

Trust and Social Capital: Revisiting an Offshoring Failure Story of a Small German Software Company

Alexander Boden¹, Bernhard Nett¹, Volker Wulf^{1,2}

¹Institute for Information Systems and New Media, University of Siegen

²Fraunhofer Institute for Applied Information Technology, St. Augustin
{ *alexander.boden | bernhard.nett | volker.wulf* }@uni-siegen.de

Abstract. While work organization and social capital are known to be important factors for offshoring success, there is little empirical evidence on how these aspects evolve in the course of offshoring projects. In the literature, trust has been discussed as a personal disposition to abstain from control in a given situation, and was found to remain surprisingly stable in some cases. By analyzing the relation between control and trust in the course of a failed offshoring project, we want to add to the discussion on social capital as a factor for successful offshoring. The results of our long-term ethnographic study are somewhat paradox: in our case, ongoing conflicts motivated attempts to strengthen control, although personal trust and social capital remained strong. Despite the fact that the confidence of the partners in their offshoring project was weakened over time, the trust among the partners prevailed. However, social capital was not only unable to save the offshoring project—it also seemed to hinder the conflict resolution in some regards. Therefore, we argue that while social capital is an important factor, it should not be regarded as a context-free asset, but rather (in Bourdieu's perspective) as a risky investment.

Introduction

With ongoing globalization, offshore software development has become quite common. For instance, consulting agencies promote Global Software Engineering

(GSE) as a means to reduce costs and as a driver for process improvements in case activities are reengineered and streamlined as part of the move. However, while wage differences may offer options to reduce costs, the spatial, temporal and cultural issues in globally distributed cooperative work are still challenges and need to be better understood (Cataldo et al. 2006; Gutwin et al. 2004; Herbsleb et al. 2000).

Tackling these issues, it is often argued that GSE needs formalization of processes and a high level of social capital to be successful (Levina & Vaast 2008). Features such as formalization and social capital accumulation may—to a certain degree—be influenced when establishing the offshore cooperation. However, there has been little empirical evidence on this topic for later stages of offshore cooperation (King & Torkzadeth 2008). This is even more astonishing as these factors are likely to affect flexibility which is regarded as a major demand for software development in general (and especially for small enterprises). Therefore, we need to learn more about how the relationship among clients and vendors evolves within offshoring projects and which factors contribute to or oppose efficient cooperation (Fisher et al. 2008).

CSCW has a long tradition of researching problems of distributed cooperation. For example, CSCW studies have expounded the importance of awareness, tool appropriation, self-organization, behavior, interaction and communication in different kinds of work groups by means of ethnographic studies. However, there are very few in-depth studies which look at the particularities of cooperative work in off-shored software projects—specifically when small companies are involved.

In order to add to the understanding of offshoring, we conducted a long-term and in-depth ethnographic case study in a small German company between 2006 and 2008. During this time, the company developed software in an international team of German and Russian developers. In the end, the cooperation was terminated due to ongoing problems. By revisiting this failure case and the related conflicts over a longer period of time, we offer a complementary view compared to studies on best-practices. We investigate whether trust and social capital changed or remained stable over time in the offshoring project, and how these factors affected the offshoring relationship between the involved teams. In order to provide a detailed analysis, we investigated *articulation work* (Strauss 1988) conducted in the offshoring project.

The paper starts with a discussion of offshoring literature before it describes the research method applied in the ethnographic study. The description of the case is followed by a discussion under the perspective of articulation work. It turns out that trust and social capital indeed share some similarities, but are no guarantee for successful offshoring. Related findings are explained in the final chapter.

Offshore Cooperation in the Literature

Apart from challenges which are typical for any software development, GSE projects have to be conducted under particular organizational, cultural, spatial, temporal and legal conditions which can pose complex obstacles (Herbsleb et al. 2005; Ramesh et al. 2006). For example, temporal differences can lead to bottlenecks in regard to time for collaboration and coordination, while cultural differences can lead to mutual misunderstandings. As spatial distribution can harden or even constrain possibilities for control considerably, it often affects the necessary level of loyalty and trust among collaborators.

