Low Degree of Separation Does Not Guarantee Easy Coordination

Zane Galviņa [1], Darja Šmite [1,2]

[1] University of Latvia / Riga, Latvia /{Name.Surname}@lu.lv

[2] Blekinge Institute of Technology / Karlskrona, Sweden / {Name.Surname}@bth.se

***Abstract*— In the times of increased global competition, software companies are forced to search for more effective development practices and often team up with onshore and offshore partners to develop faster and better products. In this paper we empirically explore a highly distributed onshore development project with a complex coordination structure. Our findings demonstrate that onshore development projects are not protected from coordination and communication challenges and task allocation complexities. Previously reported qualitative findings regarding organizational problems in this paper are supplemented with quantitative measurements of the true coordination delays and additional analysis of coordination patterns and their evolution.**

Keywords – Distributed development, Software development process, Onshore outsourcing, Coordination delay.

I. INTRODUCTION

A significant amount of papers have been written about the problems within globally distributed development. Among the most frequently mentioned issues are temporal distance, cultural differences, and resulting challenges with respect to communication, coordination, and control [1-3]. Although collaborations within the same country (onshore) may be viewed as a solution to many of these problems, it is false to assume, that all challenges attributed to distribute development are automatically avoided. This is mainly because one of the major challenges mentioned in relation to globally distributed development is coordination [1, 2]. Coordination in this context means - integration or linking together of different parts of an organization to accomplish a collective set of tasks [2]. Unfortunately, coordination concerns are very common in onshore development as well. Practical experiences suggest that even if all team members are situated in one country but distributed in different premises coordination problems emerge, since communication challenges occur from 30 m of separation [4].

In this paper we focus on computer-mediated coordination and present findings from studying a highly distributed onshore software development project. The aim of our study is to explore efficiency of relying on computer-mediated coordination and measure delays associated with different coordination patterns.

II. BACKGROUND AND MOTIVATION

Development teams that coordinate their tasks successfully exhibit better performance [5]. From a wide range of coordination theories, in this paper we rely on the modern model of coordination. In contrast to traditional model, which defines coordination as the act of coordinating activities toward a common goal, the modern model sees coordination as the act of coordinating dependencies between activities toward a common goal [6]. Because traditional model is more commonly used in the literature, the vast majority of research does not pay deserved attention to studying dependencies. However, dependencies in distributed development are highly important. While theoretically independent task allocation to individual developers or collocated team members is preferable, modularization of work is not always easy to achieve [7]. The problem of task allocation strategies and support of product-level dependencies has been studied by Conway [8], who suggested that software product structure inevitably reflects the organizational structure. For distributed development this means that remote teams that work on interrelated tasks shall be linked. However, in practice, geographically distributed development teams are at a disadvantage because of the negative impact of distance on coordination of work among software developers [5, 9]. As a consequence, coordination breakdown is one of the common problems in distributed projects, expressed in e.g. coordination delays [10, 11]. In comparison with collocated development, coordination time in distributed environment can become up to 2.5 times longer [10].

In this study we focus on measuring coordination delays associated with distributed work in an offshore outsourcing project. In contrast to existing research, we explore a highly distributed industrial project with four organizations working on completion of the joint goals. Our research is driven by the following research questions:

RQ1: How unclear organizational structure affects task flows in a highly distributed software project?

RQ2: What is the mean time of cross-organizational task coordination delays?

III. RESEARCH METHODOLOGY

This paper presents a single-case study [13] where participant observation method [14] was employed. In our case participant observation method consists of two types of observers. The leading author too the role of an active participant [15], who was working as a system analyst in the project studied. The second author took the role of a privileged observer [15].

In order to answer the research questions, our observations were focused on coordination and task allocation processes. The scope of our investigation was limited to studying only one of the two sub-systems developed in the project (more details in Section IV).

All tasks in the project were assigned via project tracking tool Jira [16] or by email. Thus we collected all task assignments from Jira, which included problem reports, tasks and customer reports regarding one sub-system as well as emails that contained references to working tasks. Only tasks that required programming were considered.

At the moment of investigation there were 1025 task assignments from Jira, and 111 task assignments were related to the studied sub-system. In addition 38 emails with one or more task assignments were collected for analysis.

Due to limited data availability the measurement of coordination paths focused on analyzing time between the assignment date of the task and the date when it reached the end respondent. It is worth noting that we analyzed calendar days without distinguishing weekends or holidays. We believe that in this case it was not important since two of the studied organizations practiced work over weekends.

IV. PROJECT OVERVIEW

The aim of the project was to move to a new platform in order to follow new political regulations that restricted the need of integration of many systems together.. Secondly there was a need to add a new functionality.

