Pitfalls in Remote Team Coordination: Lessons Learned from a Case Study

Darja Šmite[1], Nils Brede Moe[2], and Richard Torkar[3]

[1] University of Latvia, Latvia
[2] SINTEF, Norway
[3] Blekinge Institute of Technology, Sweden
Darja.Smite@lu.lv, Nils.B.Moe@sintef.no, Richard.Torkar@bth.se

Abstract. As companies become more and more distributed, multi-site development is becoming a norm. However along with the new opportunities, geographic distribution is proven to increase the complexity of software engineering introducing challenges for remote team communication, coordination and control. In this article we present an illustrative singe-case study with an intra-organizational intra-national context focussing on the effect of geographic distribution on team coordination practices and how this influences remote team performance. Based on our findings we conclude that a) distribution significantly influences the nature of coordination; b) remote team coordination mechanisms can't be chosen disregarding the complexity of the given tasks and c) the distribution of work on complex software development tasks shall be avoided.

1 Introduction

Distributed software teams, in which members interact with one another across geographic, organizational, and other boundaries, are becoming commonplace in software organizations. Distributed software teams can be composed of the best individuals for the task regardless of their physical or organizational location, thus enhancing the quality of software development. However, problems of geographic distribution, such as decreased speed of work [1, 2], loss of communication richness [2, 3] and coordination breakdown [2, 3] experienced in highly distributed global software engineering projects may also be faced in an intra-organizational intra-national software development environment.

While most opportunities of work distribution are found on the business level, most challenges are introduced at the level of development practice [4]. Lacking knowledge and expertise, managers tend to underestimate the complexity of remote team coordination. Lack of proximity and ability to use well-known proven practices for team coordination makes project managers uncomfortable and insecure. Subsequently, missing trust in a distributed project may even lead to termination of further collaboration [5].

Coordination of work is an important aspect of teamwork and team leadership [19]. Coordination together with communication and collaboration are recognized as the key enablers of software development processes [20].

A. Jedlitschka and O. Salo (Eds.): PROFES 2008, LNCS 5089, pp. 345–359, 2008.

Motivated by the importance of distributed work and software development teams, the objective of this paper is to explore and discuss the interrelation between geographic distribution, team coordination styles and team performance. The core research questions are therefore:

RQ1: What are the consequences of geographic distribution on how work is coordinated in a distributed software team?
RQ2: What characterizes the effect of different coordinating mechanisms on distributed team performance?

We first illustrate the nature of remote team coordination with a case study and then use related literature to understand the effect of different coordination mechanisms on the remote team performance.

The rest of the paper is structured in the following way: in section 2 we describe related literature. Section 3 describes our research method in detail. In Section 4, we present results from a case study on work coordination in a distributed environment. We discuss our results and research questions in section 5. Finally, recommendations on how to coordinate work in a distributed project conclude the paper.

2 Related Research Overview

2.1 Coordination of Software Work

Software development consists of solving tasks with different complexity, and different task complexities require different coordination mechanisms [6]. Mintzberg [6] proposes the following coordination mechanisms:

1. Mutual adjustment—based on the simple process of informal communication, achieved by a continuous exchange of information among participants;
2. Direct supervision—one person takes responsibility for the work of others by issuing instructions and monitoring their actions;
3. Standardization—of which there are four types: work processes, output, skills (as well as knowledge), and norms.

The coordinating mechanisms may be considered as the most basic elements of structure, the glue that holds the organization together [6]. The mechanisms may act as substitutes for each other to some degree, but all will typically be found in a reasonably well-developed organization.

Simple tasks are easily coordinated by mutual adjustment, but when work becomes more complex, direct supervision tends to be added and takes over as the primary means of coordination. When things get even more complicated, standardization of work processes (or outputs) take over as the primary coordinating mechanism, in combination with the other two. Then, when things become really complex, mutual adjustment tends to become primary again, but in combination with the others (Fig. 1). This means that even though mutual adjustment should be the most important coordinating mechanism when developing software even in a global context, the need for the different coordinating mechanisms depends on the different tasks.

