Glaser, 1992, p. 39). More specifically, codes are compared with other codes to produce concepts, codes are also compared with concepts to produce new concepts, and concepts are compared with other concepts to produce categories.

3.4.3 Memoing

Memos are written notes to record reflections on the data and codes and their relationships as they occur to the analyst while coding and writing (Glaser, 1978, p. 83). Our memos usually consisted of a few statements or questions and were written throughout the whole study. An example of a memo written during the analysis had the title "Many participants" and contained the question "Does the number of participants affect the duration?" The memos were written and managed by the first author throughout the study using the tool MacJournal.[2]

3.4.4 Quantitative Measurement of Meeting Attitude

To obtain a quantitative measure of interviewees' attitude about meetings, the first two authors studied all interview transcripts and independently scored the attitude towards DSMs on a Likert scale from 1 to 5 (strongly negative to strongly positive). The authors had substantial agreement (weighted kappa = 0.72) (Landis and Koch, 1977).

[2] MacJournal is a registered trademark of Mariner Software, www.marinersoftware.com.

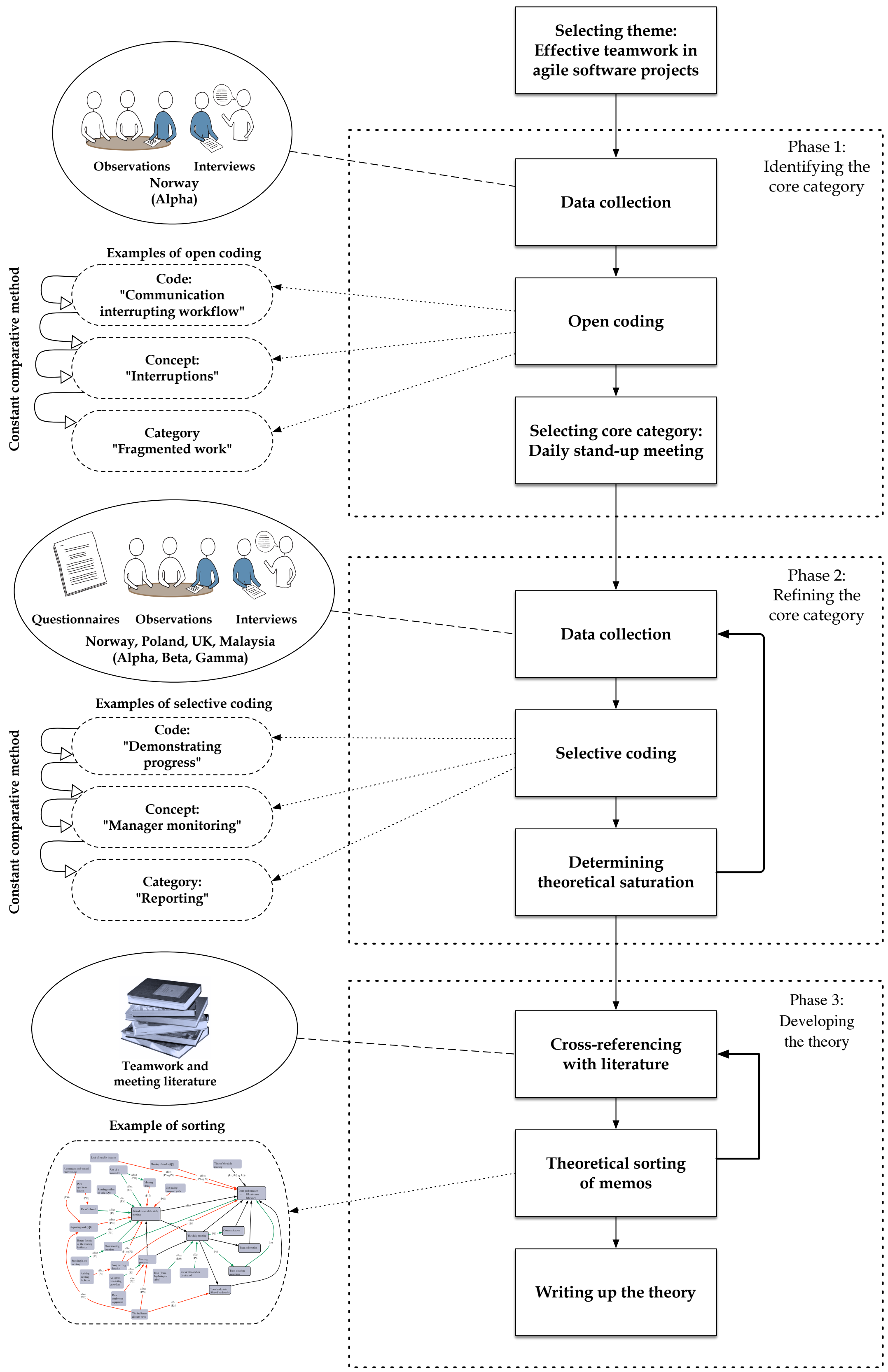


Figure 2: Our grounded theory research process

3.5 Phase 1: Identifying the Core Category

3.5.1 Data Collection

The theoretical sampling process in classic grounded theory begins with initial data collection and analysis before any core category has been identified (Glaser 1978). By adhering to the Glaserian method, we established our general area of interest as "effective teamwork in agile software development projects." As the study progressed, our particular interest, that is, the core category was identified as "DSMs in agile software development projects." We chose to conduct the first interviews in Alpha 1 since they had been practicing agile methods for almost a decade. The interviews in Phase 1 lasted between 25 and 61 minutes (average 39 minutes) and was conducted and transcribed by the first author.

3.5.2 Open Coding

With systematic reading and coding of all interview transcripts, initial categories emerged. We started the coding as soon as the first interviews were transcribed using the qualitative research tool NVivo.[3] Adhering to the prescribed method of open coding (Glaser, 1992, p. 48), we had no preconceived codes at this point. Each transcript was coded in its entirety in detail, because at this early stage in the process, we could not know which data would be relevant. We used the constant comparative method, as illustrated on the left side of Figure 2. We first compared codes within the same interview. We then compared codes from one interview with codes from other interviews, data based on observations, and our memos. Our open coding process generated 46 codes. Table 6 shows examples of codes that were assigned to statements in the open coding process.

3.5.3 Selecting the Core Category

Open coding comes to an end when a core category is selected (Glaser, 1992, p. 39). Glaser (2001, p. 200) emphasizes that researchers should tolerate confusion in the open coding process, because discovering a core category may be challenging.

If more than one core category is found, a researcher must select one of them, since grounded theory centers on one core category. Glaser (2001, p. 201) states that selecting the core category may seem like a big commitment, but fear of selecting an unsuitable core category may result in the research taking too long: "It is best to test out a core category and then another if the first does not work, than to drift two years in open coding." If other interesting core categories are also identified, they can be investigated in separate studies.

Three potential core categories emerged from our open coding: *Fragmented work*, *Communication* and *The daily stand-up meeting*. We selected the last one as our core category because it related to most other categories in a meaningful way. Furthermore, all participants expressed concerns related to the practice, and they were very eager when they talked about it.

[3] NVivo is a registered trademark of QSR International, www.qsrinternational.com.

Table 6: Examples of open coding

Role in Alpha	Statements	Codes
Architect	"No, the teamwork was not very effective last week. I received 3-4 phone calls a day from the other site. They sort of interrupt me in the middle of what I am working on. (…) I think I have to switch between tasks a lot, and after an interruption, I have to start over what I was doing."	*Communication interrupting workflow*, *Task switching*, *Start over*
Developer	"Our biggest challenge is communication (…) I really look forward to getting the videoconference equipment so that we can have daily face-to-face meetings with the other site. Often the interruptions caused by all the questions on MSN [chat client] reduce my efficiency and make me frustrated. Another benefit of having the stand-up with the other site will be that we will be in charge of the meeting, and since we have a limit of work tasks in progress, maybe we can affect the other team positively. Now they start too many tasks at once."	*Daily face-to-face meetings*, *Frustrated by interruptions*, *Work in progress*
Project Manager	"For many years, we have debated how to run the daily stand-up meeting. Some people think that since we work on different tasks, they don't need an overview of what others are doing. They find the meeting irrelevant and to last too long. So we have tried to reduce the time spent in the meeting, but then others complain that they lose an overview of what is happening in the project."	*How to run the daily stand-up meeting*, *Overview of what others are doing*

3.6 Phase 2: Refining the Core Category

3.6.1 Data Collection

The aim of theoretical sampling is to ensure that new data contribute to theory development (Glaser, 1992, p. 101). Selecting new interviewees and sites for observation follows from the coding process results; it is not based on random selection.

Consequently, we did not plan where we would collect subsequent data; we did not know in advance where the research would lead. Once we identified the DSM as the core category, however, we decided to gather data from Alpha 2. We thought it would be valuable to investigate DSMs from the perspective of both sites in the multicultural, distributed project. Furthermore, Alpha 2 was about to start using video equipment in DSMs, which we thought could generate interesting insights into distributed DSMs.

When analyzing the data collected from Alpha with regards to the emerging theory of DSMs, we did not know the theoretical concepts that were specific to this company. Thus, we had to collect data from other companies. We chose Beta and Gamma, which were also part of the research project on teamwork referred to in Section 3.1, because both companies conducted DSMs regularly.

3.6.2 Selective Coding

In the selective coding, we coded only the transcript passages that were pertinent to DSMs. The coding involved extensive use of the constant comparative method. As an example, Table 7 shows statements that were coded as *Demonstrating progress* . After *DSM attitude* emerged as a category, all statements were also coded as either positive or negative (towards DSM), as shown in the last column.

Table 7: Examples of selective coding

Company	Role	Statements	Code	Attitude
Beta	Scrum Master	"Everyone knows that they have to report something. So they need to do something."	Demonstrating progress	Positive
Gamma	Developer	"It motivates you to work because in the meeting, you have to tell what you have done."	Demonstrating progress	Positive
Beta	Developer	"What I don't like with the daily meeting is reporting progress when you haven't had much progress from the day before; it is a short interval to report on."	Demonstrating progress	Negative
Gamma	Developer	"You can feel the pulse go up because you are supposed to talk about the things you have done and show progress."	Demonstrating progress	Negative
Gamma	Project Manager	"Developer X always tells a lot of details. I think he wants to prove that he has done a lot of work."	Demonstrating progress	Negative

3.6.3 Determining Theoretical Saturation

The selective coding continues until the researcher has sufficiently elaborated the core category and its connections to other relevant categories (Glaser, 1978, p. 53). In February 2013, we reached the point at which we believed that additional data collection would not generate any new results (i.e., the point of theoretical saturation).

3.7 Phase 3: Developing the Theory

3.7.1 Cross-referencing with literature

We consulted meeting and teamwork literature for relevant theories. The theory that we found most relevant was the Adaptive Structuration Theory (AST). AST examines organizational change facilitated by different types of structures for social action (DeSanctis and Poole, 1994). Although AST links structure primarily to the structure provided by advanced information technologies, structure can also be provided by processes, procedures and organizations (DeSanctis and Poole, 1994). Structures consist of *structural features* and a *spirit.* The *spirit* is the researchers current interpretive account (based on multiple sources of evidence) regarding the values and goals of the practice. AST posits that how a practice is used is impacted by the *faithfulness* to which a team uses the practice in keeping with the spirit in which it is meant to be used, the team's *attitudes* toward the practice, and the team's *level of consensus*.

We also consulted other theories and frameworks, including the Theory of Activity Regulation (Zijlstra et al., 1999), Conservation of Resources Theory (Hobfoll, 2001), Collaborative Performance Framework (Bedwell et al., 2012), Group Effectiveness Model (Cohen and Bailey, 1997), Meeting Satisfaction Model (Reinig, 2003), Shared Mental Models Theory (Cannon-Bowers et al., 1993) and Team Situation Awareness (Endsley, 1995). Consulting this literature helped us perform a well-founded analysis of the data in the last phase of the grounded theory process (see Figure 2).