There were four organizations involved in the development of the project and all of them were situated in the same city but in different premises. One of those organizations was leading the project, which we further refer to as prime contractor (D1). This organization contracted work to two external organizations - sub-contractors (D2, D3), at the beginning of the project. After a while one of those organizations (D3) involved an additional organization (D4) in the project, but outsourcing relationship was hidden from the prime contractor. Therefore it led to a situation where official structure of a development team (set by D1) differed from the real one.

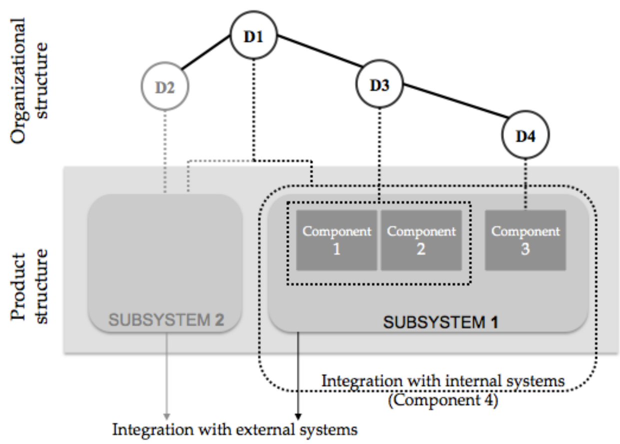

Figure 1 High-level structure of task allocation regarding the product and corresponding organizations

The whole system consisted of two separate sub-systems (see Fig. 1). In this paper we focus our analysis on sub-system 1 and aim at understanding how the distribution of work affected the time necessary for coordination of tasks among the three participating organizations.

The project started in the beginning of 2011 and was planned to finish by January 2012. Nevertheless the project didn’t reach this goal specifically because of delays in completion of sub-system 1. To coordinate the work between all developers from the three organizations, the Jira tool was used. However, since organization D4 presents a role of a “hidden” organization, they were not included in this process. Task assignments to D4 were allocated through email, and coordinated primarily by D3. More detailed description of the project is reported in [17].

V. RESULTS

Issues associated with unclear organizational structure surfaced in our previous research [17] and are analyzed further in this paper. During the project we observed the way coordination of tasks was organized. We realized that because of unclear responsibilities and hidden organizational relationships coordination flows took much more time than expected and implied by the initial coordination structure. In the beginning we expected that tasks that concerned the hidden organization would inevitably be ineffective. More detailed investigation showed that coordination flows changed over the course of the project. Thus we will present the changes of coordination flows, as well as the median time for them.

A. Coordination flows

Several task coordination patterns emerged from studying different flows in different stages of the project (see Fig. 2).

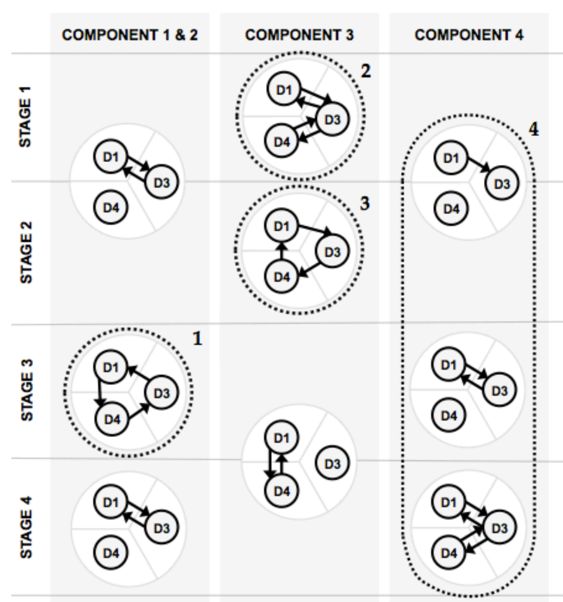

Figure 2 Task coordination flows of Subproject 1 (problematic flows are numbered and highlighted by dotted lines)

Stages in Figure 2 represent a part of project timeline. In reality each stage has different length and represents a pattern of coordination of tasks at a specific point of time.

Stage 1: Since the official organizational structure consisted of only three organizations at the beginning of the project, parts of the work that was assigned by D1 to D3 was further forwarded to D4. At this stage D4 didn't have an account in Jira and the tasks were sent via email.

Stage 2: This stage we distinguished from the moment when D4 gained access to Jira. During this time D4 employees were acknowledged as official participants. We expected this to improve the situation, however the roles and responsibilities, and more importantly the differences between D3 and D4 were not yet clear and thus complicated the situation for some cases even more.

Stage 3: During stage 3, organization D1 started to reassign delayed tasks from D3 to D4. We assume that D1 realized that many tasks were shifted to another person by D3, thus they assumed that organization D3 has changed the responsible person in order to receive tasks. It is worth mentioning that this decision was made by D1 based on assumptions only. Finally it led to a situation when those coordination flows that were previously clear were damaged (see correct flows on Fig.4).