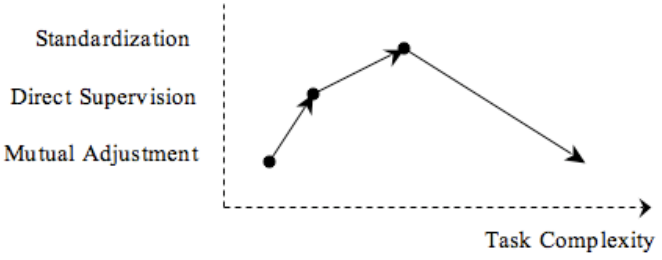

Fig. 1. Complexity of task [6]

Mutual adjustment and direct supervision can be categorized as coordinating by feedback [18], where coordination is adjusted continually as people observe the effects of their own and others' actions. However, geographic distribution is associated with reduction of informal contact, lack of proximity and task awareness [4], which are crucial for applying these coordination mechanisms. One dominant perspective on software development is rooted in the rationalistic paradigm, which promotes a product-line approach to software development using a standardized, controllable, and predictable software engineering process [19]. This perspective is inspired by a mechanistic worldview [20], making standardization of work processes the most important coordinating mechanism. Unfortunately, organizational diversity and disparities in work practices make it difficult to only rely on standardization [5]. These problems are often underestimated and discovered too late in the projects [5].

2.2 Coordination of Distributed Software Work

Research in globally distributed software development (GSD) is in the phase, where case studies of various kinds (in most cases having a qualitative focus) provide input for researchers and organizations to, in the end, improve upon activities in this context. The same applies to this paper and hence relevant work can be found in different contributions where lessons learned, models, problem descriptions and communication patterns are analyzed using a case study approach.

Coordination in distributed development and especially in GSD has caught the attention of several researchers. Cataldo et al. [7] present four case studies, where they exemplify coordination breakdown problems in GSD focusing on how these problems could still occur in spite of supporting tools and processes (one of the findings was to emphasize lateral communication in GSD projects). Holmström et al. (a) [8] present a number of challenges with respect to temporal, geographical and socio-cultural distance and combined with the conclusion that GSD should be trimmed down towards nearshoring. Additionally, Holmström et al. (b) [4] (containing two case studies) investigate how agile processes can help in reducing different types of "distances", i.e. they found some indications that agile practices may assist in alleviating some of the problems in GSD.

Crowston et al. [9] try to explain the performance of teams working for Free/Libre Open Source Software (FLOSS) from a GSD perspective. The conclusion is that the

problems faced by these teams can be recognized as human-centred. In this paper we further examine this issue from an intra-organizational intra-national industry context.

Herbsleb et al. [10] present a number of lessons learned from nine distributed projects. They propose future research on different dimensions of coordination mechanisms and how they play together. In this paper we look at communication patterns, project management issues, the effect of traveling and different communities of practice, i.e. issues which Herbsleb et al. introduced in [10].

Finally, at least three other types of research contributions can be found in the field of coordination research in GSD. First, the quantitative studies (which are fairly uncommon in GSD research) of Herbsleb and Mockus [1], where they discovered that distributed work is significantly slower than co-located through mining source code repositories for further analysis. Second, contributions covering tool support in a GSD context; here Boden et al. [11] and Fomseca et al. [12] can serve as good examples. Third, and final, we find contributions focusing on coordination analysis [13] and predictions [14] that help in better understanding coordination issues in GSD.

3 Research Design and Methodology

3.1 Study Context

To contribute to the existing literature on remote team coordination we exemplify the nature of coordination mechanisms in a geographically distributed environment by a single-case study run in a Northern European software organization nationally distributed across two locations. The development project involved a customer from Norway that engaged the case company that in its turn outsourced some work to its remote location within the national borders. In this study we focused on the intra-organizational relationship, where the central team with several years of experience as a supplier, was now put into an engaging partner role.

The company made a decision to employ developers in one of the poorest regions in the same company for decreasing their development costs. The company rented an office and employed a group of local software engineers all working as developers.

The goal of the project was to deliver a web-based e-commerce application system built around mordern technology. The project started in May 2005 and ended in April 2006. In total 1,460 man-days were used exceeding the planned effort by 12%.

The project suffered from ambiguity in the initial set of requirements because a great amount of changes was reported by the end customer during the project. Therefore, schedule deviations were foreseen in advance and negotiated with the customer without extending the deadlines. However this caused extra work during the weekends for the development teams. Additional expenses were not planned by the customer and resulted in the scope of the project being narrowed. As a result, testing activities were limited and several requirements were not implemented. This sequentially decreased the end customer satisfaction.

The quality of the delivered software was perceived as poor by the project manager due to a high amount and severity of bugs uncovered during system testing (1,144 bugs) and acceptance testing (220 bugs). The manager regularly expressed his low satisfaction with the performance of the remote team, which served as a motivation

for our investigation. In addition to the project goal, the project manager aimed to verify the suitability of the remote team for future globally distributed projects.