Hence, trust and social capital have been pointed out to be key factors for tackling challenges of distributed team cooperation (Hinds & McGrath 2006; Levina & Vaast 2008; Rottman & Lacity 2008). Trust has been characterized as a complex, multi-layered concept, which is—amongst others—related to expectations, experiences, and knowledge (e.g. is the trustee competent? Is his behavior predictable? Is he good-willing? Is he opportunistic?) (Imsland 2005). For our case, trust can be interpreted as a psychological state which allows for greater levels of self-organization, and for an abandoning of (available) control mechanisms (Zolin et al. 2004). In a similar fashion, social capital refers to network ties of “goodwill, mutual support, shared language, shared norms, social trust and a sense of mutual obligation that people can derive value from” (Huysman & Wulf 2004). As “social glue” holding together communities, social capital is expected to promote cooperative behavior in communities and organizations (Putnam 2000; Cohen & Prusak 2001).

According to this optimistic view, organizations (or teams/communities within organizations) with high levels of social capital will have a higher motivation to cooperate (Huysman & de Wit 2004). However, with regard to offshoring, social capital can be difficult to be fostered (cf. Cramton & Hinds 2007; Hinds & McGrath 2006): as teams are distributed spatially, face-to-face contacts are usually reduced to few limited timeframes. At the same time, relying on ICT for cooperation implies a higher risk of misunderstandings (Olson & Olson 2000; Billings & Watts 2007), especially in cross-cultural teams. For the same reasons, conflicts can be very difficult to handle, if they occur (Hinds & Mortensen 2005).

As a consequence, recent studies have stressed the importance of initial perceptions of trustworthiness for long-term relationships of international teams (Lee et al. 2008). Inter-personal trust, once established, was found to remain more or less stable in the course of distributed projects, at least in case of cross-functional teams (Zolin et al. 2004). However, it is still not clear if the same rule applies to homogeneous fields with uni-functional dyads, such as in software-offshoring projects, where developers usually should be able to assess the development work done by the other team more easily as compared to cross-functional teams.

By revisiting the failure case story of a small German software company, we want to analyze if social capital shares the detected self-preserving effects of inter-personal trust relation, or if it may at least benefit from them. As trust can only be understood within a particular context (Zolin et al. 2004), it is necessary to take a situated perspective in analyzing the actual work practices, the distribution of labor as well as the related formal regulations and conflicts in offshore relations. In this context, the case of a failure story can be interesting if it allows for the differentiation between continuities and discontinuities.

However, continuities within formal regulations neither guarantee their factual continuity, nor does changing regulations guarantee factual discontinuity, as in reality established patterns may prevail under new labels, or formal regulations may fail. Hence, it is the practical organization of collaboration, not merely the mental models of its organizers, which has to be taken into account. As we were interested in long-term dynamics of offshoring software development, a methodological focus was needed that covers all above-mentioned aspects. This made us adopt the concept of *articulation work*, as we will point out in the following section.

Methodology

The concept of articulation work was introduced by the sociologist Anselm Strauss for the analysis of interdependent actions of cooperating actors (Strauss 1988). Articulation work, similar to coordination, is needed to regulate the division of labor: it centers on decision-making regarding who is supposed to do what, when, where, how, with which quality, etc.? To a certain degree, everybody involved in collaborative work has to reflect not only about his/her work, but also about its organization. In this regard, articulation work is also related to trust and social capital, as it entails issues of trading formal control versus flexible self-organization (Boden et al. 2008).

Generally, coordination is seen as the organization of collaborative work. However, not everything which is necessary for collaboration is explicitly discussed and regulated as coordination, and often the organization of work is more complex than perceived by many actors. For instance, collaboration may need meetings or discussions between developers, the management, and the customers, but it may also include the administration of a program for a certain task (setting up a related infrastructure), fixing a broken server, or implementing a communication infrastructure for collective work organization— aspects which are seldom interpreted as coordination.

