Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 121 (2017) 123–128

Procedia Computer Science

www.elsevier.com/locate/procedia

CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies, CENTERIS / ProjMAN / HCist 2017, 8-10 November 2017, Barcelona, Spain

Coordination in multi-team programmes: An investigation of the group mode in large-scale agile software development

Torgeir Dingsøyr[a,b,*], Knut Rolland[a], Nils Brede Moe[a], Eva Amdahl Seim[a]

[a]*SINTEF, PO BOX 4760 Sluppen, 7465 Trondheim, Norway*
[b]*Department of Computer, Norwegian University of Science and Technology, 7491 Trondheim, Norway*

Abstract

Coordination of work teams is critical when managing large programmes that involve multiple teams. Prior studies of knowledge work indicate that such work relies heavily on coordination through "personal" modes such as mutual adjustment between individuals or through scheduled or unscheduled meetings. We studied how coordination through scheduled and unscheduled meetings change over time in two large software development programmes. Findings include transitions from scheduled to unscheduled meetings and from unscheduled to scheduled meetings. The main implication is that programme management needs to be sensitive to the vital importance of coordination as well as the coordination needs as they change over time.

Keywords: coordination, programme management, agile software development, software engineering.

* Corresponding author. Tel.: +47 930 08 714
E-mail address: torgeird@sintef.no

1. Introduction

Coordination of work teams are of critical importance when managing large projects that involve multiple teams. Multi-team projects are used in many domains, often to "*achieve high quality innovations in a satisfactory time-to-market*"[1] and in such programmes "*hundreds of people may be required to develop components of a new product simultaneously*"[1]. Much of the resources used on innovations today are used on software development. Coordination was early identified as a particular challenge in software development projects. In the 90ies, software projects were often associated with overruns on time and cost, and many referred to a "software crisis". As Kraut and Streeter[2] state, "*While there is no single cause of the software crisis, a major contribution is the problem of coordinating activities while developing large software systems. We argue that coordination becomes much more difficult as project size and complexity increases.*"

Since then, new methods for software development have been suggested, what is referred to as agile software development[3, 4]. The practices in this field have also inspired the project management discipline[5]. These methods were however, intended for small, self-managing and co-located teams. Nevertheless, the popularity of these methods has spurred use also in large development programmes. This article examines a specific type of coordination practice in large-scale agile development programmes: The use of the "group mode". Prior studies of knowledge-work indicate that such work relies heavily on coordination through "personal" modes such as mutual adjustment between individuals or through scheduled or unscheduled meetings. We analyse how coordination through scheduled and unscheduled meetings ("group mode") change over time in two large software development programmes that make use of agile development methods, and thus emphasize informal coordination. We ask the research question: *How are group mode coordination practices used in large-scale agile development?*

2. Coordination modes

A common understanding of coordination is to manage dependencies between e.g. tasks, resources or technology[6]. Three main determinants for coordination mechanisms are identified in prior literature[7]: *Task uncertainty* - the "difficulty" and "variability" of work undertaken by an organizational unit. Higher degrees of complexity, thinking time to solve problems or time required before an outcome is known indicates higher task uncertainty. *Task interdependence* - the extent to which persons in an organizational unit depend on others to perform their work. A high degree of task-related collaboration means high interdependence. *Size of work unit* - the number of people in a work unit. Increases in participants in a project or program means an increase in work size unit.

There are a number of mechanisms that can be applied to achieve coordination, and coordination is usually exercised through several mechanisms[8]. Van de Ven et al.[7] proposes three coordinating modes, which is used by Dietrich[8] in their study of multi-team projects: by programming or codification (impersonal mode), and coordination by feedback (or "mutual adjustment"[9]) on the individual (personal mode) or on a group level (group mode). The impersonal coordination mechanisms are codified, and require minimal verbal communication between people once implemented. Examples include pre-established plans, process documentation, intranet pages and roadmaps. Coordination by mutual adjustment or feedback is based on informal communication. In the personal mode, individual role occupants serve as the mechanism for making mutual task adjustments through either vertical or horizontal channels of communication. The mechanisms for vertical communication are usually line managers and unit supervisors. In the group mode, the mechanism for mutual adjustment is vested in a group of role occupants through scheduled or unscheduled meetings.