Process Distribution: The entire project was divided into a set of subsequential phases. Most of the activities were performed on site by the central unit. Development tasks were distributed between the on-site and off-site (central and remote) teams. However, the tasks were allocated according to the independent software architecture modules and primarily did not request close dependency between the distributed programmers. Table 1 shows the lifecycle process distribution between the teams:

- Customer unit consisting of 2 people (overall project manager and systems analyst)
- Central supplier unit consisting of 6 people (project manager, 2 system analysts, 2 testers and a programmer)
- Remote supplier unit consisting of 5 people (a team leader and 4 programmers).

The project manager and the remote team leader have worked in the studied company for 3 years. System analysts and testers had more than 10 years of experience in the same company. Programmers' experience varied between 2-5 years.

Table 1. Lifecycle process distribution

Lifecycle activities	Customer	Supplier: Central unit	Supplier: Remote unit
Requirements Analysis	X	X	
High Level Design		X	
Detail Architecture		X	
Development		X	X
Unit Testing		X	X
System Testing		X	
Acceptance Testing	X	X	

Thus, software processes in this project were distributed between the members of an intra-organizational and intra-national team. According to Prikladnicki et al. in [17] the project can also be characterized as an internal domestic supply.

Task Complexity: The developed software product was perceived as a complex system from both an architectural and technological perspectives. This was particularly discussed when the project was summed up in the end during the post mortem meeting. Another reason for work complexity was lack of experience of the developers with the new technologies the project was built on and unfamiliarity with the business domain. This required close cooperation and joint problem solving.

Process Quality Assurance: Since the supplier organization implemented a quality system and certified its processes according to the ISO 9001:2000 standard, the project followed standardized guidelines for requirement specification, task management, progress reporting and monitoring, and other activities. The project manager established a specifically tailored quality plan describing the procedures to follow. Weekly teleconferences were organized to discuss urgent problems and plans. A special tool was used for allocating tasks and monitoring progress. This tool was also used to monitor the workload of each developer.

3.2 Data Sources and Analysis

In this case study we have used multiple data sources: individual and group interviews, post mortem analysis, risk survey and participant observations; all collected by the first author, to identify problems related to distribution and the effect of different coordination styles on team performance.

We conducted four individual qualitative interviews, which were from 30 to 60 minutes long in the central location and one a 90 minutes group interview with the remote development unit in their location. We also organized a post mortem analysis meeting [16] at the end of the project. The meeting involved all project members in the discussion of what went well and what did not work in the project, concluded with a root-cause analysis of the major issues. In addition, a risk checklist was distributed to the central project manager and the remote team leader in order to identify problems related to collaboration. The first author of this paper was also involved in the quality assurance and process improvement activities during the last 6 months of the project. This enabled the possibility to use participant observation [16] when observing both teams. In this study we are mainly relying on qualitative interviews [15] and participant observation [16].

Multiple sources of evidence enabled triangulation when analysing the material. This helped us increased validity and allowed to create a broad view of the project, also collecting the "unsaid issues"; the remote team reported many problems and was open to the researcher during the group interview performed at their premises, however, they did not to report any problems related to collaboration, communication or coordination during the post-mortem meeting.

The qualitative analysis of the data was performed in several steps. First we analyzed the problems reported and observed from both teams separately. This allowed us to see the diversity of evidence dependent on the chosen investigation method and the team perspective. Next we analyzed the interrelation between the problems, geographic distribution and related them to the three coordination mechanisms. At the end, we tracked the influence of different coordination mechanisms on the team behaviour.

We believe that the selected case is internally representative, since the studied project involved all of the employees in the remote location. The internal validity could be improved by observing several collaborative projects, but this was not possible to address, since the selected project was the only ongoing project between the central and the remote locations at the time of investigation. We also believe that such factors as poor working environment that may have affected the remote team motivation shall be seen as a part of the supply chain management and coordination.. Our study is limited because it is only validated in the context of a Northern European intra-organizational intra-national collaboration and therefore has potential threats to external validity. Hence, the findings cannot be generalized widely to practitioners from other countries. To improve the potential for external validity of our single-case study, we applied existing theory as recommended by Yin (2003) and emphasize our contribution as exploratory-illustrative rather than theory-building. Reliability of the study can be judged upon the description of the methodology used and the context of the intra-organizational intra-national collaboration.

4 Coordination of Work in a Distributed Project

We now present how the different coordinating mechanisms were used in the case study and discuss the reasons of coordination pitfalls.