In this regard, the concept of articulation work aims at including *all* necessary (meta-)work to make work work. Hence, it offers a more holistic understanding of cooperative work than concepts of coordination: while the latter usually govern the distribution of tasks and responsibilities, articulation work includes formal and

informal coordination mechanisms (Schmidt & Simone 1996) as well as related meta-work, which the actors themselves are sometimes not even aware of (Star & Strauss 1999).

As a consequence, it might not always be clear what should be regarded as meta-work or coordination when it comes to particular efforts. What one actor sees as necessary in this regard does not need to be the same as the perception of another. The same is true for scientific observers, who are influenced by their perception of the case. In this regard, coordination may be understood as the explicit model resulting from self-organization, and meta-work as the related practical conclusions, both of them being dependent on cognition and practical interpretation. In contrast, articulation work is the amount of all related contributions, strategies and conflicts; it is the distributed *agency* of collaboration, not its result.

Articulation work takes the individual perceptions about coordination neither as per se correct descriptions of the distribution of labor, nor as pure illusions; instead, they are understood as necessary points of departures for related analyses. Explicit (coordination) and practical premises (meta-work) of collaboration are regarded as important challenges for the individual positioning of actors within an anticipated field of opportunities. By contrasting conceptions of collaborators with each other and by analyzing empirical evidence on collaboration practices and their outcomes, articulation work studies attempt to take the interests of the collaborators seriously by discussing them retrospectively against the background of all accessible knowledge about the collaboration and its impacts.

This kind of analysis allows the address of the differences between the lived (factual) and the planned, explicit organization (cf. Argyris et al. 1985). The latter is generally more “logical” (at least at first glance) than the lived organization, which in contrast generally responds to situated particularities in a more complex way. This duality of formalized and informal organization has been discussed within the CSCW community (cf. Suchman 1987) for a long time and led to a much broader understanding of cooperative work in this community when compared to hierarchical models of coordination (Schmidt & Bannon 1992; Faergemann et al. 2005).

In order to study articulation work, we did not only have to look at efforts of coordination and meta-work, but also had to analyze them by contrasting the anticipated logic of the process with what we observed as the factual one. Revisiting the history of a cooperative project over several years with our theoretic stances is difficult, and requires careful examination. Unfortunately, our access to the company was limited to particular timeframes, and we had to reconstruct (and interpret) parts of the case study by relying to narrative interviews with the involved actors. However, we tried to overcome the limitations of our approach by a triangulation of several ethnographic research

methods, comprising semi-structured and narrative interviews, participant observation as well as artifact analysis.

In order to understand the logic of offshoring strategies, we started by collecting related conceptions in the literature. Furthermore, we conducted a semi-structured interview with the German manager of the company in 2006 that centered on his general perception of offshoring, as well as on his particular offshoring strategy.

For the investigation of articulation work practices, we drew on participant observations which were conducted by visiting the German SME several times during 2007 and early 2008. The first two observations lasted one week each and focused on local and distributed software development practices in individual and collaborative work situations and tasks. We were also allowed to analyze artifacts such as emails, chat protocols, internal work papers and whiteboard sketches, and we conducted many informal interviews with developers and the German project manager during our stay. The findings were documented by means of field notes and photos, which were taken during the research.

Our analysis of the collected data was based on Strauss' and Corbin's Grounded Theory (1998). After each step, the transcripts of the material, both field notes as well as interview data, were scrutinized. Data was coded during a process that consisted of several stages. At first, we composed categories based on the findings in the collected data. Then these categories were related to each other and evolved during the further research. These categories were analyzed under the presented articulation-work perspective. First, we attempted to differentiate between formal work organization (taken from the interviews) and the factual work practices we had observed. Then, we tried to identify converging and different perceptions of the offshoring project, as well as reconstructing related interests on the basis of a careful examination of our data.

As a further step, we refined the results of our analysis by conducting extensive narrative interviews with the German project manager during a third on-site visit, as well as a Skype interview with the Russian team manager.