3.7.2 Theoretical Sorting of Memos

Theoretical sorting is an essential step in the grounded theory process in which one is supposed to do most of the theoretical coding (Section 3.4.1). The theoretical sorting of memos is the key to presenting the theory to others in words and writings (Glaser, 1992, p. 109). At the beginning of the theoretical sorting of memos, we printed all memos on paper, and we looked for similarities and connections in the ideas. For each memo we asked, as Glaser suggests (1978, p. 123), "where does it fit in?"

We did most of the theoretical coding in this stage of the analysis. (We identified a few relationships in earlier stages.) We explored the coding families *Six C's*, *Social arena*, *Dimensions*, *System parts* and *Model*. We found that the *Model* family was the most suitable one to explore our data. When using this method, the researcher models the "theory pictorially by either a linear model or a property space" (Glaser, 1978, p. 81). To model the memos and their relationships, we used the diagramming tool Omnigraffle Pro[5]. An example of a cycle of memo sorting using the Model coding family is depicted in Figure 3, in which each box represents the title of a memo, and the arrows represent relationships between memos. For example, at this point in time, a memo was titled "Many participants". The red arrow from this memo to the memo "Spending time (duration)" illustrates the proposition that many participants in a meeting cause the meeting to last longer. New memos were created based on theoretical sorting and cross-referencing with the literature and included in the next cycle of sorting.

3.7.3 Writing up the Theory

A grounded theory should use the fewest possible concepts to explain as much variation as possible in the phenomenon under study with the greatest possible scope (Glaser 1978, p. 125). The propositions that we decided to include in the final theory were supported by statements provided by at least seven interviewees. We found this as an appropriate threshold because increasing the threshold to eight would explain

[5] Omnigraffle Pro is a registered trademark of The Omni Group, www.omnigroup.com.

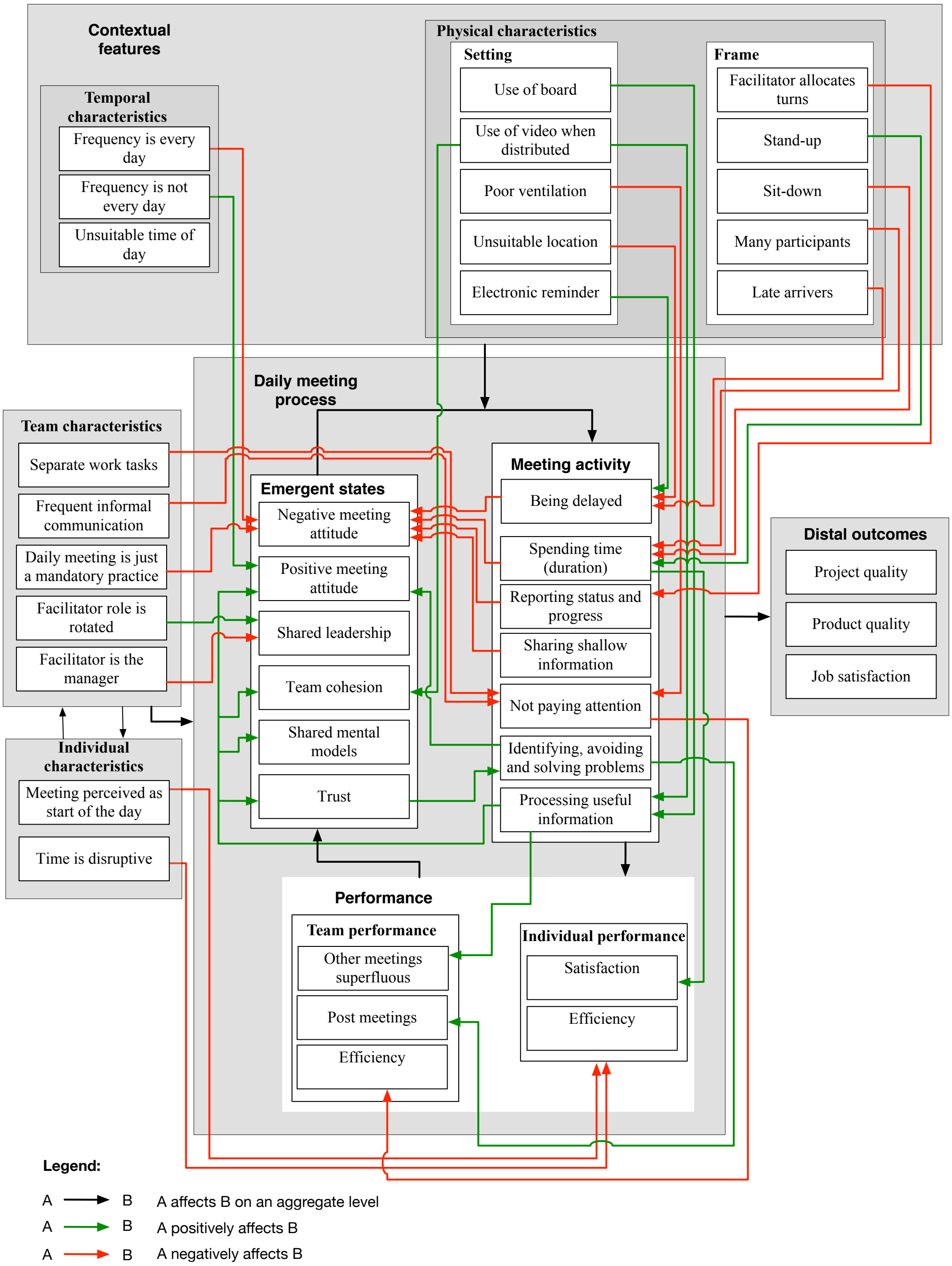


Figure 3: Snapshot of sorted memos during theoretical sorting

too little variation, while a threshold of six would lead to a too comprehensive theory. Still, there could be important propositions that were not explicitly supported by as many as seven interviewees. Therefore, to identify such important propositions we analyzed our observational data. We found only one proposition that we found to be important that was not mentioned by at least seven interviewees that is, “Standing in the meeting (2)”; see Table 9 and Proposition 2a in Section 4.2.

We pursued this goal in creating the DSM theory described in the next section. We propose this theory as a starting point for understanding the key constructs and relationships of DSMs in agile software development projects, adhering to the principle that a grounded theory is readily modifiable (Glaser, 1978, p. 129).

4 Results

This section describes the principal results of the study in the form of an initial theory of DSMs. The theory describes how certain team characteristics and physical characteristics affect the meeting process, which in turn affects the participants' attitude towards DSMs. The constructs and propositions of the theory are depicted in Figure 4 and listed in Table 8. The table follows the four-component structure as outlined in Sjøberg et al. (2008). The *constructs* are the basic elements of the theory, the *propositions* are the interactions among the constructs, the *explanations* describe why the propositions are as specified and the *scope* of the theory describes the universe of discourse in which the theory is applicable. This section further describes the theory, including selected measurements and quotations drawn from the interviews to show how the theory is grounded in the data.

Table 9 gives an overview of the positive and negative opinions stated by the interviewees on each of the concepts. The number in the parenthesis after the opinions indicates how many statements expressed that opinion. Italics indicate that an opinion was stated by at least seven interviewees. For example, the two statements that constitute "Demonstrating progress (2)" under category, "Reporting progress" in Table 9 can be found in the first two rows of Table 7. Table 10 gives an overview of the positive and negative aspects of DSMs as expressed in the questionnaires.

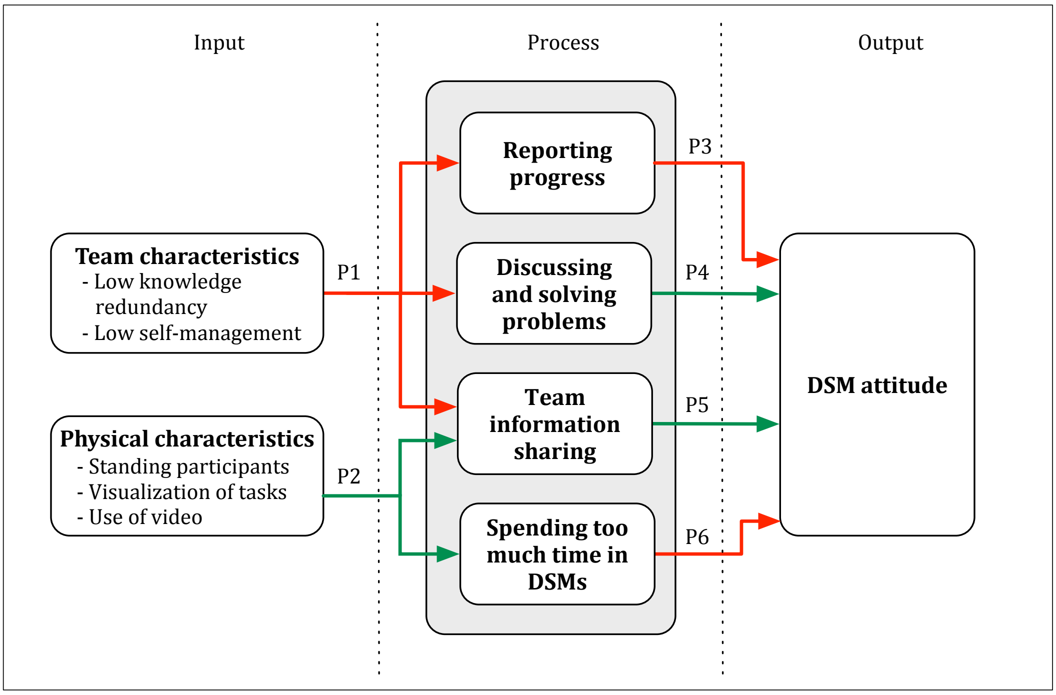


Figure 4: A theory of DSM

Table 8: Constructs, propositions, explanations and scope of the theory

Constructs	
C1	Team characteristics
C2	Physical characteristics
C3	Reporting progress
C4	Discussing and solving problems
C5	Team information sharing
C6	Spending too much time in DSMs
C7	DSM attitude
Propositions	
P1	Team characteristics negatively affect the DSM process. a) Low level of knowledge redundancy negatively affects the DSM process in the form of less team information sharing and problem discussions and solutions. b) Low level of self-management negatively affects the DSM process in the form of more reporting of progress and less team information sharing.
P2	Physical characteristics positively affect the DSM process. a) Standing in DSMs decreases duration. b) Visualizing tasks positively affects the DSM process in the form of more team information sharing. c) Using video positively affects the DSM process in the form of more team information sharing.
P3	Reporting progress negatively affects the attitude towards DSMs.
P4	Discussing and solving problems positively affect the attitude towards DSMs.
P5	Team information sharing positively affects the attitude towards DSMs.
P6	Too high frequency and too long duration of DSMs negatively affect the attitude towards DSMs.
Explanations	
E1	A low level of knowledge redundancy in the team negatively affects the DSMs; participants are uninterested in what others are doing because it does not concern them. A low level of self-management makes it easy for authoritative managers to use the meeting to obtain status information, mainly useful to themselves.
E2	Standing format decreases the duration of the meeting because standing is less comfortable than sitting. The sit-down meetings lasted 63% longer than the stand-up meetings. Using a board or other visualization tools positively affects the participation because people can relate what participants say to the tasks for the iteration. Using video in a distributed meeting positively affects the participation because people pay more attention than if using a phone. Then also the communication is positively affected.
E3	When participants do not pay attention or are uncomfortable in the meeting caused by reporting progress, their attitudes towards the meeting are negatively affected.
E4	When participants are supported in identifying, avoiding and solving problems in a DSM, attitudes of the participants are positively affected because spending time on these activities is perceived as useful.
E5	Sharing information within a team makes the participants obtain an overview of who is doing what.
E6	Frequency as high as every workday and long duration negatively affect attitudes because participants feel that their time is wasted.
Scope	
S1	Agile software development projects.