4.1 Standardization

Face-to-face contact is the richest communication channel we have, and any electronic channel is significantly poorer [18]. Hence, if coordination is based on coordination by programme (strandardization) in a distributed team, where coordination is affected through instructions and plans generated beforehand, this will reduce the level of face-to-face communication, which again tends to hinder effective communication and the possibility for coordination by mutual adjustment.

Case-study: Standardization was selected as a prior ground for project coordination, following processes certified according to the ISO 9001:2000 standard. The project manager established guidelines (project quality assurance plans, communication plans, different software development related process handling guidelines) for the team and expected everybody to follow them.

As the project went on, the project manager realized that even though the work processes were standardized, there were disparities in work practices between the central and remote units. Not working according to the established processes as expected, resulted in a lack of understanding of the context of decision-making, and decreased the level of trust between the distributed team members. These disparities considerably decreased the predictability of the team member performance and mutual cohesion.

The project manager reported that the remote team performance was unexpectedly low. Although the collocated team members in the remote location were cohesive, their informality caused problems for team coordination: the systems analysts from the central location reported that the remote team unit acted “as a joint body”, independently shifting their work tasks and interpreting unclear requirements. The remote team was able to act independently, which was good, but making operational decisions without sufficient information often caused subsequent rework. Coordination by standardization didn’t encourage frequent feedback from the remote location, therefore, it caused delays and absence of a jont problem resolution, resulting in the remote team making operational desicions on insuficcient grounds . In addition, due to the remote team’s independent behaviour, the project manager felt he had a limited visability of what was done by whom and when, which he felt caused an inability to effectively coordinate and plan the remote team’s work load. This illustrates that despite the official project guidelines, according to the project manager the remote team acted differently than expected.

Often lacking awareness of the distant activities, the project manager perceived the remote team to be less productive than expected. The amount of defects uncovered during testing was percived as too high and decreased his cognition even more. However, there could be another explanation for this. Despite the standard technological infrastructure, the remote team suffered from old-fashioned computers and slow communication lines. This required several hours for remote code compilation per

day. Thereby, unjustified application of new unstable architectural solutions caused additional rework. These problems were reported to the project manager only at the end of the project during the post-mortem analysis meeting. However, uncertainty of the remote activities and not satisfied with the quality delivered, resulted in the desire of the project manager to increase the use of direct supervision for team coordination during the project.

Discussion: Relying on coordination by standardization has led to a list of problems in relation to a) disparities in work practices, b) little feedback, and c) lack of transparency of the remote activities.

The project manager chose standardization as the primary coordinating mechanism independent of the tasks to be solved, and their complexity. While the remote development team struggled with technological issues because of an unknown development platform in addition to an increased number of changes, lack of mutual adjustment reduced the possibility for solving these issues effectively and a joint decision-making process between the distributed teams. Change-driven agile practices that were recognized as key for the remote team cohesion went into conflict with the plan-driven standardized work practices of the central unit. These disparities caused misunderstandings, e.g. project guidelines prescribed each developer to report their effort individually, however, the reported progress could be misleading, since there was a high level of workload shift between the remote unit members.

4.2 Direct Supervision

According to Takeuchi and Nonaka [21] management should establish enough checkpoints to prevent instability, ambiguity, and tension from turning into chaos. At the same time, managers should avoid the kind of rigid control that impairs creativity and spontaneity [21]. Therefore direct supervision must be used with care.

Case-Study: Because the project manager lacked previous experience with remote team coordination and coordination by standardization was problematic because of the limited feedback and disparities in work practices, he then started relying more on direct supervision as the most important coordination strategy.

The remote unit communicated directly with the systems analysts and testers whenever necessary. Some of the tasks and most of the problem reported were also coordinated directly without the project manager's and remote team leader's involvement. This produced a situation where the local project manager had a feeling that he was not in charge of all the activities, which together with the geographic distribution and lacking transparency caused him a "headache" because he felt he lacked control of the project. The project manager also reported that it was difficult to spread awareness of everyday activities and more importantly rapid changes across the distance. Feeling he had no control over the remote team, he increased monitoring and started requiring daily progress reports. At the end of the project, he even suggested to install a video camera on the remote site.

Feeling not trusted and afraid of collaboration determination, the remote team started to report even fewer problems. This again resulted in the project manager perceived the remote team as inactive and lacking initiative in the weekly teleconferences, often

remaining silent. This again limited his ability to supervise the remote team and decrease the cognition even more.