The Case Study

The offshore software development project we researched was conducted by a German SME. The company offers a standard software solution for process modeling as well as services in the field of business process management. Being part of a holding, the management and sales of the company were handled by an office in Bonn with seven employees. The holding had several other offices, for example in Hamburg (data processing) and Düsseldorf (holding-management). Additionally, about 200 business consultants worked as freelancers in close cooperation with the company.

The software development we studied was carried out by an office in Berlin with seven employees. Apart from the development, the team in Berlin was accountable for the customer support as well as the management of the offshore cooperation with the Russian partner company. This cooperation had been started in 2002, when the German software office (at that time not yet integrated into the holding) had decided to found an offshore branch office in Saint Petersburg in order to reduce development costs.

The decision to locate the branch to Russia was based on a personal friendship of the German entrepreneur with a Russian developer who, according to the German manager, was trusted to be a competent and loyal team manager. This developer had been employed and ordered to hire three further developers in Saint Petersburg. The whole team was invited to Germany in order to become acquainted with the code base of the company. The team was able to take on the leading role of software development after a couple of months.

The German team manager described how the development of a formal model of work distribution marked the beginning of the cooperation. This model defined different roles and tasks for the teams. It included the role of the (German) project manager, the (Russian) team manager, the (Russian) software developers and the (German) testers. Thus, the German project manager wanted to oversee the development of the offshore team directly. In disciplinary or legal matters, the local team manager could be involved.

The German team concentrated on quality assurance, which involved the helpdesk for customers, the testing of the developed code and the strategic planning of upcoming versions. Thus, the definition of new features in terms of specifications and the description and classification of newly discovered bugs were under the responsibility of the German project manager and his team, while the offshore branch was responsible for the execution of the development. In the daily work, the results of the tests, descriptions of new features customers had asked for or bugs that were encountered by the helpdesk team and similar information would be communicated to the offshore branch for investigation. This usually involved personal visits of the German project manager to the offshore site shortly before new releases. During these visits, the German project manager helped handle the bugs (usually discovered in the last minute) and discussed the features of the following release with the Russians. The Russians in turn were to document their progresses in terms of monthly reports and review their code on a regular basis for quality assurance.

In the cooperation, members of both teams relied on several tools, which included a shared code repository (SourceSafe, situated in Germany) and IDEs/compilers (for C/C++, Java and Visual Basic), a bug tracking system as well as a product and development database based on Lotus Notes. For daily communication, a Lotus Notes plug-in called "Sametime" provided instant-messaging and screen-sharing functionality. Sametime allowed for the integration into the

Tool	Type	Used for
SourceSafe	CVS	Managing source-code
Product database	Lotus Notes database	Administrating specifications and releases; tracking progress of work
Development database	Lotus Notes database	Sharing templates for specifications, bug reports and formal work conventions
Sametime	Lotus Notes plug-in	Communicating via Instant Messages; sharing screens
SQA	Bug tracking system	Administrating and tracking bugs
Borland / Eclipse	IDE	Working on code

Table I. Tools provided for cooperation.

Lotus Notes environment and for encrypted communication and recording of screen-sharing sessions for later reviews (see table I).

Changes to the Division of Labor

According to the German team manager, the quick growth of the offshore team soon required certain adjustments of the formal division of labor. As he explained, it had become increasingly difficult to specify new features quickly enough to keep the growing offshore team busy—especially, when the number of Russian developers had exceeded the size of the German team. As the German project manager put it: “One day of development required one day of writing specifications” (Field notes, March 11, 2008).

As it became harder and harder for the German team to keep up with their work, the decision was made to change the formal division of labor. The Russian developers were now to write the specifications themselves, which were then in turn checked by the German team. According to a German team member, this decision was also based on the high competency of the Russian developers, who were trusted to have a deep understanding of the technical feasibility since they were in charge of the development. This change allowed the German team to reduce its work significantly and to enable the further growth of the offshore team which soon reached a size of up to 15 developers, as the German project manager reported.