Table 9: Positive and negative opinions from interviews

Category, *Concept*	***Positive opinion (code)***	***Negative opinion (code)***
Team characteristics		
Interaction		High within-team interaction making DSM superfluous (3)
Knowledge redundancy		*Being uninterested in what others are saying (12)*
Self-management	Manager obtaining status information	*Reporting status to the manager (7)*
Purpose of conducting DSM		Conducting just because guideline or policy says so (6) Not being aware of the purpose of the meeting (3)
Physical characteristics		
Format	Standing in the meeting (2) Sitting in the meeting Fixed seating	Many don't like standing Combination of people standing and sitting
Setting	*Seeing each other on video (7)* No distributed meetings	Poor conference equipment (2) Lacking a suitable location (5)
Visualization tools	*Using boards in the meeting (9)* No board in the meeting	Poor synchronization between boards (5)
Participants	Observers acquiring good understanding of the team (3) Good attendance Speaking English in the meeting (2)	Observers speaking when not supposed to People working from home not attending Too many people attending
Reporting progress		
Demonstrating progress	Demonstrating progress (2)	Demonstrating progress (3)
Paying attention		*Being disengaged or uncomfortable because of reporting of progress (15)*
Discussing and solving problems		
Identifying and avoiding problems	Identifying and avoiding problems (6) Uncovering dependencies (2)	
Back-up behavior	*Solving problems (getting help) (13)* Asking for help	
Making decisions	Making decisions (2)	
Team information sharing		
Team monitoring	*Obtaining an overview of what others are doing (22)* Sharing information (5) Sharing knowledge (3) Praising (2) Learning Showing interest	Not understanding what others are saying (3) Irrelevant and shallow information (6) Long technical discussions (5)
Communication	Forcing individuals to communicate (2) Answering the three questions (4) Mutual trust	Too formal communication Answering the three questions (4) No mutual trust
Team orientation	Team spirit and being involved (4) Feeling like a part of a team Socializing (seeing each other) (2) Erasing boundaries (2) Being more open	Not feeling like a part of a team (2)
Coordination	Coordinating work (3) Inviting distributed members to post meeting (2) Being aware of the need of a post meeting (3)	No common goal (3)
Turn-taking procedure	Round-robin (4) Facilitator deciding the tasks to discuss (2)	Round-robin (2) Facilitator deciding next speaker or task to discuss (2) No one taking the role as a facilitator
Temporal characteristics		
Frequency	Conducting the meeting every day Making other meetings superfluous (5) Attendance is a habit	*Every day is too frequent (8)* Attending two DSMs each day Conducting DSMs only when the manager is present (3) Stress caused by time spent in addition to other meetings
Time of day	A break from work (2) Conducting the meeting at start of the day (3) Don't need to be held in the morning	Interruption of workflow (6) The meeting is held too early in the morning
Starting promptness	Using an alarm to signal start Punishing latecomers (2) Agreeing on importance of being on time	Relying on facilitator to signal start Waiting for latecomers before starting People arriving late to the meeting (3)
Duration	*Keeping the meeting short (7)* Using a countdown-timer (2) Being allowed to leave Being efficient (3)	*Duration sometimes too long (13)* Duration sometimes too short (2)

Table 10: Positive and negative opinions of DSMs from questionnaires

Category, *Concept*	***Positive opinion (code)***	***Negative opinion (code)***
Physical characteristics		
Setting	Seeing each other on video	Poor conference equipment (2)
Visualization tools		Tools out of sync with real status
Participants		Too many people attending Not all team members attending (2)
Reporting progress		
Manager monitoring		Project Manager not giving status
Attention		Being disengaged
Purpose		Not being aware of the purpose of the meeting
Discussing and solving problems		
Identifying and avoiding problems	Identifying and avoiding problems (3) Revealing problems	
Back-up behavior	Solving problems (receive help) (6) Asking for help (2)	Being unprepared (2)
Team information sharing		
Team monitoring	Obtaining an overview of what other team members are doing (6) Obtaining a better understanding of who does what Setting focus Knowing the current situation (3) Sharing task information Sharing knowledge	Repetitive (2) Irrelevant and vague information (3)
Communication	Focusing on getting things done Being flexible	Forcing individuals to talk Not everyone talking
Team orientation	Being involved Socializing (seeing each other) (4)	
Coordination	Making it easier to contact team members later Knowing that team members are available for discussion	Assigning tasks
Temporal characteristics		
Duration	Being efficient	Duration sometimes too long (7)

4.1 Team Characteristics

The members of nine teams were co-located in an open-space office. The team members often talked to each other during the workday. The three teams that had distributed team members also communicated frequently by using chat and e-mail. A high degree of informal communication outside DSMs reduced the necessity and benefits of the meeting. One developer said, “I can say that today there was nothing interesting on the stand-up, because everybody said something which I already knew.” An architect stated, “If someone faces an obstacle, we are supposed to flag this at the stand-up, but my experience is that if people are stuck, they don’t wait until the next day but try to find someone who can help right away.” A project manager said, “Team members have asked me why we have stand-up meetings, because they feel that they already know what is happening in our team.”

Most of the teams promoted knowledge redundancy, which means that the same knowledge is shared among more team members (Rindfleisch and Moorman, 2001) but found it difficult to apply it in practice, because many team members had expertise in different technical areas and therefore had specialized roles. Low knowledge redundancy in the team negatively affected DSM processes in that the team members did not pay attention in the meeting, which resulted in less interaction. Twelve interviewees stated that they were not interested in what others were saying in the meeting because their roles or tasks were not affected by the information shared. One developer stated, “In theory, everyone in the team should be able to solve every task, but in practice you work in the area you know. […]

Personally, I am not very interested in what others are doing." A technical writer said, "The daily meetings do not work that well because they take too much time. We can spend a lot of time talking about solutions, and I think that is boring because it is irrelevant to me." Our findings confirm a previous study that found that over time, as the team members' roles became increasingly specialized, the extent of shared mental models and communication decreased (Levesque et al., 2001).

P1a: Low level of knowledge redundancy negatively affects the DSM process in the form of less team information sharing and problem discussions and solutions.

A premise in agile development is that software teams are self-managed. In the present study, all teams had applied Scrum for several years, but the degree of self-management was generally low. Teams whose members were not in control of DSMs indicated a low degree of self-management. A quote from a developer illustrates this point: "We have stand-up meetings only when the project manager is here. He probably feels that he needs an overall picture. So, I think the stand-up meeting means a lot to him at least."

Another indicator of low self-management in several teams was the Scrum Master's control of the allocation of speaking turns in the meeting. In contrast, in a few teams, high self-management was indicated by turn-taking following a round-robin approach; that is, the person standing at one point of the semicircle started, and the other participants then continued in a counter-clockwise sequence. Another indication of high self-management in these teams was the rotation of the Scrum Master role.

The leadership style of the Scrum Master sometimes made it harder for the team to be completely self-managed. For example, a Scrum Master, who was also a project manager, was aware that he affected the DSMs: "I don't think they would conduct the daily meetings in the same way if I had not been present. They need someone who has the overview and asks the unpleasant questions. I don't believe there is such a thing as a completely self-organized team. There must be someone who is firm and gets things done." Interestingly, we observed that when this project manager was relocated and a developer took on the Scrum Master role, the level of self-management increased and the average meeting duration decreased from 27 minutes (median 28) to 19 minutes (median 17). We observed that the meetings became less formal and that team members shared more information among each other instead of reporting to the manager. Our observations indicate that the reduction in meeting time was mainly due to less reporting of status.

P1b: Low level of self-management negatively affects the DSM process in the form of more reporting of progress and less team information sharing.

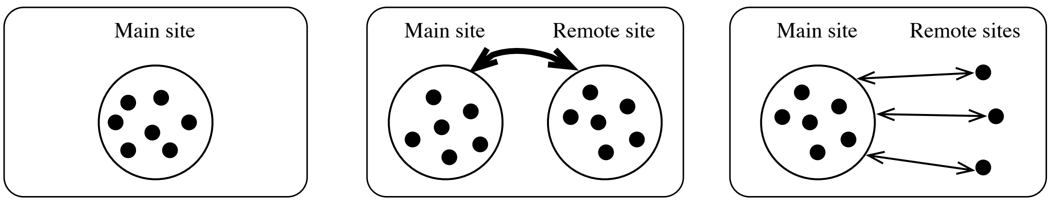
4.2 Physical Characteristics

Table 11 shows the physical characteristics of the observed meetings. The three different meeting set-ups that we observed are shown in Figure 5. Alpha had both co-located and distributed meetings. In the beginning, Alpha 1 had co-located meetings with only Alpha 1 team members attending (set-up 1). When the team from Poland (Alpha 2) visited Norway, these two teams had co-located meetings. When Alpha 2 went back to Poland, the two Alpha teams had distributed meetings with the use of video (set-up 2). When Beta and Gamma had distributed meetings, they did not use video but several team members attended using phone (set-up 3). All the roles of the teams were represented in the DSMs with the exception of the PO in Alpha who was located in Switzerland.

Table 11: Physical characteristics of the observed DSMs

Comp-any	Team	Location	Language	Set-up	Board	Main turn-taking procedure	Roles present*
Alpha	1	Office space	Norwegian	1	Interactive whiteboard	Task-oriented	A, D, PM, T
	1 and 2 same site	Office space	English	1	Interactive whiteboard	Task-oriented	A, D, PM, T, TL
	1 and 2 different sites-	Office space	English	2	Interactive whiteboard	Task-oriented	A, D, PM, T, TL
Beta	1	Meeting room	Norwegian	3	Physical board	Facilitator allocated	D, SM, T, TW
	2	Meeting room	Norwegian	3	None	Facilitator allocated	D, SM, T, TW
	3	Meeting room	English	1	Backlog on projector	Round-robin	D, PO, SM, T
Gamma	1	Meeting room	Norwegian	1	None	Round-robin	A, D, PM
	2	Meeting room	English	3	None	Speaker selected next	A, D, PO, SM
	3	Meeting room	English	3	None	Facilitator allocated	A, D, SM

*A = Architects; D = Developers; PM = Project Managers; PO = Product Owners; SM = Scrum Masters; T = Testers; TL = Team Leaders; TW = Technical Writers.



Set-up 1: Fully local

Set-up 2: Distributed over two sites with video

Set-up 3: Distributed over several sites with phone

Figure 5: Meeting set-ups of DSMs

The average number of participants in the observed meetings varied among the teams from 6.9 to 16.7 with an overall average of 10.7, as shown in Table 12. A general recommendation is that the number of participants in a meeting should be as low as possible, but high enough to represent many viewpoints (Romano and Nunamaker, 2001). Doyle and Straus (1993) suggest that a meeting of seven to fifteen participants is ideal for decision-making and problem solving because the meeting is then large enough to allow for a facilitator but small enough to be informal and enable all participants to be involved. We found a negative, nonsignificant correlation between the average number of participants (4^{th} column in Table 12) and the average score of these participants' attitude towards DSMs (Figure 7) (Spearman $\rho = -0.33$, $p = 0.29$). This finding is consistent with research on group size, which indicates that the smaller the group, the more likely participants are satisfied with the group meeting (Hare, 1952).