Discussion: Geographic distribution made remote team coordination a complex task and led the project to a chain of problems in relation to a) lacking proximity and transparency, b) fear of losing control, c) lack of trust and d) decreased feedback.

Our case illustrates a closed loop that starts with a lack of trust and belief of the project manager in the remote team's ability to perform, leading to increased monitoring and supervision, which eventually leads to a lowered morale of the remote team, unwillingness to collaborate and little feedback. The remote team's silence and delays given the lowered levels of trust are misinterpreted by the project manager and trust decreases even more. Again, this increases the project manager's desire to monitor. It was a deadlocked situation.

When a manager in whom the employee has little trust gives negative feedback, it is likely that the employee will doubt the accuracy of the feedback [23, 24]. This hinders the team leader from managing the team effectively.

4.3 Mutual Adjustment

Mutual adjustment in its pure form requires everyone to communicate with everyone [18]. Therefore to employ mutual adjustment as the prime coordinating mechanisms the team or network need to be dense and co-located, and since our communication abilities are limited, that means they also have to be small [18]. Usually there is a limited possibility for face-to-face communication in a GSD project.

Case-Study: Because the project managers first relied on standardization, mutual adjustment was not seen as an option for coordination with the remote team. When standardization did not work as expected, he started relying on direct supervision was increased, but because of problems with trust and giving feedback, the level of mutual adjustment was reduced even more. Achieving effective mutual adjustment was also difficult because of the geographic distribution, no travelling to the remote office and only relying on instant messaging tools and phone as primary communication means.

As a consequence of heavy monitoring, both sides reported the decreased level of trust; which again reduced the possibly for mutual adjustment even more. Though the remote unit had previous work experience with the system analysts from the central location.

Geographic distribution and lack of transparency caused psychological discomfort for the project manager. He claimed: *"I am not sure if they are working over there"*. The project manager recognized he lacked a belief in the remote team's ability to perform and this was also felt by the remote team lead and reported through the risk survey. A number of contributions in other research disciplines has reported on numerous occasions that studied and analyzed this type of organizational behaviour e.g. [3, 25]. Trust was also affected by the lack of face-to-face meetings, poor socialization, too little communication, misunderstood silence and unwillingness to discuss collaborative problems. In the project this was a significant impediment to apply mutual adjustment for team coordination.

The remote team claimed that there was an increased amount of seemingly unimportant questions that were never communicated through distance, however thei

would have been resolved if the systems analysts travelled on a regular basis to the remote location.

The diversity of social situation in the capital city and the small town increased the gap between the two distributed teams. The benefits that were organized for the central location were not offered for the remote location. Their office got low technological infrastructure, old computers and communication lines. The remote-team leader complained: *"It took some time to even organize the supply of drinking water. It was not easy to convince the management to order the service and it wasn't easy to find the service supplier in our region"*. After feeling the lack of trust and belief in their performance from the central location, they were afraid to complain about their problems. This subsequently decreased the team's psychological comfort and ability to rely on mutual adjustment.

Discussion: Several of the problems described are examples of impediments to establishing mutual adjustment in geographically distributed projects. The case shows that geographic distribution and missing trust affected the project manager's and the team's behaviour. While the project manager didn't trust and believe in success of collaboration, the team tried to hide their problems. Given development tasks that were perceived as complex, in addition to frequent changes, stressed by the deadlines and the usage of unfamiliar technologies, the remote team relied on mutual adjustment to coordinate and leverage their work within the team. Because face-to-face meetings were costly, and the project was distributed to save costs, neither systems analysts nor the project manager ever travelled and, therefore, lack of proximity of the remote team's activity caused coordination breakdown.

5 Discussion

5.1 Pitfalls in Remote Team Coordination

In this paper we try to understand, what is the effect of geographic distribution on how the work is coordinated in a distributed environment. The area of distributed software work is relatively new, and the majority of research conducted in this area is exploratory in nature. Contributing ro related research [26, 27, 2] our case study serves as an example, where distributed software teams rely on formal mechanisms (standardization), such as detailed architectural design and plans, to address impediments to team communication that result from geographical separation. However, in contradiction to a common view that the most effective way to manage global software teams is reliance on methodological standardization [2], the results of our case study indicate that coordination by program didn't work as intended, due to problems with disparities in work practices and limited feedback from the remote location.

Trying to improve the situation, direct supervision became the next dominating coordination mechanism. However, lack of proximity and trust, troubled the ability for direct supervision. Trying to control the remote team, the project manager from the central unit stumbled upon the problem of lacking trust and subsequently missing feedback again.