However, delegating the requirement-engineering tasks to the offshore team led to significant problems, as the Russians lacked the necessary context knowledge: “The required information is very detailed: what does the user

interface look like, what conflicts may prevail with other features and what are special cases etc.” (Field notes, March 11, 2008). As the project manager pointed out:

“Knowledge concerning the practical usage and the technical background has to be combined in a creative way in order to find a solution. There is a difference between requirements specifications [considering the context-of-use] and design specifications, [being limited to the technical background]. The Russians tended to produce the latter” (Field notes, March 12, 2008).

According to the project manager, the problem was exacerbated by the Russian team’s poor English skills. While only rudimentary English skills would be needed for the coordination and control of already defined tasks, the definition of new features or the transfer of context information would be much more complex, thus sometimes exceeding the skills of the Russian colleagues. The German project manager explained: “The chats took much time and it was very difficult to transfer the related knowledge. It is easy to assign tasks or take over results, but it is hard to explain what needs to be done” (Field notes, March 11, 2008).

On the other hand, the Russians also reported problems concerning this way of cooperation:

“People [from the German team] (...) had no time to review them [the specifications], so the developers started to work without acceptance of specifications. (...) [So the] specifications did not follow the real implementation, or it took too much time for writing specifications” (Interview, May 28, 2008).

Attempts of Standardization

Faced with severe problems of communication and knowledge transfer, the company introduced a higher level of standardization to their documentation. Thus, standardized forms for documents, conventions for bug descriptions, source code comments and specific languages were developed. By providing examples and checklists, seen as help for the Russian developers with their tasks, the company expected to reduce the amount of communication and to ensure the quality of the produced documentation. The related documents were stored in the development database.

However, the complexity of writing specifications in combination with the missing background knowledge still made the tasks difficult and inconvenient for the Russian team, as the German project manager explained: “[The Russians] lacked the understanding of the program and the context of its use and the work is very unattractive, as it is very challenging and not well supported by tools” (Field notes, March 12, 2008). In addition, the German team reported increasing difficulties with the offshore developers, who started to ignore tasks that were

recognized by both teams as being unpleasant and annoying. This mainly included the writing of documentation and specifications as well as the tracking of the work with log-files.

“The Germans introduced forms to the product database in which the Russians should have entered their tasks with the expected beginning and end. They did this, but only at the beginning of the planning stage. As everything is very complex and unexpected dependencies occur, it is impossible to anticipate everything. Thus data needs to be updated regularly, but the Russians did not do so” (Field notes, April 12, 2008).

The following excerpt of a conversation illustrates this problem. The dialogue was taken from the chat-log of an online meeting between the German project manager and one of the Russian developers. The initiator of the online-meeting was the project manager who wanted an overview of the developer’s tasks. Using Sametime, the project manager was able to take control of the mouse and screen of the Russian developer and test the newly implemented features in this way. The inspection was accompanied by a chat discussion and took nearly three hours. The subjects of the discussion were the tasks (mainly feature specifications) in the product database, which were worked off feature by feature:

”Project manager I know we spoke MANY times about it... (...)
 it is impossible for me to follow progress if you don't write comments!
 so please don't let me repeat it again :-((...)
Developer I don't understand what should I write here, the implementation is
 fulfilled in 100 %
Project manager let me show you how i do it in my tasks.

On the shared screen, he [the German project manager] shows Dmitry some comments he has written. He opens one of his tasks, where he has already noted his progress like in a diary. The comments hint at problems he encountered, and at discussions with the developers. Then, he opens the product database and starts to comment on another task:

- started implementation
- bss [abbrev. name of another developer] send me new idea, so i stopped implementation. see info above.

Project manager you can decide the details in the comments.
 you should however add info that may be useful to you and to other
 people.
 this may help you keep notes on tasks (instead of using paper :-))
 or for example when you stop a task or need to restart it after some
 time.... you can use this to remember what you have to do.
 in any case whenever you update STATUS, PROGRESS or DATES....
 then you should add a comment regarding the reasons of the update.
Developer o.k.”