Software projects often solve complex tasks with high uncertainty. Van de Ven et al.[7] found that increase in task uncertainty leads to a substitution of the impersonal coordination with horizontal coordination mechanisms and group meetings. High task uncertainty gives a need for extensive and dynamic knowledge exchange to solve problems and adjust for emerging changes[7]. Dietrich[8] also point to prior studies which found that technological novelty relate to a higher rate of group meetings instituted by management. The scheduled meetings are effective because physical proximity allows richer communication which enables swifter and more flexible coordination[10]. However, coordination in the group mode based on mutual adjustment requires everyone to communicate with everyone. With 5 people there are 10 links, and with 15 people 120 unique links. Therefore, to employ mutual adjustment as the prime coordinating mechanisms, groups need to be kept dense - and, since our communication abilities are limited, that means they also have to be small[10]. Increased unit size, however, is associated with greater use of impersonal coordination and hierarchy (but no decrease in group mode coordination)[8]. In large software projects, mutual adjustment can take part within smaller teams, or groups of team representatives acting on behalf of their group. Such coordination can be understood as layered mutual adjustment, usually with hierarchy and bureaucratic control[10]. A key challenge with layered mutual adjustment based on hierarchy pertains to complex problem solving. It is challenging to plan who should be involved in which decisions, and to communicate all-important decisions. Scheduled meetings are typically used for routine meetings, involving planned communication, while unscheduled meetings are used for unplanned communication between more than two participants. In agile development using Scrum, group mode coordination at team level is ensured through sprint (or iteration) planning meetings, daily scrum meetings, sprint demonstration meetings and retrospectives[11, 12]. Layered mutual adjustment is ensured in large scale agile by e.g. Scrum of Scrum[13].

3. Method

This study builds on two broad case studies of large-scale development programmes *Alpha* and *Beta*, which investigates how agile methods were adapted in the very large scale. Previous studies[14, 15] show how large development programmes dealt with method tailoring, technical architecture, customer involvement and inter-team coordination. We have taken material from two cases and further analyzed our data material on coordination focusing on use of the group mode (see characteristics of the programmes in Table 1).

Alpha was chosen because practitioners described it as a successful very large program that used agile development methods to a large degree. The whole program was co-located, and coordination mechanisms could be studied in a setting that is well suited for agile methods. The *Alpha* program developed a new office automation system for a public department. The program was managed by the department and involved two main consulting companies as subcontractors in the project *development*.

Beta was selected as one of Norway's largest IT-projects with extensive use of agile methods. The project involved complex integration among a wide variety of internal and external information systems, involving various stakeholders with divergent interests. Moreover, before starting *Beta*, the supplier, an international consulting company had been part of *Alpha* that was celebrated nationally as a great success. This project is often used as a template for other large-scale agile projects in Norway.

Our study draws on the established tradition with theoretically informed interpretive case studies in information systems[16, 17] and hence aims at following relevant guidelines for such research[18, 19].

Both programmes were planned according to a model based on PRINCE2[20] with distinct phases. The development programs were conducted using the agile development method Scrum in the construction phase, but this phase was preceded by "analysis of needs" and "solution description" phases and followed by an "approval" phase for each release of the developed product.

Table 1. Characteristics of the *Alpha* and *Beta* programmes.

Characteristic	Alpha programme	Beta programme
Number of people involved at the most	175	120
Number of development teams	12	5
Employees in customer organization	380	7 000
Duration	2008-2012	2011-2014

Our data collection started when the programs were finished, using individual interviews in Beta, group interviews in Alpha and internal and external documents for both cases as shown in Table 2. We analyzed the material in a tool for qualitative analysis, focusing here on reporting findings related to group mode coordination.

Table 2. Data collection from the *Alpha* and *Beta* programmes.

Data source	Alpha programme	Beta programme
Individual interviews	0	27
Group interviews	9 two-hour interviews with a total of 24 participants	0
Documents	External experience report	Tender documents
	Internal experience report	Project documents such as plans and scope
		IT-strategy documents

4. Results

Group mode coordination took place through a number of scheduled meetings as well as arenas for unscheduled meetings as shown in Table 3. We first describe scheduled meetings at program and project levels, first describing meetings that existed in both programs and then describing the difference between the programs:

At program level, the only arena where everyone would meet was at the demonstration meetings, which were held every three weeks. In addition, the program management met two times a week in a forum, which was called "Metascrum". The Metascrum included managers from the main projects and the central program management, giving attention to "high-level" obstacles to progress and assessment of risks in the program. At *Alpha*, a new arena was introduced well into the program, the "open space technology". Open space was a way to get the whole program to discuss challenges and improvement initiatives. In addition, there were separate meetings to identify dependencies in tasks before work was assigned to teams. At *Beta*, the meetings varied over the nearly four years of development, but meetings concerning overall software architecture, project managers meeting, and project owners meeting were conducted regularly. These meetings involved participants from both the Consultant Company and the Customer. In the later part of the programme, a meeting referred to as the "Bug Board" was also established to coordinate actions for solving critical problems on technical issues, mercantile issues or processual issues.

In *Alpha* and *Beta* at the project level, there were three main types of scheduled meetings: The meetings prescribed by the agile method Scrum, meetings in the main projects in the program, and fora at project level to share experience across the development teams.