Our case also proves that coordination by mutual adjustment is troublesome because of the geographic distribution. Lacking trust, limited opportunities for face-to-face

contact and constant feedback in the distributed environment make it difficult to use mutual adjustment when coordinating work, which again hinders effective communication. This is one of the reasons for distributed organizations having problems to display the same cohesiveness, resilience, and endurance as a "physical" organization, and it is likely that a distributed organization will therefore experience a handicap that must be outweighed by other factors [18]. Frequent communication is important in distributed teams for providing constant confirmation that team members are still there and still working [28]. If feedback is provided on a regular basis, communication improves, which in turn leads to greater trust and improve team performance [29, 30]. Lacking mutual adjustment results in choosing either direct supervision or standardization as the dominating coordinating mechanisms.

5.2 The Effect of Coordination Mechanisms on Remote Team Performance

Exploring the characteristics of the effect of different coordination mechanisms on distributed team performance, we have gathered the observations that indicate that lacking balance in selecting coordination mechanisms not only leads to coordination pitfalls, but also affects the remote team performance.

Relying on standardization as the primary coordinating mechanisms one should be sure that there are no disparities in work practices, since it is difficult to discover such disparities using standardization. Moreover, disparities in work practices can lead to troubled understanding of the manager's decisions, misunderstanding of requirements and misbehaviour during implementation and testing [5].

The effect of direct supervision, can have devastating effects (such as decreased team performance or a total project breakdown) when people do not trust each other. Trust is a premise for getting necessary feedback, which is needed for direct supervision to be successful [5].

By not fulfilling the most basic needs [25], e.g. existence and relatedness, or the most basic hygiene factors [31], e.g. working conditions, it is not likely that the employees will enjoy their work and therefore perform accordingly (a paycheck is not enough [32]). Additionally, a manager must always adapt the leadership style no matter what type of coordinating mechanisms might come into play. Employing a task-oriented and authoritarian style [3], when it is not appropriate, will seriously influence work throughput in most teams.

In the case study described in this paper the manager would most likely have benefited by applying situational leadership [33],[34]. The manager, who tried to empower and delegate reponsibility to the team [35], ended up with a role where he was defining and instruct the team what to do, and then making it impossible to empower the team Furthermore, the attempts of the remote team to cope with the complexity of the given tasks by applying mutual adjustment on their site were not appreciated.

5.3 Implications for Practitioners

With our case study we exemplify the behaviour of the project manager who didn't trust the remote team and the effect of his coordination style on the remote team performance and psychological comfort. We therefore recommend:

Adjust Your Coordination Mechanisms: Consider the nature and perceived complexity of the tasks (both technical and business related) given to the distributed team and adjust the coordinating mechanisms taking this into account. Since the remote team may have problems understanding the tasks it is important to use a mechanism that encourage frequent communication and feedback.

Furthermore, estimate the level of organizational diversity and disparities in work practices before relying on standardization. In addition, managers should avoid the kind of rigid control that impairs creativity and spontaneity [21]. Therefore direct supervision must be used with care. A premise for being able to adjust the coordinating mechanism is frequent feedback and trust. This can be achieved by using the measures suggested by Moe and Smite [5].

Make it Possible to Apply Mutual Adjustment When Needed: A software organization often deploys experts in multi-disciplinary teams that carry out projects in a complex and dynamic environment. Such organizations can be classified as innovative, where mutual adjustment is the most important coordinating mechanism [6]. The managers should avoid rigid control (direct supervision), which impairs creativity and spontaneity [21]. Mutual adjustment is also important when solving complex tasks. Therefore there is a need for mutual adjustment in any software development project. Relying too much on standardization and direct supervision, together with having problems achieving trust will make it difficult to apply mutual adjustment.

Avoid Complex Task Distribution: Carmel claimes that the cost of coordinating work increases when either the tasks are new or uncertain, or when the work units become more interdependent [2]. Distribution of interdependent and tasks perceived as complex exacerbates the problems related even more.

In addition, making the remote team responsible for entire modules is important, i.e. from planning to testing, the team gets a deeper understanding of the tasks it is working on. Training and investment in a long time relationship is also important. Extending the existing studies on globally distributed software development (GSD) that recommend to 'trim down GSD towards nearshoring' [8], we emphasize that distributing software development within national and organizational borders still, in some ways, faced the same difficulties as those faced in the GSD context. Therefore, we recommend evaluating all pros and cons before starting a distributed project and avoiding pure cost reduction deals.