Table 12: Number of participants and proportion standing

Comp-any	**Team**	***N* meetings**	**Avg. number of participants (SD)**	**Avg. number of participants main site (SD)**	**Avg. number of remote participants (SD)**	**Avg. proportion of team standing, %**
Alpha	1	6	10.3 (2.2)	10.3	0	100
	1 and 2 same site	3	16.7 (2.1)	16.7	0	100
	1 and 2 different sites	10	16.3 (1.8)	8.7 (3.1)	7.6 (2.4)	100
	Team mean	**7**	**14.4 (3.6)**	**11.9 (4.2)**	**2.5 (4.4)**	**100**
Beta	1	18	8.6 (2.2)	6.7 (1.8)	1.9 (1.5)	58
	2	24	6.9 (1.3)	5.8 (1.7)	1.0 (1.0)	52
	3	15	10.4 (2.1)	10.4 (2.1)	0	100
	Team mean	**19**	**8.6 (1.8)**	**7.7 (2.4)**	**1.0 (0.9)**	**69**
Gamma	1	1	9.0	9.0	0	100
	2	1	9.0	3.0	6.0	0
	3	1	9.0	5.0	4.0	0
	Team mean	**1**	**9.0 (0.0)**	**5.7 (3.1)**	**3.3 (3.1)**	33
All	**Mean**	**9**	**10.7 (3.2)**	**8.4 (3.2)**	**2.1 (1.4)**	**67**

We found a positive, mostly nonsignificant correlation between the number of participants and duration in Beta (B1: $\rho = 0.35$, $p = 0.23$; B2: $\rho = 0.32$, $p = 0.17$, B3: $\rho = 0.60$, $p = 0.02$). In Alpha (all meetings in both teams), we found a negative correlation ($\rho = -0.3$, $p = 0.32$). The reason for shorter meetings when more people attended was that they focused the discussion on tasks instead of each individual to save time.

The primary rationale for standing in a meeting is that its duration is supposedly reduced because standing is less comfortable than sitting (Bluedorn et al., 1999). In our study, one developer said, "It is absolutely necessary that we stand. The few times we tried to sit because one team member was pregnant, the meetings lasted for 20-30 minutes.” The teams we observed that had all participants standing had meetings with an average duration of 12 minutes, while the teams that had all or some of the participants sitting had meetings that lasted 19 minutes on average; that is, we found sit-down meetings to be approximately 60% longer than stand-up meetings (independently of whether the meetings were distributed or not).

P2a: Standing in DSM decreases duration.

Using a board to visualize tasks had a positive effect on the DSM process. Interviewees indicated that what people said in the meeting was then more relevant to all of the participants. One developer explained an additional benefit: “Now we focus on how to get the tasks to move across the board, and, as a consequence, the tasks get done faster. It has affected the stand-up meetings. It is very positive that they now feel useful.” Another developer said, “I think it is important that we view the backlog in the meeting because then the team don’t forget the direction of the user stories and the sprint objective. It also reminds us that there are still a lot of tasks waiting to be done.”

P2b: Visualization of tasks positively affects the DSM process in the form of more team information sharing.

Alpha used video in DSMs (Set-up 2). The team members in the distributed project stated that it was important that the remote project members also attend DSMs and that they could see each others’ faces. One developer explained, “When I heard that we would start using video, I thought it wouldn’t change anything because I thought it was good enough to use phones. Now I can say that it actually did change a lot because if you see people’s reactions and their gestures, you immediately know whether they are listening or bored, and whether they don’t understand what you said. Then you have to use other words. Using video also helps because you immediately see who is talking and who is going to say something more.” Dorairaj et al. (2012) also found that distributed teams preferred video conferencing over telephone conferencing in distributed DSMs.

P2c: Use of video positively affects the DSM process in the form of more team information sharing.

4.3 Reporting Progress

The interviewees reported that they became disengaged or uncomfortable when responses to the question “What have you done since last meeting?” tended to become progress information to the manager. Nine interviewees stated that when the meeting was about reporting progress, many team members tuned out. An architect said, “I don’t think people listen to what is said in the stand-up meeting. Everyone wants to be finished with the round as quickly as possible to get back to work." Some participants said that they often did not pay attention because they were focused on remembering what they had accomplished since the last meeting and preparing what they were going to say. One developer stated, "In the beginning, the stand-up meetings were according to the book and that was a waste of time because it was like: 'I will work on the same as I did yesterday, and it is going okay.'"

Seven of the interviewees stated that it was unpleasant to report what they worked on. One developer explained that the previous Scrum Master made them feel as though team members were being called to account for being behind schedule: "The feeling I got in the meeting was 'Why haven't you managed it? You have to pull yourself together!' The meeting was more like blaming us for not having done what had been planned. It felt like an oral exam." The fact that people were disengaged or uncomfortable negatively affected their attitude towards the meeting. One developer stated, "Several participants do not want to involve in the discussions, so it is a waste of time." Another said, "The bad thing about Scrum is that you have to do daily updates. I don't like those meetings. It is kind of reporting. You have to report every single day."

P3: Reporting progress negatively affects the attitude towards DSMs.

4.4 Discussing and Solving Problems

Back-up behavior is a critical component of teamwork that denotes team members helping each other and showing willingness to provide and seek assistance (Dickinson and McIntyre, 1997). Thirteen interviewees stated that they appreciated sharing information about obstacles and getting help in the meeting. One developer stated, "What you did yesterday and what you plan to do today are part of the plan, whereas the impediments faced are unexpected and could mean that our planning has missed something or we did not fully understand the user story." When obstacles were discussed in the teams, the team members tended to support each other in finding a solution. One architect described the process: "There were some people in the beginning that didn't like Scrum at all. They were not comfortable with the communication. They felt that if they would share obstacles, it would have consequences for them, and they might be seen as incompetent. But when we got over that, people became more open and would share questions and problems."

P4: Discussing and solving problems positively affect the attitude towards DSMs.

4.5 Team Information Sharing

Obtaining an overview of what other team members were doing was the most frequently mentioned positive outcome of attending DSMs. In the literature, this phenomenon is called *team awareness*, which is explained as "… an understanding of the activities of others, which provides a context for your own activities"

(Dourish and Bellotti, 1992). It is not at static state, but the result of recurrent processes of information sharing within a team (Salas et al., 1995).

Attending DSMs was also said to help erase boundaries between roles and sites and to increase team spirit, cohesion and trust. A technical writer stated, “I don’t think the daily meeting works that well, but it is nice to attend even though not everything is directly related to me because I get the bigger picture. I know what others are doing, and I feel more involved, even though I cannot contribute with much.”

A positive outcome that was mentioned was that the meeting forced individuals to communicate. One developer said, “The information flow in the project is not very good; the stand-up meeting is the main source of information.” However, one developer in another team stated, “The way the meeting is today is not worth much. It is a difficult balance how much you should say. Some share too many details, while others just say, ‘I work on this.’ Something in between would be the best. But it is not easy to know what to say from day to day.”

Several DSMs in the studied teams resulted in post meetings that involved the subset of participants for whom the follow-up topic was relevant. Post meetings were considered particularly important in Alpha because the project was distributed and therefore organizing ad hoc meetings would require additional effort. One developer explained, “It is much easier to ask someone at the other site to stay a couple of minutes after a meeting than to invite them later. It might be tricky to invite someone to a separate meeting because you don’t know when they are available.” Paasivaara et al. (2008) also found that DSMs resulted in informal communication across sites after the meeting. In Alpha, the post meetings were seldom initiated by the Scrum Master, while in the other companies the meetings were usually initiated by the Scrum Master. We found no pattern that the duration of the DSM affected the likelihood of having a post meeting

P5: Team information sharing positively affects the attitude towards DSMs.

4.6 Temporal Characteristics

Table 13 shows temporal characteristics of the observed meetings. Although we refer to the investigated meetings as the “daily” stand-up meetings, not all teams conducted them daily; three teams conducted meetings three or four times a week. The average over the companies was 4.6 times a week. Four of the interviewees mentioned that they appreciated having days without the meeting. A Scrum Master said DSMs were worth the time but would appreciate conduct them less frequently. He explained, “Company policy says it is a Daily Scrum, so we have to have it often, but I think twice a week would be fine, and many of the team members agree with me.” Another Scrum Master described why they held the meeting every day: “It will require more effort to meet more seldom. Then you have to remember that it is not a meeting today, it is tomorrow. Additionally, you have to remember what you did two days ago when you inform about what you have done.”

When we asked whether there were time periods in the iterations during which DSMs were more important than in other time periods, a developer stated, "I guess its importance might get higher in the middle of the sprint, where the team already has developed part of the code, which is where new problems or potential problems are discovered."

One manager said DSMs increased the level of stress because he had to attend these meetings in addition to other mandatory meetings. This statement reflects Luong and Rogelberg's (2005) findings, which showed that a high number of meetings was associated with increased feelings of fatigue and subjective workload.

All teams held their DSMs in the morning. A tester explained that conducting meetings in the morning was important so that everyone would know what they needed to do for the rest of the day. A developer said, "The daily meeting is a way to start the day. Ten o'clock has sort of become the 'clock-in' time although some managers are not happy with that. According to company policy, all staff is supposed to arrive at work at 8:30."

We observed that the time before the meeting was often spent on tasks that did not require full concentration. One developer said, "Normally I will use the time

Table 13: Frequency, time of day and duration of the observed DSMs

Company	Team	Total number of meetings observed	Frequency, times a week	Time of day	Avg. duration in minutes (SD)
Alpha	1	6	5	09:00 a.m.	11.7 (2.6)
	1 and 2 co-located	3	5	09:00 a.m.	7.0 (3.0)
	1 and 2 distributed	10	5	09:00 a.m.	11.9 (3.7)
	All teams	**19**	**5.0 (mean)**		**10.2 (2.8)**
Beta	1	18	4	10:30 a.m.	20.2 (4.4)
	2	24	4	10:00 a.m.	24.8 (9.9)
	3	15	5	10:00 a.m.	19.0 (9.2)
	All teams	**57**	**4.3 (mean)**		**21.3 (3.0)**
Gamma	1	1	3	09:45 a.m.	9.0
	2	1	5	09:30 a.m.	13.5
	3	1	5	09:45 a.m.	18.0
	All teams	**3**	**4.3 (mean)**		**13.5 (4.5)**
Total: company level (*N* = 3)			**4.5 (mean)**		**15.0 (5.7)**
Total: team level (*N* = 9)			**4.6 (mean)**		**15.0 (5.8)**
Total: all meetings (*N* = 79)			**NA**		**18.9 (8.9)**

before the meeting to settle down, read new e-mails, update Scrum Works Pro if I forgot to update it last night, and get coffee." A project manager said the meeting could be a disruptive interruption for the team members, but it could also enforce discipline because people want to check their e-mail and read the newspaper anyway; the time before the meeting may be a natural block of time to do these things. Our findings are consistent with what Sharp and Robinson (2010) observed in XP teams: DSM "heralded the real start of the day." They found that the time between developers arriving at work and the DSM being held was used to eat breakfast, check e-mail and read the newspaper.

Many of the teams struggled to find a time that satisfied everyone. The challenge was to conduct the meeting as early as possible, but not before all team members had arrived at work. A Scrum Master described the situation as follows: "We have team members who arrive before 7 a.m. and team members who prefer to arrive at 10 a.m. I think the interruption caused by the stand-up meeting is what people find negative, so I try to conduct the meeting as early or late as possible so that the team members get the largest blocks of time to work, but this block is different for everyone."

Team members also noted that DSMs could be a disruptive interruption, with one developer saying, "The daily meeting interrupts my workflow and it takes time to resume the work after the meeting." Another stated that the meeting was particularly disruptive during intensive programming. This is consistent with what Solingen et al. (1998) found: The recovery time after an interruption was longer when the interruption occurred during programming because it requires deep concentration. However, some interviewees appreciated being interrupted. One developer said, "I believe the stand-up meetings intend to break the developers out of their zone, so they can take a breather to evaluate their situation and update the team." A break from a task may give the subconscious time to process complex problems (Jett and George, 2003).