Scrum of Scrums were held in the three development subprojects at *Alpha* and in the main programme at *Beta* with Scrum masters and subproject managers from 3-6 development teams. Project managers sometimes participated in these meetings. One subproject at Alpha had daily Scrum of Scrum meetings in the beginning, but reduced the frequency to three times per week. A topic discussed here were resources, "now we have two people who are ill in the team, and we have given away a person to the environment team, how shall we manage to deliver our stories in the iteration?" (subproject manager). In addition, retrospectives were sometimes held across teams in the subprojects, but overall this was an activity within each team.

In *Alpha*, the projects architecture, business and test had meetings with their own staff and the people who held roles in the development teams. In the business project, much of the work concentrated on managing dependencies, "there were dependencies throughout the program" (technical architect). One of the participants in meetings in the business project said, "when we talked to the product owner, the product owner said, "we need you to do this", but then we had to explain that to achieve that we first need to do these tasks" (functional architect). The meetings in project architecture focused on establishing architectural guidelines, but also focused on coordinating work amongst the development teams to reduce the number of teams working on the same part of the codebase. "This was to reduce the possibility of making trouble for each other - which we did". The codebase was organized to reduce these challenges and in meetings teams declared that "this is our central area of work this period, so please limit work in that area" (technical architect).

In *Beta*, also several other meetings for coordinating across teams and roles were established in the later parts of the programme. Most profoundly, some members from different teams to coordinate and uncover interdependencies involved in the following sprint first practiced a meeting referred to as "ready-to-sprint". This meeting turned out to be crucial to distribute work in a way that made the different teams work as autonomous units as far as possible. These meetings had different participants as roles and individuals relevant for uncovering and analyzing interdependencies varied from sprint to sprint. These meetings first grew out of the pressing need for coordinating across teams experienced by individual team members, and later sanctioned by project managers as a practice to adopt in a more systematic manner.

Experience-sharing across teams were the focus of several scheduled meetings at sub-project level: "Experience forum", "Lunch seminars" and "Technical corner" are examples of meetings that existed during the *Alpha* program. A topic discussed at the experience forum, was how to liven up the retrospectives, this was then a topic discussed amongst all participants in the development teams in one project. Participation in these meetings was voluntary.

Unscheduled meetings were easy to organize due to the open workspace. Unplanned meetings frequently took place around the boards that were available for each team. These were used to "discuss solutions, draw and make sketches" (subproject manager). These discussions spanned development teams and roles. The project management was placed on tables so that they could see most of the boards and thus quickly get an overview of status of the teams. If the project managers noticed discussions, they could inquire about the issue and say that "this problem I know was addressed by another team two iterations ago, let us get "Ola" over here and see if he can help" (subproject manager, *Alpha*). A Scrum master and developer stated that they learned "very much" in the program during these discussions around the boards, but it was important to have sufficient coordination arenas so that people realize that "we need to talk". The program also started to use a group chatting tool (Jabber) to ease informal coordination, what we can see as a type of unscheduled virtual meetings. This tool was introduced during the program, which enabled asking several people for help without interrupting them. This channel was used for several purposes, from asking technical questions to inform about the next wine lottery.

Informants emphasized the importance of the unscheduled meetings. One said, "I think the combination of scheduled and unscheduled coordination that just appeared was very important" (scrum master and developer, *Alpha*).

Table 3. Examples of scheduled and unscheduled meetings in programmes *Alpha* and *Beta*.

Examples of meetings	Alpha programme	Beta programme
Scheduled	Metascrum	Metascrum
	Scrum of Scrums	Srcum of Scrums
	Subproject meetings	Architecture meeting
	Open Space	Ready-to-sprint meeting
	Experience forum	Bug board
	Lunch seminars	
Unscheduled	Open work area	Open work area
	Group chat tool	Ad hoc coordination

5. Discussion and conclusion

We have described use of group mode coordination in two large-scale software development programmes using the agile development method Scrum. We discuss our research question "*how are group mode coordination practices used in large-scale agile development?"* through emphasizing two main findings and their implications:

First, we see that the group mode is extensively used as shown by the large number of scheduled and unscheduled meetings in the programmes. Table 3 shows examples of these meetings, and we see that scheduled meeting arenas include both meetings prescribed by Scrum as well as a number of other meetings. The meetings related to the agile method Scrum were kept throughout the program, and the iteration length remained at three weeks. We believe that having many meetings was important to build knowledge and relations early in the program. Van de ven et al.[7] state that an increase in task uncertainty has been found to lead to a substitution of impersonal coordination with horizontal coordination mechanisms and group meetings. Our findings support this position.