The time of this stage is very close to the planned delivery time of the project when the last iteration of the work began. Interestingly, at this stage D3 realized that many of the tasks that they received were falsely assigned, and 16 assignments were sent back to D1.

Stage 4: This stage emerged after the officially planned project deadline. As we can see the coordination flows that were created at this stage do not match with initial component allocation as outlined in Fig. 1.

B. *Coordination delays*

Analyzing task coordination flows, we can see that the evolution of coordination patterns changes dramatically and sometimes contains redundant flows, cycles or excludes the necessary flows. Thus it becomes obvious that there should be coordination delays. In order to understand how much time it takes to deliver tasks to the correct end recipient, we have calculated the median time value for each type of task flows. Delays are measured in calendar days (including weekends and holidays, if any). The following results were calculated for each flow (flows numbered in Fig. 2).

Flow 1		Flow 2		Flow 3		Flow 4	
Min	15d 20h	Min	3 h	Min	0h	Min	138d 16h
Max	138d 2h	Max	104 d 20 h	Max	74d 21h	Max	184d 20h
Median	40d 22h	Median	18 d 7 h	Median	18h	Median	138d 16h

Figure 3 Coordination delays of problematic flows

Component 1&2: One of the organizations, D3, was developing component 1 and 2 as it was planned in the beginning of the project. Figure 2 at the first stages shows an effective way of work coordination where D1 allocated tasks directly to D3. At these stages there are no coordination delays.

Unfortunately, by the time when D1 started to assign tasks to D4, the lack of clarity between D3 and D4 organizational boundaries led to coordination breakdown. In total, we identified four incorrectly assigned tasks, which caused significant delay. The loss reached the median coordination time of about 40 days (see Fig. 3).

Component 3: Two coordination flows with significant delays emerged when studying task coordination in component 3. We suggest that the delays were caused by the unavailability of Jira for D4. Thus tasks assigned through Jira had to be tunneled through email. The mediation of tasks inevitably resulted in delay. We analyzed 35 task assignments. The maximum delay value reached 74 days 21 hours.

As noted earlier, eventually D4 received access to Jira and coordination flow changed. Although the organizational changes were insignificant, most of the delays were substantially reduced. D1 still kept assigning the tasks that were meant for D4 to D3. The difference was that D3 changed the responsible person in Jira and thus did not have to mediate D4 tasks through emails anymore. After gaining accessing to Jira, we have still identified 16 assignments that D1 misallocated to D3 instead of D4. Minimal time for D3 to change the responsible person to one of the D4 employees took 3 h, but the maximum 104 days 20 h.

Component 4: In retrospect, this component can be considered as the "hardest" part of the project. We analyzed 16 tasks during the course of the project. Interestingly, these tasks first were assigned to D3, then were sent back to D1 and then mediated to D4 through D3. This was mainly because of unclearly communicated and misunderstood responsibilities for this component. In result, the minimal coordination delay time was 138 days 16 h, more than 4 months (see Fig.3). It is worth noting that in the beginning of the project, when the system was divided into components, it was planned that every organization will perform requirements engineering, implementation and delivery of work on assigned components. Strangely, D1 completed requirements for component 4, at the same time assuming that the development work will be done by D3, which neither felt responsible nor had the capacity or expertise to complete this task. At the end it was decided to forward the work to D4.

VI. Discussion and Conclusion

In this paper, we presented results from a case study project that failed to meet the deadline due to multiple coordination problems. We studied project events in the light of coordination issues and identified several areas of concern. Results presented above confirm that even the low degree of separation in a project may lead to important coordination issues. In our case involvement of developers from four different organizations turned out to be too complicated to understand responsibilities. We observed that during the course of the project participants discussed who has to do what and often argued about the distribution of responsibilities. In such cases, the legal documents were consulted, and several conflicts occurred as in the work on component 4 for which nobody felt responsible.

If we look at the architecture of the product the desired communication and coordination structure seems evident. Expected coordination flows are demonstrated in Fig. 4. The results showed that some components reached effective coordination flow at the end of the project, but component 4 was coordinated with the dramatic delays all the time.

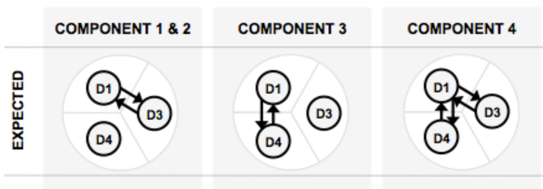

Figure 4 Expected coordination flows

Variable coordination flows and huge delays in the studied project led to missed deadlines. We attribute the main reasons for this to 1) Unclear organizational structure, and 2) Unclear responsibilities and thus inability to effectively coordinate the tasks to the correct recipient. We argue that addressing these two problems would be enough to save the project.