Not starting promptly negatively affected the participants' attitude and several teams had some kind of punishment for late arrivers. We identified three reasons for a meeting starting late: waiting for people to arrive, waiting for a "critical mass" and waiting for connection to be established. Often, the meeting did not start promptly because not all team members had arrived at work. One developer said, "We start it about 10:30 every morning. But if someone is late, we give a buffer. So, once every team member is here, we'll start the standing meeting." Another reason for delay was that the participants that were present waited for a "critical mass" before they started the meeting. Furthermore, meetings were also delayed due to time spent connecting with distributed team members by phone. In Alpha, the team members were reminded of the meeting by a signal given automatically at 9 a.m. by an electronic board. This forced everyone at work to attend the meeting on time since it was hard to ignore the loud signal. A project manager in Alpha described an additional benefit: "I feel it is more nagging if you as a Scrum Master or project manager go around and say 'Let's have the morning meeting.' The alarm is a mechanism that makes the job easier."

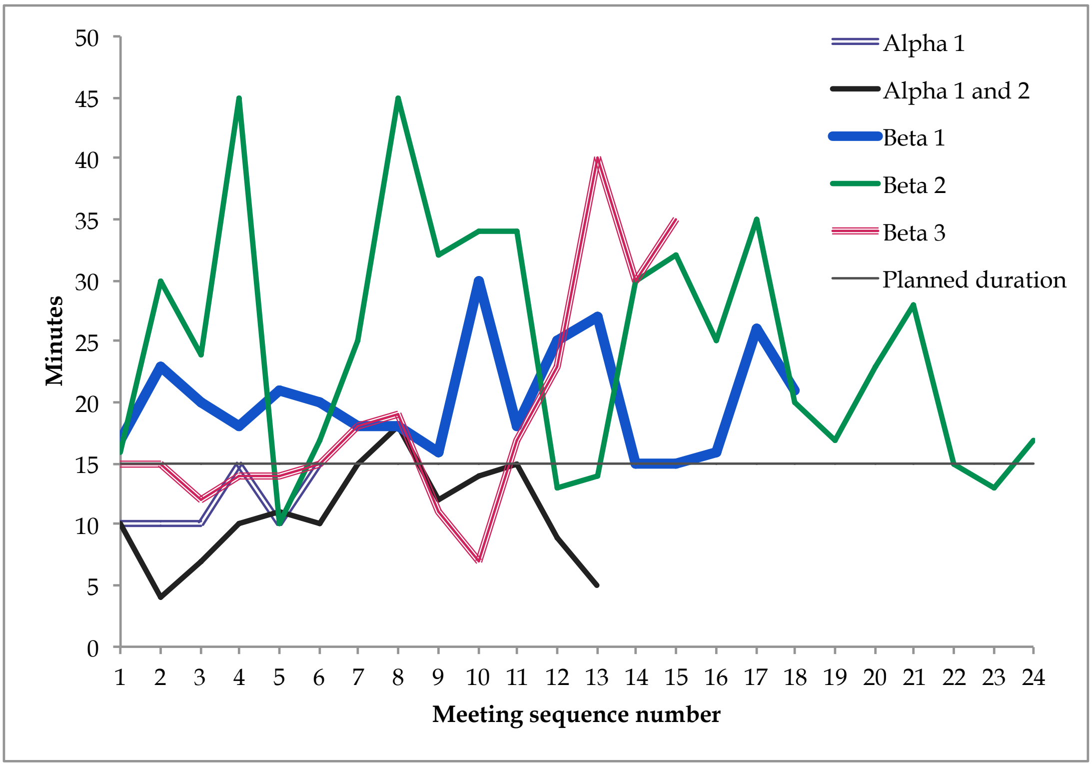


Figure 6: Duration of DSMs in Alpha and Beta

Table 13 shows the average duration of the observed meetings at the team level, company level and total. All teams planned 15 minutes for the meeting, and the observed DSMs actually lasted, on average at the company level, exactly 15 minutes. However, there were great variations. In Beta, they struggled with adhering to the 15-minute time limit. Their average meeting length was more than 50% longer than that in Alpha (21.3 versus 10.2 minutes). At the team level, the average duration varied between 7.0 and 24.8 minutes. The duration of the meeting was noted as too long by thirteen interviewees, five of whom belonged to Beta 2, which had the longest average duration.

Figure 6 shows the meeting duration over time in Alpha and Beta (Gamma is not shown because of few observations in this company). There was a sharp drop in Beta 2 at Meetings 5 and 6. These meetings were shorter because the regular Scrum Master (project manager) was absent, which is illustrated by an excerpt from a transcribed meeting:

> Developer X: “On Monday the Scrum Master is back and will lead the meetings again.”
> Developer Y: “Oh, no!” [Everybody laughs] “Then we will be back to half an hour.”

Meetings 12 and 13 in Beta 2 were kept brief because the team had other meetings scheduled (a demo meeting and a company meeting). The graph shows a sharp increase in duration after Meeting 10 in Beta 3. The extended length of Meeting 13 was due to an intense discussion in which disagreements occurred between the testers and the developers. They argued about adding test cases late in the sprint, and one of the testers started crying. Meetings 14 and 15 required extra time for recovering activities after Meeting 13.

The teams tried different practices to keep meetings short. One practice was to have the meeting facilitator remind the participants to keep the discussion on target. Another practice was to use a countdown timer. A developer explained, "Our updates have stretched a little because sometimes we go straight into technical discussions of our roadblocks instead of waiting for the daily update to finish first. Before we start the daily update, the Scrum Master sets the clock timer to 15 minutes. If we have not finished when the timer goes off, the team can walk away to continue their work without the need to wait for the rest to finish updating."

P6: Too high frequency and too long duration of DSMs negatively affect the attitude towards DSMs.

4.7 Attitude towards DSMs

By reading all the interview transcripts, the first and second authors scored the interviewees' attitude towards DSMs on a scale from 1 to 5.

Figure 7 shows the first and second authors' attitude scores of the interview transcripts, aggregated to team level. The score for each team varied from 2.8 to 4.0. The teams were slightly more positive than negative on average (3.5).

In Alpha 1 and 2, we had the opportunity to investigate the attitude towards certain aspects of DSMs by giving team members the questionnaire described in Table 5. Figure 9 shows the respondents' scores on specific statements. The average overall scores in Alpha 1 and 2 were 3.4 and 3.0, respectively. Although not directly comparable, the overall attitude in these teams accorded with the scores from the interviews, which were 3.1 and 3.0, respectively. These results indicate consistency between the first and second authors' interpretation of the interviews and what the participants themselves indicated in the questionnaire.

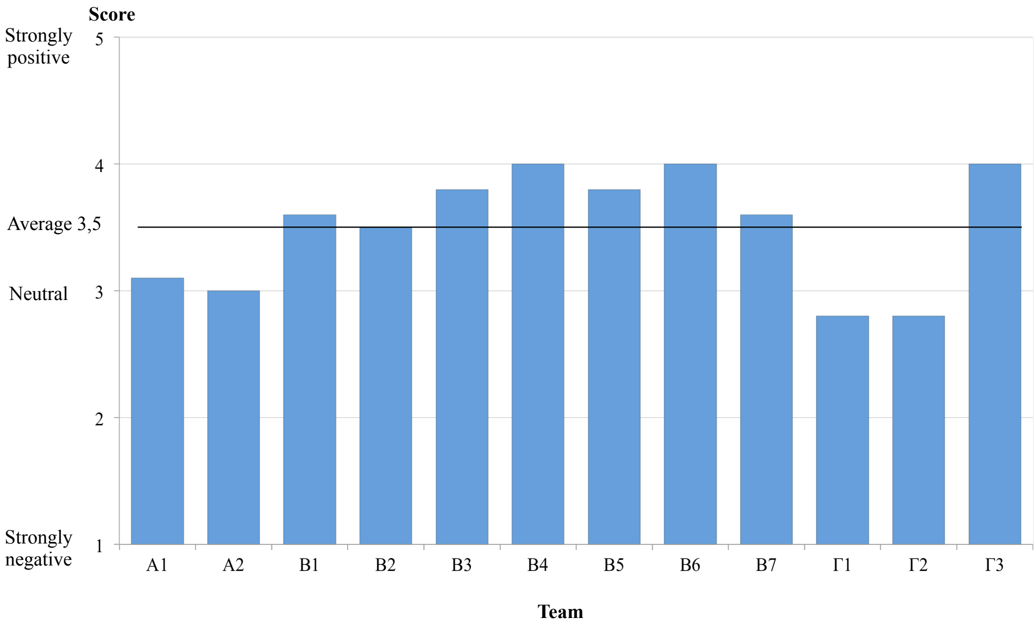


Figure 7: Average meeting attitude for each team

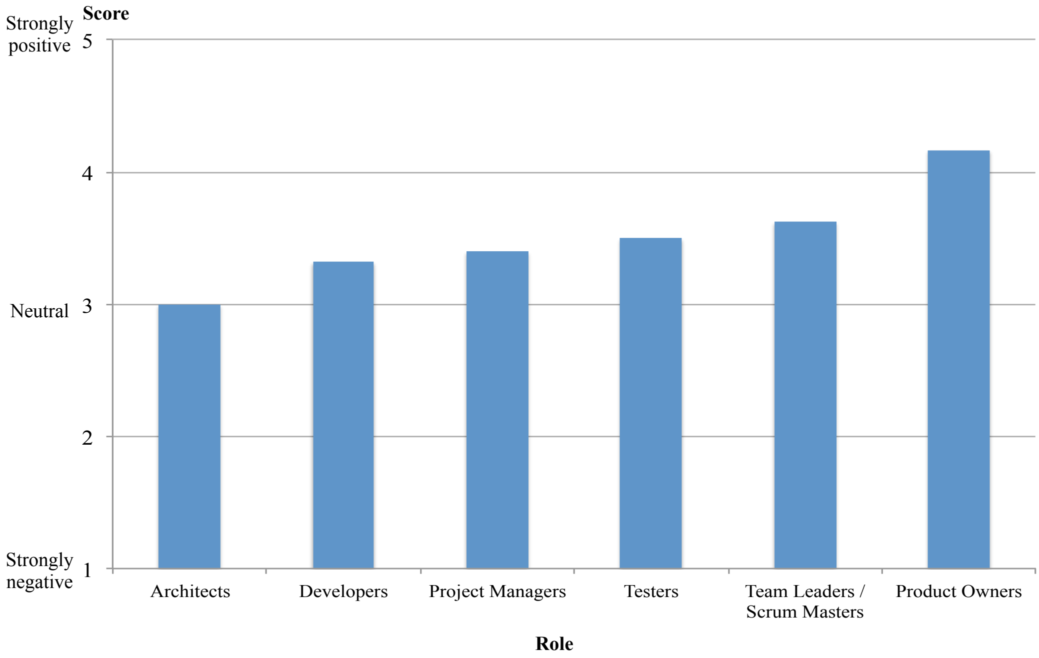


Figure 8: Average meeting attitude for each role

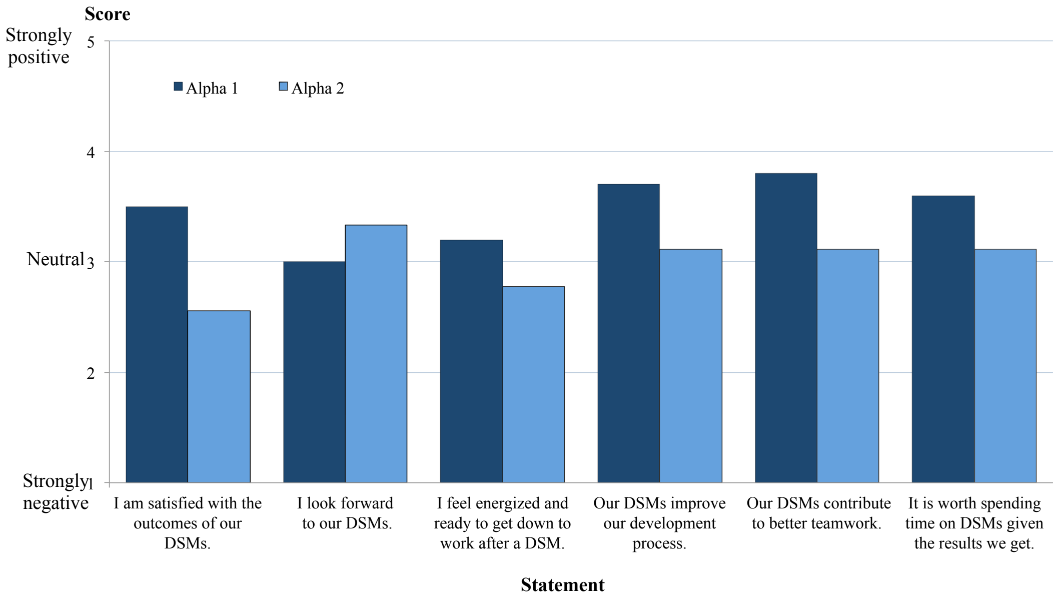


Figure 9: Meeting attitude of teams Alpha 1 and Alpha 2 from questionnaire

5 Discussion

This section describes the practical and research implications of the results, the limitations of the study and future work.