(Field notes, July 11, 2007).

Similar discussions concerned the conduction of internal code reviews of the offshore team as well as other examples of missing documentation.

Selling the Offshore Organization

According to the project manager, the problems with motivation were exacerbated when the decision was taken to sell the branch office to the Russian team manager in 2004. This decision was related to ongoing problems with the cooperation, as the Russian team manager reported:

“When we started [we were] four people (...). [In] 2004, all of these four developers left the company, because they were not satisfied with the situation. And from my side I wasn't able to do anything, to keep them (...). Because I had always to discuss any small question with Berlin” (Interview, May 29, 2008).

Furthermore, the decision was related to the challenges of handling the complex legal and organizational requirements of running an offshore branch. The communication with the local authorities turned out to be a serious and permanent challenge for the small German company, having no previous experience with Russian law. Hence, by changing the status of the Russian partner to that of an independent company, the German entrepreneur hoped to avoid many of the legal problems of managing an international company.

Thus, in 2004 the decision was made to continue the work by means of a contract between the SME and a now legally independent Russian company. As the Russian team manager explained, the Russians were quite happy with this change: “After we started to work as an independent company it got much easier for me to take decisions (...). And before, it took long discussions with Germany about why it was required” (Interview, May 29, 2008).

However, according to the German project manager, this change had dramatic consequences for the international cooperation. The Russian team manager, now being the proprietor instead of the employee, started to expand his company and look for new customers in order to reduce his dependence on the German SME.

In the interview, the German team manager described this strategy as expectable and even understandable. However, there were also unforeseen consequences, as it became much more difficult to continue the cooperation when the Russian team manager increasingly reduced his commitment to the cooperation. As finding new business partners became the main goal of the Russian partner, the German developers again were unable to control the Russian developers, who (from the perspective of the Germans) lacked discipline.

Even worse for them, according to the German project manager, the Russian team manager had been the most experienced and trusted team member abroad (especially since so many others had left the company), and his change of interest

led to severe problems, as the other Russian developers were unable to perform his duties with the same professional standard:

“[The German project manager] was unhappy that [the Russian team manager] was not available as a developer anymore from one day to another. As he was the manager instead of the developer now, the relationship had changed: instead of giving orders, everything was subject to negotiation” (Field notes, April 12, 2008).

In this regard, the dependency on the Russian developers made it difficult for the German team to enforce a reasonable accomplishment of tasks (especially of inconvenient ones) by the Russian team. “[The Russian developer] agreed to change his behavior, but he did not do it. And the Germans apparently were unable to convince him” (Field notes, April 12, 2008).

Salaries and Infrastructure

The problems with the offshore developers hit the company at a disadvantageous point of time. In 2006, the German company had been taken over by a holding. At the same time, according to the German project manager, the development costs had almost tripled compared to the situation in 2002. As both sides reported, the level of the salaries was an ongoing field of conflicts between the sides. As the Russian team manager explained:

“Finally they realized that they paid much more than they expected. (...) Salaries grew up too much in Saint Petersburg, and (...) I think, currently it makes not big sense to outsource from Germany to (...) Saint Petersburg. Because prices are comparable. (...) [And I told them] I was not ready to continue our contract on these terms” (Interview, May 28, 2008).

Because of the poor performance in combination with the rising development costs, the holding decided to reduce the size of the offshore team to eight—a decision, which further increased the frustration of the offshore team, as the German project manager reported.

According to him, the reduction in the number of employees belonging to the offshore team made it easier to coordinate the shared development, but the financial problems prevailed. He explained that this was due to the growing importance of Saint Petersburg as a software region. Western companies were in search for offshore developers, and the job market was growing rapidly. The lack of social security (sometimes seen as an argument for the attractiveness of a country) made income the only security for employees, and thus contributed to increased salaries. Policies of the German SME to keep salaries low were a constant field of conflict in the offshoring cooperation. At the same time, the small team size made the company especially vulnerable to fluctuation of team

members, while the low level of specialization required extensive training of new developers.