Second, we have described two main transitions over time within the group mode: At Alpha, there was a high number of scheduled meetings initially, but a gradual transition to unscheduled meetings. Informants state that the initial scheduled meetings were very important for efficient use of unscheduled meetings later. At Beta, we find examples of unscheduled meetings that were formalized as the programme management identified the importance of these meetings.

Our study supports the finding that group mode coordination is central to achieving inter-team coordination in large programs, and in particular, we highlight the role of unscheduled meetings to achieve effective coordination in knowledge work. The project management frameworks such as PRINCE2 puts emphasis on activities and roles, which can influence project managers to establish scheduled meetings. For knowledge, work, we believe the unscheduled meetings are of great importance and project managers should strive to foster these meetings. Programme management needs to be sensitive to the vital importance of coordination as well as the coordination needs as they change over time. In the future, we plan to develop a further understanding of the "layered mutual adjustment" we have identified in large-scale software development programmes. One of the limitations of this study is that we have collected research data after programme completion. In the future, we would like to perform data collection during programme execution in order to more closely examine changes over time.

Acknowledgements

This work was supported by strategic internal projects at SINTEF on large-scale agile development and the project Agile 2.0 supported by the Research council of Norway through grant 236759 and by the companies Kantega, Kongsberg Defence & Aerospace, Sopra Steria, and Sticos.

References

1. Hoegl, M. and Weinkauf, K., "Managing task interdependencies in Multi‐Team projects: A longitudinal study," *Journal of Management Studies,* vol. 42, pp. 1287-1308, 2005.
2. Kraut, R. E. and Streeter, L. A., "Coordination in software development," *Communications of the ACM,* vol. 38, pp. 69-81, 1995.
3. Abrahamsson, P., Salo, O., Ronkainen, J., and Warsta, J., "Agile software development methods: Review and analysis," VTT Technical report2002.
4. Dingsøyr, T., Nerur, S., Balijepally, V., and Moe, N. B., "A Decade of Agile Methodologies: Towards Explaining Agile Software Development," *Journal of Systems and Software,* vol. 85, pp. 1213-1221, 2012.
5. Conforto, E. C., Salum, F., Amaral, D. C., da Silva, S. L., and de Almeida, L. F. M., "Can agile project management be adopted by industries other than software development?," *Project Management Journal,* vol. 45, pp. 21-34, 2014.
6. Malone, T. W. and Crowston, K., "The interdisciplinary study of coordination," *ACM Computing Surveys,* vol. 26, pp. 87-119, Mar 1994.
7. Van de Ven, A. H., Delbecq, A. L., and Koenig Jr, R., "Determinants of coordination modes within organizations," *American sociological review,* pp. 322-338, 1976.
8. Dietrich, P., Kujala, J., and Artto, K., "Inter‐Team Coordination Patterns and Outcomes in Multi‐Team Projects," *Project Management Journal,* vol. 44, pp. 6-19, 2013.
9. Mintzberg, H., *Mintzberg on management: Inside our strange world of organizations*: Simon and Schuster, 1989.
10. Groth, L., *Future organizational design: the scope for the IT-based enterprise*, 1999.
11. Strode, D. E., Huff, S. L., Hope, B. G., and Link, S., "Coordination in co-located agile software development projects," *Journal of Systems and Software,* vol. 85, pp. 1222-1238, 2012.
12. Xu, P., "Coordination in large agile projects," *The Review of Business Information Systems,* vol. 13, p. 29, 2009.
13. Paasivaara, M., Lassenius, C., and Heikkila, V. T., "Inter-team Coordination in Large-Scale Globally Distributed Scrum: Do Scrum-of-Scrums Really Work?," in *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ed New York: IEEE, 2012, pp. 235-238.
14. Dingsøyr, T., Moe, N. B., Fægri, T. E., and Seim, E. A., "Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation," *Empirical Software Engineering,* pp. 1-31, 2017.
15. Rolland, K. H., Fitzgerald, B., Dingsøyr, T., and Stol, K.-J., "Problematizing Agile in the Large: Alternative Assumptions for Large-Scale Agile Development," in *International Conference on Information Systems*, Dublin, Ireland, 2016.
16. Walsham, G., "Interpretive case studies in IS research: nature and method," *European Journal of Information Systems,* vol. 4, pp. 74-81, 1995.
17. Walsham, G., "Doing interpretive research.," *European journal of Information Systems,* vol. 15, pp. 320-330, 2006.
18. Klein, H. K. and Myers, M. D., "A set of principles for conducting and evaluating interpretive field studies in information systems," *MIS Quarterly,* vol. 23, pp. 67-93, 1999.
19. Sarker, S., Xiao, X., and Beaulieu, T., "Guest editorial: qualitative studies in information systems: a critical review and some guiding principles," *MIS Quarterly,* vol. 37, pp. iii-xviii, 2013.
20. Bentley, C., *Prince2: a practical handbook*: Routledge, 2010.