5.1 Normative definition based on the Empirical Findings

Throughout this article the practice investigated is called "DSM" ("Daily Stand-up Meeting") because most literature includes the word "daily" when referring to the practice. However, the findings indicate that the meeting should not necessarily be held daily, but to distinguish it from an ad hoc meeting, the meeting should be conducted regularly. Consequently, it may be sensible to replace "daily" in the name of the practice with "regular, that is, "regular stand-up meeting".

Based on the empirical investigation we define a regular stand-up meeting as follows:

> *A regular stand-up meeting is a communicative event involving two or more people; it is regularly scheduled with a pre-arranged time and place; it is organized and managed by the team; its primary purpose is to make team members obtain a shared understanding of the current activities of other team members; its duration is not more than 15 minutes; and the participants stand.*

5.2 Empirical-based guidelines for DSM

We compare the results of the current study with the recommendations for conducting DSMs according to "Daily Scrum" (Sutherland and Schwaber, 2013a) and "Daily stand up meeting" (Wells, 2013). Our empirically based guidelines are summarized in Table 14.

Table 14: DSM Guidelines

Characteristic	Regular Stand-Up Meeting (based on empirical data)
Purpose	Obtain a shared understanding of the current activities of other team members
Potential benefits	Improve communication, knowledge sharing and team orientation. Identify, avoid and solve problems
Potential pitfalls	Reporting status that is not relevant to all team members. Wasting time because team members are disengaged. Interruption of workflow
DSM questions	The team members explain the following: 1. What will I do today to help our team accomplish the iteration goal? 2. What problems do I know of that may prevent progress?
Format	Meeting participants stand up in a semicircle in front of a physical or electronic board
Turn-taking	Round-robin
Frequency	Must be held regularly, but the frequency may differ depending on iteration phase, need and level of informal communication
Time of day	Find the least disruptive time for the team
Duration	As brief as possible (think "huddle" instead of "meeting") but maximum 15 minutes

Purpose

The main purpose of DSMs should be to enable team members to obtain a shared understanding of the current activities of other team members, which in turn supports their own activities. This is consistent with the purpose of DSMs as stated in the description of the XP guidelines: "Communication among the entire team" (Wells, 2013). Synchronization of activities and planning, which appears to be the purpose of DSMs according to the Scrum guidelines (Sutherland and Schwaber, 2013a), was not the main purpose of DSMs according to our study. However, some synchronization and planning was indirectly performed through discussions of what to do and how to deal with obstacles.

Potential benefits

Consistent with the Daily Scrum and XP guidelines, our findings support that DSMs improve communication, not only in the sense of communication happening in the meeting but also in post meetings; DSMs make it easy to conduct follow-up discussions afterwards in post meetings. Particularly in distributed projects, organizing meetings for technical discussions across sites may be difficult without a preceding DSM.

A major part of the communication resulted in sharing information and knowledge and identifying problems that were solved or simply avoided. This finding is consistent with those of Rising and Janoff (2000), who reported that solving and clearing obstacles were the best parts of DSMs. Pikkarainen et al. (2008) also found that resolving problems in the meeting was one of the most valuable outcomes, even though agile guidelines suggest the meetings not be used for discussing solutions. To increase the probability that problems are shared, organizations should provide an environment in which participants are comfortable sharing their obstacles and receive help to overcome them, either in the meeting or immediately afterwards. Identifying and avoiding problems may entail major cost savings because finding problems earlier is more cost effective than finding them later (Boehm, 1984).

The Scrum Guide (Sutherland and Schwaber, 2013a) states that DSMs highlight and promote quick decision-making. We did observe quick decision-making, but one should not ignore the negative effects of the short time allowed for making decisions, including lack of time to consider alternatives and document the decisions made. A further discussion of this topic can be found in (Drury et al., 2012; Stray et al., 2012a).

Furthermore, consistent with the guidelines of Scrum and XP, we found that DSMs reduced the need for other meetings. Another benefit of DSMs was that the increased team orientation that they generated helped erase boundaries between roles and sites.

Potential pitfalls

As stated by Sutherland and Schwaber (2013b), the DSM is too often seen as a status event. This statement is also confirmed in our study: Reporting progress was the greatest pitfall of DSMs because it often resulted in disengaged or uncomfortable participants. Other pitfalls are waste of time and interruption of workflow. The pitfalls are further described in the next subsections.

DSM questions

Our study revealed that the second and third DSM questions (Table 1) of the Scrum and XP guidelines are most useful. Answers to the first question tend to become dominated by status reporting. In addition to the negative aspects of reporting that have already been described, the first question consumes a large proportion of the total time in DSMs at the cost of the time available for the more important second and third questions. In Stray et al. (2012a), we reported that the proportions of the statements concerning the DSM questions were distributed as follows: first question 53%, second question 25% and third question 22%. Stray et al. (2012b) observed that a substantial amount of time for the first question was spent on self-justification; that is, participants used the time for detailed explanations of what they had done and why they had not achieved as much as expected, if that was the case. In the present study, we observed that several teams had stopped answering the three questions because it took too much time. Additionally, in a study of Scrum-of-Scrum meetings, the teams stopped answering all three questions because they found it too time-consuming; they wanted to focus on obstacles (Paasivaara et al., 2012).

To emphasize that the DSM is a planning event and not a status event, the three DSM questions were revised in the 2013 edition of the Scrum Guide (Sutherland and Schwaber, 2013b). We suggest removing the first question completely, which would allow more time for answering the other two questions. Other means are more efficient than DSMs for providing information about what has been achieved; for example, most status inquiries are no longer necessary in visual workplaces because the project status is constantly displayed, updated and accessible to all (Bell and Orzen, 2010). If need be, additional oral information about what has been accomplished could be provided on the fly when the second and third questions are being answered.

Format

The teams that had all participants standing had considerably shorter meetings than those that had some people, especially the Scrum Master, sitting. This is consistent with the results of a previous experiment that found that the sit-down meetings lasted 34% longer than the stand-up meetings but both types of meetings produced decisions with the same quality (Bluedorn et al., 1999). Consequently, we support the recommendation stated on the XP homepage that people should stand for the meeting. Our findings also suggest that it is valuable for the team to stand in front of a board, or other means of visualizing tasks, for the meeting.

Turn-taking

When the facilitator decided the order of speakers, the meeting tended to shift towards reporting status. Boden (1994, p. 89) also found that when the facilitator allocated the turns, the meeting often became conversations between the facilitator and the respondents. To reduce this tendency, we recommend taking turns by using a round-robin approach and rotating the role of the meeting facilitator. The meeting facilitator does not have to be the Scrum Master, particularly if the Scrum Master is also the manager.

Frequency
The stand-up meeting does not necessarily have to be held daily. Teams may decide to have it three or four days a week, which many of the teams in this study practiced and appreciated. Additionally, there might be phases in the iteration during which it would be beneficial to conduct the stand-up meeting more frequently than during other phases. Thus, it may be sensible to replace “daily” in the name of the practice with “regular” and call it a “regular stand-up meeting,” as shown in Table 14.

Time of day
If team members all arrive at the same time in the morning, starting the workday with a DSM is a good idea. However, in our study, people arrived at different times, and few of the team members began working on development tasks before the DSM was held. In such cases, we recommend conducting the meeting just before lunch to avoid the DSM becoming the start of the actual workday. There would then be no additional interruption, and discussions could be continued during lunch. The proximity to lunch may also serve as a motivation to end the meeting on time. DSMs have an indirect cost called resumption lag, which is the time it takes for a person to collect their thoughts and return to a task after an interruption (Parnin and Rugaber, 2011). Our general recommendation is that teams should strive to find the least disruptive time for the meeting.

Duration
Even though the meetings in our study lasted 15 minutes on average, perceiving the meeting as being too long was the most frequently mentioned negative aspect, both in the interviews and in the questionnaire. We argue that DSMs should be as short as possible and not last more than 15 minutes. Furthermore, ending the meeting on time is important because participants tend to stop paying attention when the time limit is reached. Viewing a DSM as a “huddle,” a brief gathering of team members, instead of a meeting might positively affect the duration and reinforce the informality of the meeting.

5.3 Limitations

Researcher bias and dependency on case and context make it difficult to generalize the results of this study. It is reasonable to suspect that DSMs and their effects would vary across different types of organizations and employees. For example, the teams that we studied had used DSMs for more than two years. Our results may not necessarily be applicable to teams that just have started using DSMs.

Regarding data collection, the presence of researchers in a DSM may be intrusive and alter the behavior of the meeting attendees. Nevertheless, we believe this effect was small because most of the teams were observed over a long period of time, which meant that the participants became used to being observed. Tape recorders used in meetings may also bias the social situation, although we observed no difference in the behavior of the meeting attendees when the meeting was audio-recorded.

Furthermore, the attitudes towards DSM may affect the DSM process itself, either positively or negatively. For example, a person with a negative attitude towards DSM might be reluctant to contribute in the meeting.

To address potential limitations, we used the seven principles for conducting an interpretive field study that were proposed by Klein and Myers (1999), as presented in Table 15. In addition, we undertook data triangulation to reduce researcher bias. We conducted interviews, observed meetings and collected questionnaire data. Such triangulation may yield more reliable data than the use of only one data source because what people report is not always consistent with reality; for example, people tend to underestimate the amount of time they spend in face-to-face meetings when they are asked how much time they spend in various activities (Panko and Kinney, 1995). Interviews yield subjective data; however, the more interviewees stating the same opinion, the less likely bias is associated with that opinion (Diefenbach, 2009). The high number of interviews in our study enabled us to cross-check and compare the data, which produced results that are more convincing than would be possible with fewer interviews.

Table 15: Application of Klein and Myers principles for interpretive field research

The seven principles (Klein and Myers, 1999)	Practiced in this study
1. The Hermeneutic Circle	We observed and interviewed teams over a period of 26 months, which allowed us to study DSMs from different viewpoints and in different phases of the projects. We conducted a large number of interviews and observations, moving back and forth between the three companies. Both the data collection and analysis involved multiple researchers, who analyzed the data systematically in an iterative process, adding more interviews and different data sources to the analysis through theoretical sampling.
2. Contextualization	We describe in detail the characteristics of the observed DSMs and the interviewees. We also describe the size and industry of the companies.
3. Interaction between researchers and subjects	We had lunch, dinners and informal conversations with the team members. This socializing made it easier for us to be trusted and gave us valuable insight in the teams. Our findings were presented and discussed in all three companies and led to feedback.
4. Abstraction and generalization	Investigating DSMs in different national and organizational cultures helped us understand the general perceptions and nature of the practice. By using grounded theory, we abstracted our results into a theory.
5. Dialogical reasoning	Our analysis generated DSM constructs that we combined to form a theory. We frequently referred to the data to ensure that codes were representative and checked relationships among codes and themes. In the third phase of the research process, we cross-referenced our results with literature on teamwork and meetings.
6. Multiple interpretations	We collected data from twelve teams in four countries. We interviewed 60 persons with various roles and were able to document multiple viewpoints and their reasons. The large amount of data collected reduced the chance of biases and systematic distortions.
7. Suspicion	We had a critical approach when analyzing the data; both positive and negative aspects of DSMs were studied. The researchers were external to the organizations, having no interest or agenda beyond creating an unbiased view of DSMs.