Keep the Team Small: Given that mutual adjustment in its pure form requires everyone to communicate with everyone, the team or network needs to be compact [18].

5.4 Implications for Future Research

While this case study illustrates the effect of coordination by standardization and direct supervision, our future work will focus on exploring the effect of coordination by mutual adjustment. Holmström et al. state [4]: despite the fact that the more common view is that agile methods are not applicable for GSD, agile practices may assist in alleviating some of the distribution problems. We would therefore be interested in exploring distributed software teams that mainly rely on mutual adjustment while performing tasks of different complexity. Future research should also study how work

is coordianted when there exist a more mature relationship between the remote and local team.

Along with the research in the area of global software development, we also emphasize the importance of empirical research in the intra-organizational intra-national context to supplement the understanding of internal domestic supply chains.

6 Conclusions

While organizations become more distributed, software development tasks of different complexity are often performed by distributed software teams. Team coordination necessary to make sure that it contributes to the overall objective faces new challenges with respect to geographic distribution of project managers and their teams. The overhead of control and coordination associated with any software project is astounding [2]. It is therefore important to understand the pitfalls of team coordination in order to make a distributed project successful.

We have found that:

- Failure applying standardization may result in direct supervision becoming the dominating coordination mechanism, but this only increases the problems related to decreased communication and missing feedback.
- Trust and a common understanding of the work processes is a premise for succeeding with standardization, direct supervision and mutual adjustment.
- Application of frequent communication and feedback through mutual adjustment is essential in overcoming the complex tasks.

Although theory suggests that different coordination mechanisms shall dominate in different situations dependent on task complexities [6], geographic distribution introduces certain impediments in applying each of the mechanisms discussed in this paper. In particular, disparities in work practices put under threat the path of standardization, lack of proximity troubles direct supervision and distribution makes it difficult to apply mutual adjustment, which is important for complex tasks. Thus, coordinating mechanisms shall be chosen thoroughly and in balance.

Acknowledgments. This research is supported by the Research Council of Norway under Grant 181658/I30, the Knowledge Foundation in Sweden under a research grant for the project *BESQ*, European Social Fund under grant "Doctoral student research and post doctoral research support for university of Latvia" and the Latvian Council of Science within project Nr. 02.2002.

References

1. Herbsleb, J.D., Mockus, A.: An Empirical Study of Speed and Communication in Globally Distributed Software Development. IEEE Transactions on Software Engineering 29(3), 481–494 (2003)
2. Carmel, E.: Global software teams: collaborating across borders and time zones. Prentice-Hall, Englewood Cliffs (1999)

3. Tannenbaum, R., Schmidt, W.H.: How to Choose a Leadership Pattern. Harvard Business Review 51, 162–174 (1973)
4. Holmström, H., Fitzgerald, B., Ågerfalk, P.J., Conchúir, E.Ó.: Agile Practices Reduce Distance in Global Software Development. Information Systems Management 23(3), 7–18 (2006)
5. Moe, N.B., Šmite, D.: Understanding a Lack of Trust in Global Software Teams: A Multiple-Case Study. In: Software Process Improvement and Practice. John Wiley & Sons (in press, 2008)
6. Mintzberg, H.: Mintzberg on Management: Inside Our Strange World of Organizations. Free Press, New York (1989)
7. Cataldo, M., Bass, M., Herbsleb, J.D., Bass, L.: On Coordination Mechanisms in Global Software Development. In: Proceedings of the International Conference on Global Software Engineering (ICGSE 2007), pp. 71–80. IEEE Computer Society Press, Munich, Germany (2007)
8. Holmström, H., Conchúir, E.Ó., Ågerfalk, P.J., Fitzgerald, B.: Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance. In: Proceedings of the International Conference on Global Software Engineering, pp. 3–11. IEEE Computer Society Press, Costão do Santinho, Florianópolis, Brazil (2006)
9. Crowston, K., Annabi, H., Howison, J., Masango, C.: Effective Work Practices for Software Engineering: Free/libre Open Source Software Development. In: Proceedings of the 2004 ACM Workshop on Interdisciplinary Software Engineering Research, pp. 18–26. ACM Press, Newport Beach (2004)
10. Herbsleb, J.D., Paulish, D.J., Bass, M.: Global Software Development at Siemens: Experience from Nine Projects. In: Proceedings of the 27th International Conference on Software Engineering (ICSE 2005), pp. 524–533. ACM Press, St. Louis (2005)
11. Boden, A., Nett, B., Wulf, V.: Coordination Practices in Distributed Software Development of Small Enterprises. In: Proceedings of the International Conference on Global Software Engineering (ICGSE 2007), pp. 235–246. IEEE Computer Society Press, Munich, Germany (2007)
12. Fonseca, S.B., Souza, C.R.B.d., Redmiles, D.F.: Exploring the Relationship between Dependencies and Coordination to Support Global Software Development Projects. In: Proceedings of the International Conference on Global Software Engineering (ICGSE 2006), p. 243. IEEE Computer Society, Costão do Santinho, Florianópolis, Brazil (2006)
13. Wiredu, G.O.: A Framework for the Analysis of Coordination in Global Software Development. In: Proceedings of the 2006 International Workshop on Global Software Development for the Practitioner at International Conference on Software Engineering, pp. 38–44. ACM Press, Shanghai, China (2006)
14. Herbsleb, J.D., Mockus, A.: Formulation and Preliminary Test of an Empirical Theory of Coordination in Software Engineering. In: Proceedings of ACM Symposium on the Foundations of Software Engineering (FSE), Helsinki, Finland, pp. 112–121 (2003)
15. Myers, M.D., Newman, M.: The Qualitative Interview in IS Research: Examining the Craft. Information and Organization 17(1), 2–26 (2007)
16. Dingsøyr, T.: Postmortem Reviews: Purpose and Approaches in Software Engineering. Information and Software Technology 47(5), 293–303 (2005)
17. Prikladnicki, R., Audy, J.L.N., Damian, D., de Oliveira, T.C.: Distributed Software Development: Practices and Challenges in Different Business Strategies of Offshoring and Onshoring. In: Proceedings of the International Conference on Global Software Engineering (ICGSE 2007), pp. 262–274. IEEE Computer Society Press, Munich, Germany (2007)