For the future projects we recommend to first recognize the formal and informal structure of the project organization and communicate it to all parties involved. It is also essential to clarify responsibilities for systems components or set of tasks in advance. We also note that some of the tasks were “lost” in the system. Advanced tools or effective communication shall ideally support computer-mediated coordination.

The key conclusions and lessons learned can be summarized as follow. In relation to RQ1 we learned that a hidden organizational relationship resulted in tunneled coordination and broke the initial plan for task allocation. Among the misallocated tasks we have identified tasks that were sent around, and tasks that were simply mediated. Both types of coordination challenges in some cases resulted in significant delays. Inability to coordinate some of these tasks finally caused the project failure to meet the deadline.

In order to answer RQ2 we calculated the median of each type of coordination flow. Results are dramatic. We observed that developers approached tasks one by one, and misallocated cases were identified with a huge delay. Certain tasks piled up for 2-3 month and only then were forwarded to the correct recipient.

Finally we conclude that clear organizational structure and clear responsibilities are the major success factors for distributed projects even with low separation. We argue that the low degree of separation could be also challenging since organizations do not have the feeling that the project structure is complicated as in global projects.

ACKNOWLEDGMENT

We thank our industrial partners for the opportunity of following the project. This work has been supported by European Social Found project No. 2009/0216/1DP/1.1.1.2.0/09/APIA/VIAA/044, as well as BESQ+ (20100311) grant from the Knowledge Foundation in Sweden.

REFERENCES

[1] S. Mohan, J. Fernandez, “ Distributed software development projects: work breakdown approaches to overcome key coordination challenges”. In *Proceedings of the 3rd India software engineering conference* (ISEC '10)., 2010, pp. 173-182.

[2] R. E. Kraut, L. A. Streeter, “Coordination in software development”. *Commun. ACM* 38, 3, 1995, pp. 69-81.

[3] V. Clerc, P. Lago, H. Vliet, "Global Software Development: Are Architectural Rules the Answer?," In *Proceedings of the IEEE international conference on Global Software Engineering* (ICGSE '07), 2007, pp. 225-234.

[4] T. Allen. Managing the flow of technology. Cambridge, 1977, Mass: MIT Press.

[5] M. E. Conway. How do committees invent. *Datamation*, Vol. 14, Nr. 4 (1968) , pp. 28-31.

[6] P. Ovaska, M. Rossi, P. Marttiin, “Architecture as a coordination tool in multi-site software development”. Software Process: Improvement and Practice, 2003, pp.233-247.

[7] D. L. Parnas, “On the criteria to be used in decomposing systems into modules”, Communications ACM, 15, 12, 1972, pp.1053–1058.

[8] M. Cataldo, M. Bass, J. D. Herbsleb, L. Bass, “On Coordination Mechanisms in Global Software Development”. In *Proceedings of the International Conference on Global Software Engineering* (ICGSE '07). IEEE Computer Society, 2007, pp.71-80.

[9] D. Šmite, N.B. Moe, R. Torkar “Pitfalls in Remote Team Coordination: Lessons Learned From a Case Study”, In proc. of PROFES 2008 int. conf., LNCS, July 2008, Italy, pp. 345-359

[10] J. D. Herbsleb, A. Mockus, T. A. Finholt, R. E. Grinter, “An Empirical Study of Global Software Development: Distance and Speed”. In: International Conference on Software Engineering, 2001, pp. 81-90.

[11] T. Wolf, T. Nguyen, D. Damian, “Does distance still matter?”. *Softw. Process* 13, 6 , 2008, pp. 493-510.

[12] G. M. Olson and J. S. Olson, “Distance matters”. *Hum.-Comput. Interact.* 15, 2, 2000, pp.139-178.

[13] R. K. Yin, “Case Study Research: Design and Methods” *Sage Publications*, 2009.

[14] N. Mack, C. Woodsong, K. M. MacQueen, G. Guest and E. Namey, “Participant Observation”, In *Qualitative Research Methods: A Data Collector's Field Guide*. Research Triangle Park, North Carolina: Family Health International, 2005, pp. 13-28.

[15] H. F. Wolcott, “Ethnographic research in education”. (C. F. Conrad, J. G. Haworth, & L. R. Lattuca, Eds.) *Complementary Methods for Research in Education*. American Educational Research Association. 1997.

[16] J. P. Rodríguez, C. Ebert and A. Vizcaino, “Technologies and Tools for Distributed Teams”. *IEEE Softw.* 27, 5, 2010, pp.10-14.

[17] D. Šmite, Z. Galviņa, “Socio-Technical Congruence Sabotaged by a Hidden Onshore Outsourcing Relationship: Lessons Learned from an Empirical Study”. In: proceedings of the Product-Focused Software Process Improvement Conference, Vol. 7343, 2012, pp. 190-202.