5.4 Future work

Our premise is that practitioners will be better able to conduct DSMs in a way that is beneficial to the organization if they increase their understanding of how the practice affects the overall productivity. Obtaining such an understanding is difficult, however. Therefore, future work should investigate DSMs more specifically from a cost-benefit perspective. Do the benefits of DSMs justify the time spent in them and the cost of interruption? This complex problem may be divided into several research questions; for example, “How do DSMs affect team performance?” To answer such a question, researchers may investigate how DSMs affect teamwork quality, defined by Hoegl and Gemuenden (2001) in terms of the subconstructs *communication*, *coordination*, *balance of member contributions*, *mutual support*, *effort* and *cohesion.* Addressing the greater challenge of cost–benefit of DSMs in a solid way requires empirical data from a larger number of software projects. A larger number of projects may also enable analyses of the effect of various context factors on the DSM process. Context factors may include, for example, aspects of company and team culture. Such factors should also be included in a refined version of the DSM theory.

6 Conclusion

Most views and claims about DSMs reported in the literature are based on anecdotal evidence. In contrast, we conducted a grounded theory study of twelve agile teams in three companies. Considering the popularity of DSMs, it is surprising that we found DSM participants to be almost neutral about DSMs on average, slightly more satisfied than dissatisfied.

The DSM may seem like a simple practice to successfully implement. Unfortunately, this is not the case. We identified factors that affected the meeting attitude positively and negatively. The two most prominent positive factors were that the team members obtained an overview of what others were doing and that the DSMs provided an opportunity for discussing and solving problems.

The two most prominent negative factors were as follows. First, reporting status progress in the meeting caused disengagement because the information was not relevant for the whole team, and many of the team members were uncomfortable about reporting such progress. Second, the DSMs were often considered to occupy too much time relative to the gains from the meetings.

The findings show that DSMs may not necessarily have to be held daily, and focus in the meetings should be on discussing and solving problems, and planning for the future rather than reporting what has been done.

A grounded theory of the DSM was proposed, consisting of seven constructs and six propositions. In order to overcome the negative factors identified, a modified set of guidelines on how to conduct the DSM was proposed to support software companies in realizing its potential benefits. Companies should evaluate these guidelines in their own context and continuously evaluate and improve the way they conduct DSMs to make the practice as valuable as possible.

Relatively few grounded theory studies have been conducted in the field of software engineering. An additional contribution of this paper is that it thoroughly describes how such a study may be undertaken in the context of agile methods.

Acknowledgements

We are grateful to Yngve Lindsjørn, Nils Brede Moe and Øystein Ingebrigtsen for assisting with data collection and to the many participants who shared their experiences with us and generously welcomed us for observations. This work was supported by the TeamIT project funded by the Research Council of Norway through grant no. 193236/I40.

Appendix A Interview guide

Introduction

- Present ourselves.
- Say thank you for participating.
- Assure confidentiality.
- Ask permission to tape record.

"Warm-up"

- How long have you been working for this company?
- How long have you been on this project?
- Is it okay to ask: "How old are you?"
- What is your role in the team?
- What are you working on now?
- Whom do you see as your team members?
- Do you collaborate with other teams?

Teamwork and meetings

- How does the team make decisions?
- Do team members show interest in other individuals' tasks?
- How are tasks allocated?
- How easy is it to complete someone else's task?
- Do you get feedback on your work?
  - How?
  - When?
- Tell me about your daily stand-up meetings
  - What is working?
  - What is not working?
  - What have you done to improve them?
  - How many attendees?
- Tell me about your retrospective meetings
- Tell me about the planning meetings
- How do you solve problems?
- What do you think of the information flow in the project?
- How do you perceive the teamwork in the project?
- Do you have an overview of what others are doing?
- What are some challenges of working with distributed teams?
- What can you think of that could improve the effectiveness of the teamwork or the project in general?
- How do you think agile methodologies and practices are working in the project?

Closing:

- Do you have any questions for me?
- Is there anything else you would like to discuss that was not covered by the questions asked?

Appendix B Observation protocol

Space

- What is the layout of the physical room?
- How are the actors positioned?

Participants

- What are the names and relevant details of the people involved?
- Is someone acting as a leader or facilitator?

Activities

- What are the various activities and discussions?

Objects

- Which physical elements are used?

Acts

- Are there any specific individual actions?
- What are the ways in which all actors interact and behave toward each other?

Events

- Are there any particular occasions or anything unexpected?

Time

- When does the meeting start?
- What is the sequence of events?
- When does the meeting end?

Goals

- What are the actors attempting to accomplish?

Feelings

- What are the emotions in the particular contexts?
- How is the atmosphere?

Closing

- How is the meeting ended?
- Is there a post meeting?

References

Allen, J.A., Sands, S.J., Mueller, S.L., Frear, K.A., Mudd, M., Rogelberg, S.G., 2012. Employees' feelings about more meetings: An overt analysis and recommendations for improving meetings. Management Research Review 35, 405–418.

Amin, Z., Guajardo, J., Wisniewski, W., Bordage, G., Tekian, A., Niederman, L., 2000. Morning report: Focus and methods over the past three decades. Academic Medicine 75, 1–4.

Anderson, A.H., McEwan, R., Bal, J., Carletta, J., 2007. Virtual team meetings: An analysis of communication and context. Computers in Human Behavior 23, 2558–2580.

Anderson, D.J., 2003. Agile Management for Software Engineering: Applying the Theory of Constraints for Business Results. Prentice Hall, Upper Saddle River, NJ.

Bedwell, W.L., Wildman, J.L., DiazGranados, D., Salazar, M., Kramer, W.S., Salas, E., 2012. Human Resource Management Review. Human Resource Management Review 22, 128–145. doi:10.1016/j.hrmr.2011.11.007

Bell, S.C., Orzen, M.A., 2010. Lean IT: Enabling and Sustaining Your Lean Transformation. Productivity Press, New York, NY.

Bluedorn, A.C., Turban, D.B., Love, M.S., 1999. The effects of stand-up and sit-down meeting formats on meeting outcomes. Journal of Applied Psychology 84, 277–285.

Boden, D., 1994. The Business of Talk: Organizations in Action. Polity Press, Cambridge, MA.

Boehm, B.W., 1984. Verifying and validating software requirements and design specifications. IEEE Software 1, 75–88. doi:10.1109/MS.1984.233702

Bureau of Labor Statistics, U.S. Department of Labor, 2014. Occupational Outlook Handbook, 2014-15 Edition, Software Developers. http://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm (accessed July 2014).

Cannon-Bowers, J.A., Salas, E., Converse, S., 1993. Shared Mental Models in Expert Team Decision Making, in: Castellan, N.J. (Ed.). Current Issues in Individual and Group Decision Making, pp. 221–246.

Cohen, S., Bailey, D.E., 1997. What Makes Teams Work: Group Effectiveness Research from the Shop Floor to the Executive Suite. Journal of Management 23, 239.

Coplien, J.O., 1994. Borland Software Craftsmanship: A New Look at Process, Quality and Productivity, in: Proceedings of the 5th Annual Borland International Conference, pp. 1–11

DeSanctis, G., Poole, M.S., 1994. Capturing the Complexity in Advanced Technology Use: Adaptive Structuration Theory. Organization Science 5, 121–147.

Dickinson, T.L., McIntyre, R.M., 1997. A Conceptual Framework of Teamwork Measurement, in: Brannick, M.T., Salas, E., Prince, C. (Eds.), Team Performance Assessment and Measurement: Theory, Methods, and Applications. Psychology Press, NJ, pp. 19–43.

Diefenbach, T., 2009. Are case studies more than sophisticated storytelling? Methodological problems of qualitative empirical research mainly based on semi-structured interviews. Quality and Quantity 43, 875–894. doi:10.1007/s11135-008-9164-0

Dingsøyr, T., Nerur, S., Balijepally, V., Moe, N.B., 2012. A decade of agile methodologies: Towards explaining agile software development. The Journal of Systems & Software 85, 1213–1221. doi:10.1016/j.jss.2012.02.033

doi:10.1007/978-1-84800-044-5_12

Dorairaj, S., Noble, J., Malik, P., 2012. Understanding Team Dynamics in Distributed Agile Software Development, in: Wohlin, C. (Ed.), Proceedings of the 13th International

Conference on Agile Processes in Software Engineering and Extreme Programming. Lecture Notes in Business Information Processing, Springer, Berlin Heidelberg, pp. 47–61. doi:10.1007/978-3-642-30350-0_4

Dourish, P., Bellotti, V., 1992. Awareness and coordination in shared workspaces, in: Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work (CSCW '92), ACM, New York, NY, pp. 107–114. doi:10.1145/143457.143468

Doyle, M., Straus, D., 1993. How to Make Meetings Work: The New Interaction Method, Reprint. ed. Berkeley Trade.

Drury, M., Conboy, K., Power, K., 2012. Obstacles to decision making in Agile software development teams. The Journal of Systems & Software 85, 1239–1254. doi:10.1016/j.jss.2012.01.058

Dybå, T., Sjøberg, D., Cruzes, D.S., 2012. What Works for Whom, Where, When, and Why? On the Role of Context in Empirical Software Engineering, in:. Presented at the: Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement (ESEM '12), ACM, New York, NY, pp. 19–28. doi:10.1145/2372251.2372256

Elsayed-Elkhouly, S., Lazarus, H., Forsythe, V., 1997. Why is a third of your time wasted in meetings? Journal of Management Development 16, 672–676. doi:10.1108/02621719710190185

Endsley, M.R., 1995. Toward a Theory of Situation Awareness in Dynamic Systems. Human Factors: The Journal of the Human Factors and Ergonomics Society 37, 32–64.

Glaser, B.G., 1978. Theoretical Sensitivity: Advances in the Methodology of Grounded Theory. Sociology Press, Mill Valley, CA.

Glaser, B.G., 1992. Basics of Grounded Theory Analysis: Emergence vs Forcing. Sociology Press, Mill Valley, CA.

Glaser, B.G., 1998. Doing Grounded Theory: Issues and Discussions. Sociology Press, Mill Valley, CA.

Glaser, B.G., 2001. The Grounded Theory Perspective: Conceptualization Contrasted with Description. Sociology Press, Mill Valley, CA.

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

Glaser, B.G., 2011. Getting Out of the Data: Grounded Theory Conceptualization. Sociology Press, Mill Valley, CA.

Glaser, B.G., Strauss, A.L., 1967. The Discovery of Grounded Theory: Strategies for Qualitative Research. Aldine, New York, NY.

Goulding, C., 1998. Grounded theory: The missing methodology on the interpretivist agenda. Qualitative Market Research: An International Journal 1, 50–57.

Hannay, J.E., Sjøberg, D.I.K., Dybå, T., 2007. A systematic review of theory use in software engineering experiments. IEEE Transactions on Software Engineering 33, 87–107. doi:10.1109/TSE.2007.12

Hare, A.P., 1952. A study of interaction and consensus in different sized groups. American Sociological Review 17, 261–267.

Herbsleb, J., Mockus, A., 2003. Formulation and preliminary test of an empirical theory of coordination in software engineering, in: Proceedings of the 9th European Software Engineering Conference held jointly with 11th ACM SIGSOFT International Symposium on Foundations of Software Engineering (ESEC/FSE-11), ACM, New York, NY, pp. 138–147. doi:10.1145/940071.940091

Hernandez, C.A., 2009. Theoretical coding in grounded theory methodology. Grounded Theory Review: An International Journal, Sociology Press 8, 51–59.