18. Groth, L.: Future Organizational Design: The Scope for the IT-based Enterprise. John Wiley & Sons, New York (1999)
19. Dybå, T.: Improvisation in Small Software Organizations. IEEE Software 17(5), 82–87 (2000)
20. Nerur, S., Balijepally, V.: Theoretical reflections on agile development methodologies - The traditional goal of optimization and control is making way for learning and innovation. Communications of the ACM 50, 79–83 (2007)
21. Takeuchi, H., Nonaka, I.: The New New Product Development Game. Harvard Business Review 64, 137–146 (1986)
22. McGregor, D.: The Human Side of Enterprise. McGraw Hill, New York (1960)
23. Dirks, K.T., Ferrin, D.L.: The role of trust in organizational settings. Organization Science 12, 450–467 (2001)
24. Salas, E., Sims, D.E., Burke, C.S.: Is there a big five in teamwork? Small Group Research 36, 555–599 (2005)
25. Alderfer, C.P.: An Empirical Test of a New Theory of Human Needs. Organizational Behavior & Human Performance 4, 142–176 (1969)
26. Ramesh, B., Cao, L., Mohan, K., Xu, P.: Can distributed software development be agile? Communications of the ACM 49, 41–46 (2006)
27. Ågerfalk, P.J., Fitzgerald, B.: Flexible and distributed software processes: Old petunias in new bowls? Communications of the ACM 49, 26–34 (2006)
28. Jarvenpaa, S.L., Shaw, T.R., Staples, D.S.: Toward contextualized theories of trust: The role of trust in global virtual teams. Information Systems Research 15, 250–267 (2004)
29. Jarvenpaa, S.L., Knoll, K., Leidner, D.E.: Is anybody out there? Antecedents of trust in global virtual teams. Journal of Management Information Systems 14, 29–64 (1998)
30. Jarvenpaa, S.L., Leidner, D.E.: Communication and trust in global virtual teams. Organization Science 10, 791–815 (1999)
31. Tagiuri, R.: Managing people: Ten essential behaviors. Harvard Business Review 73, 10–10 (1995)
32. Whyte, W.F.: Money and Motivation. Harper & Row, New York (1955)
33. Hersey, P., Blanchard, K.H.: Life Cycle Theory of Leadership. Training and Development Journal 33, 94–94 (1979)
34. Hersey, P., Blanchard, K.H.: Great ideas revisited: Revisiting the life-cycle theory of leadership. Training & Development 50, 42–48 (1996)
35. Hersey, P., Blanchard, K.H., Johnson, D.E.: Management of Organizational Behavior: Leading Human Resources. Prentice Hall, New Yersey (2001)