Hobfoll, S.E., 2001. The Influence of Culture, Community, and the Nested‐Self in the Stress Process: Advancing Conservation of Resources Theory. Applied Psychology: An International Review 50, 337–421.

Hoegl, M., Gemuenden, H., 2001. Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. Organization Science 12, 435–449. doi:10.1287/orsc.12.4.435.10635

Jett, Q.R., George, J.M., 2003. Work interrupted: A closer look at the role of interruptions in organizational life. Academy of Management Review 28, 494–507. doi:10.5465/AMR.2003.10196791

Kauffeld, S., Lehmann-Willenbrock, N., 2012. Meetings matter: Effects of team meetings on team and organizational success. Small Group Research 43, 130–158. doi:10.1177/1046496411429599

Klein, H., Myers, M., 1999. A set of principles for conducting and evaluating interpretive field studies in information systems. MIS Quarterly 23, 67–93. doi:10.2307/249410

Kniberg, H., Skarin, M., 2010. Kanban and Scrum making the most of both. InfoQ. http://www.infoq.com/minibooks/kanban-scrum-minibook (accessed July 2014).

Landis, J.R., Koch, G.G., 1977. The measurement of observer agreement for categorical data. Biometrics 33, 159–174.

Levesque, L.L., Wilson, J.M., Wholey, D.R., 2001. Cognitive divergence and shared mental models in software development project teams. Journal of Organizational Behavior 22, 135–144. doi:10.1002/job.87

Luong, A., Rogelberg, S.G., 2005. Meetings and more meetings: The relationship between meeting load and the daily well-being of employees. Group Dynamics: Theory, Research, and Practice 9, 58–67. doi:10.1037/1089-2699.9.1.58

McHugh, O., Conboy, K., Lang, M., 2012. Agile practices: The impact on trust in software project teams. IEEE Software 29, 71–76. doi:10.1109/MS.2011.118

Merisalo-Rantanen, H., Tuunanen, T., Rossi, M., 2005. Is extreme programming just old wine in new bottles: A comparison of two cases. Journal of Database Management (JDM) 16, 41–61. doi:10.4018/jdm.2005100103

Moe, N.B., Dingsøyr, T., Dybå, T., 2010. A teamwork model for understanding an agile team: A case study of a Scrum project. Information and Software Technology 52, 480–491. doi:10.1016/j.infsof.2009.11.004

Murphy, B., Bird, C., Zimmermann, T., Williams, L., 2013. Have agile techniques been the silver bullet for software development at Microsoft?, in: Proceedings of the 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. IEEE, Baltimore, MD. doi:10.1109/ESEM.2013.21

O'Neill, T.A., Allen, N.J., 2012. Team meeting attitudes: Conceptualization and investigation of a new construct. Small Group Research 43, 186–210. doi:10.1177/1046496411426485

Paasivaara, M., Durasiewicz, S., Lassenius, C., 2008. Using Scrum in a globally distributed project: A case study. Software Process: Improvement and Practice 13, 527–544. doi:10.1002/spip.402

Paasivaara, M., Lassenius, C., Heikkilä, V.T., 2012. Inter-team coordination in large-scale globally distributed scrum: Do Scrum-of-Scrums really work? in: Proceedings of the 2012 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, pp. 235–238. doi:10.1145/2372251.2372294

Paez, O., Salem, S., Solomon, J., Genaidy, A., 2005. Moving from lean manufacturing to lean construction: Toward a common sociotechnological framework. Human Factors and Ergonomics in Manufacturing 15, 233–245. doi:10.1002/hfm.20023

Panko, R., Kinney, S., 1995. Meeting profiles: Size, duration, and location. Proceedings of the 28th Hawaii International Conference on System Sciences, 1002–1011.

Parnin, C., Rugaber, S., 2011. Resumption strategies for interrupted programming tasks. Software Quality Journal 19, 5–34. doi:10.1007/s11219-010-9104-9

Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., Still, J., 2008. The impact of agile practices on communication in software development. Empirical Software Engineering 13, 303–337. doi:10.1007/s10664-008-9065-9

Reinig, B.A., 2003. Toward an Understanding of Satisfaction with the Process and Outcomes of Teamwork. Journal of Management Information Systems 19, 65–83.

Rindfleisch, A., Moorman, C., 2001. The Acquisition and Utilization of Information in New Product Alliances: A Strength-of-Ties Perspective. Journal of Marketing 65, 1–18.

Rising, L., 2002. Agile meetings. STQE Magazine - Software Testing and Quality Engineering, pp. 42–46.

Rising, L., Janoff, N., 2000. The Scrum software development process for small teams. IEEE Software 17, 26–32. doi:10.1109/52.854065

Robson, C., 2002. Real World Research: A Resource for Social Scientists and Practitioner-Researchers, 2nd ed. Blackwell Publishing, Malden, MA.

Rogelberg, S.G., Allen, J.A., Shanock, L., Scott, C., Shuffler, M., 2010. Employee satisfaction with meetings: A contemporary facet of job satisfaction. Human Resource Management 49, 149–172. doi:10.1002/hrm.20339

Rogelberg, S.G., Leach, D.J., Warr, P.B., Burnfield, J.L., 2006. "Not another meeting!" Are meeting time demands related to employee well-being? Journal of Applied Psychology 91, 86–96. doi:10.1037/0021-9010.91.1.83

Romano, N.C., Nunamaker, J.F., Jr, 2001. Meeting analysis: Findings from research and practice, in: Proceedings of the 34th Hawaii International Conference on System Sciences, Maui, Hawaii, pp. 1–13. doi:10.1109/HICSS.2001.926253

Salas, E., Prince, C., Baker, D.P., Shrestha, L., 1995. Situation awareness in team performance: Implications for measurement and training. Human Factors: The Journal of the Human Factors and Ergonomics Society 37, 123–136. doi:10.1518/001872095779049525

Schwaber, K., 2003. Scrum and The Perfect Storm http://www.controlchaos.com/storage/scrum-articles/Scrum%20and%20The%20Perfect%20Storm.pdf (accessed July 2014).

Schwaber, K., Beedle, M., 2002. Agile Software Development with Scrum. Prentice Hall, Upper Saddle River, NJ.

Sharp, H., Robinson, H., 2010. Three "C"s of agile practice: Collaboration, co-ordination and communication, in: Dingsøyr, T., Dybå, T., Moe, N.B. (Eds.), Agile Software Development: Current Research and Future Directions. Springer Berlin Heidelberg, pp. 61–85. doi:10.1007/978-3-642-12575-1_4

Sjøberg, D., Dybå, T., Anda, B., Hannay, J., 2008. Building Theories in Software Engineering, in: Shull, F., Singer, J., Sjøberg, D. (Eds.). Guide to Advanced Empirical Software Engineering, Springer London, pp. 312–336.

Sjøberg, D., Johnsen, A., Solberg, J., 2012. Quantifying the Effect of Using Kanban versus Scrum: A Case Study. IEEE Software 29, 47–53. doi:10.1109/MS.2012.110

Sjøberg, D.I.K., Dybå, T., Jorgensen, M., 2007. The future of empirical methods in software engineering research, in: Future of Software Engineering, FOSE'07, IEEE, Minneapolis, MN, pp. 358–378. doi:10.1109/FOSE.2007.30

Solingen, R., Berghout, E., Latum, F., 1998. Interrupts: Just a minute never is. IEEE Software 15, 97–103. doi:10.1109/52.714843

Sonnentag, S., Volmer, J., 2009. Individual-level predictors of task-related teamwork processes: The role of expertise and self-efficacy in team meetings. Group Organization Management 34, 37–66. doi:10.1177/1059601108329377

Spradley, J.P., 1980. Participant Observation, 1st ed. Holt, Rinehart and Winston, Austin, TX.

Strauss, A., Corbin, J.M., 1990. Basics of Qualitative Research: Grounded Theory Procedures and Techniques, 2nd ed. Sage Publications, Thousand Oaks, CA.

Stray, V.G., Lindsjørn, Y., Sjøberg, D.I.K, 2013. Obstacles to efficient daily meetings in agile development projects: A case study, in: Proceedings of the 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. IEEE, Baltimore, MD, pp. 105–102. doi:10.1109/ESEM.2013.30

Stray, V.G., Moe, N.B., Aurum, A., 2012a. Investigating daily team meetings in agile software projects, in: Proceedings of the 38th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE, pp. 274–281. doi:10.1109/SEAA.2012.16

Stray, V.G., Moe, N.B., Dingsøyr, T., 2011. Challenges to teamwork: A multiple case study of two agile teams, in: Sillitti, A., Hazzan, O., Bache, E., Albaladejo, X. (Eds.), Agile Processes in Software Engineering and Extreme Programming. Lecture Notes in Business Information Processing, Springer Berlin Heidelberg, Madrid, Spain, pp. 146–161. doi:10.1007/978-3-642-20677-1_11

Stray, V.G., Moe, N.B., Dybå, T., 2012b. Escalation of commitment: A longitudinal case study of daily meetings, in: Wohlin, C. (Ed.), Proceedings of the 13th International Conference on Agile Processes in Software Engineering and Extreme Programming. Lecture Notes in Business Information Processing, Springer Berlin Heidelberg, Malmö, Sweden, pp. 153–167. doi:10.1007/978-3-642-30350-0_11

Sutherland, J., Schwaber, K., 2011. The Scrum Guide 2011 pp. 1–16. https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum_Guide.pdf (accessed July 2014).

Sutherland, J., Schwaber, K., 2013a. The Scrum Guide 2013 pp. 1–16. https://www.scrum.org/Scrum-Guide (accessed July 2014).

Sutherland, J., Schwaber, K., 2013b. The Scrum Guide 2013 (Changes) pp. 1–2 http://www.scrumguides.org/s/Scrum-Guide-2013-Changes.pdf (accessed July 2014).

VersionOne, 2007. 2nd Annual State of Agile Development Survey http://www.versionone.com/pdf/2007-state-of-agile-survey.pdf (accessed July 2014).

VersionOne, 2008. 3rd Annual State of Agile Development Survey http://www.versionone.com/pdf/3rdannualstateofagile_fulldatareport.pdf (accessed July 2014).

VersionOne, 2009. 4th Annual State of Agile Development Survey http://www.versionone.com/pdf/2009_state_of_agile_development_survey_results.pdf (accessed July 2014).

VersionOne, 2010. 5th Annual State of Agile Development Survey http://www.versionone.com/pdf/2010_State_of_Agile_Development_Survey_Results.pdf (accessed July 2014).

VersionOne, 2011. 6th Annual State of Agile Development Survey http://www.versionone.com/pdf/2011_State_of_Agile_Development_Survey_Results.pdf (accessed July 2014).

VersionOne, 2013. 7th Annual State of Agile Development Survey. http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf (accessed July 2014).

VersionOne, 2014. 8th Annual State of Agile Survey. http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf (accessed July 2014).

VersionOne, 2014. 8th Annual State of Agile Survey. http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf (accessed July 2014).

Wells, D., 2013. Daily Stand Up Meeting. http://www.extremeprogramming.org/rules/standupmeeting.html (accessed July 2014).

Zijlstra, F.R.H., Roe, R.A., Leonora, A.B., Krediet, I., 1999. Temporal factors in mental work: Effects of interrupted activities. Journal of Occupational and Organizational Psychology 72, 163–185. doi:10.1348/096317999166581

Zijlstra, F.R.H., Roe, R.A., Leonora, A.B., Krediet, I., 1999. Temporal factors in mental work: Effects of interrupted activities. Journal of Occupational and Organizational Psychology 72, 163–185. doi:10.1348/096317999166581