In this context, the German team reported that the Russians tried to use their influence on the development. Conflicts started about the distribution of (inconvenient) tasks and the technical infrastructure. For example, instead of using Sametime for their communication, the Russians started using Google Talk, and instead of using the company's Lotus Notes Database for shared documents, the Russians switched to Google Docs for their daily work.

Furthermore, the Russian team decided to stop using the shared bug-database SQA in favor of a self-developed database in 2007:

“The management of the company wants to get monthly reports concerning the ratio of feature development against the fixing of bugs. The tools [SQA and the product database] distinguish between both kinds, but it is not possible to [...] create an automated report, which the Russians find annoying. Therefore, they plan to administrate features and bugs in a shared database and have begun to develop their own, web based solution” (Field notes, July 10, 2007).

As a result, the teams had to track bugs in two parallel systems, because the German company was reluctant to change their established infrastructure. On the other hand, the German team manager did not want to antagonize the Russian team:

“Basically, [the German project manager] likes the idea [of a shared system], but the report feature is not necessary because they only need rough estimates for the taxes. But to avoid decreasing the motivation of the Russians they let them do as they like, as long it does not involve more work for the company” (Field notes, July 10, 2007).

Therefore, he did not intervene, but his acceptance was based on the condition that the Russians took on the necessary overhead work of maintaining two systems. In addition, the Russians planned to develop an import/export filter for the automatic synchronization of the two databases.

The other changes of development tools, i.e. using Google Talk instead of Sametime and Google Docs instead of Lotus Notes, were justified mainly with the available resources of the developers' computers. Since Sametime, according to the developers, needed much processor time and memory, it was annoying for the Russians to do their everyday work. Using the web-based Google Talk would be much more convenient for them. From the perspective of the German project manager, the decision had another reason. According to him, the Russians wanted to keep up to date with the tools they used. Thus Sametime and Lotus Notes would not be as trendy as the newer Google tools.

The Termination of the Cooperation

In 2007, the size of the offshore team was further reduced to four. Finally, in early 2008 the German holding decided to stop the cooperation completely, first by reducing the team size to two, and then by suddenly stopping the offshoring by the end of the month. The decision itself had neither been unexpected nor was it unwelcome by the German partners:

“All in all, everyone was unsatisfied with the state of affairs. The Russians, because the holding paid unpunctually, the developers in Berlin, because bad work was delivered, and the holding, because everything was considered as being too expensive, and the prices were increasing further” (Field notes, March 12, 2008).

Accordingly, both teams had considered the possibility of terminating the cooperation, and the German project manager had made up a plan together with the Russian team manager which was meant to arrange this termination to be as smooth and easy as possible for both teams. According to the project manager, this was not only due to his own team’s interests, but also due to the personal friendship with the Russian team manager. In this regard, both teams said they would have liked to continue the cooperation under different circumstances, and they blamed the holding management as being the one responsible for the failing of the project.

Hence, in the end, only the abruptness of the decision caught both teams by surprise. As the Russian team manager explained:

“In the middle of December, [the German holding] said, o.k., please keep these four developers until end of May (...). So we will have five months to move the development from Saint Petersburg to Germany. (...) But [then] they said that they had changed their decision and needed only two people until the end of February. This was unexpected (...) and I had to pay salaries for them and even (...) fire one developer“ (Interview, May 29, 2008).

Analysis of Articulation Work and Social Capital

While the last chapter recapitulated the course of events from the perspective of the practitioners, we will now revisit the offshoring story from an articulation work and social capital perspective.

As we were told, the initial phase of the cooperation was supported by a high level of trust between the teams, which was based on the friendship between the German entrepreneur and the Russian team manager. Furthermore, the visit of the whole Russian team to Germany had helped to form social ties between the developers, too. However, despite this high level of social capital, the German team wanted to stay in control of the development as much as possible, as software development was still deemed as the core competency of the company.

The Russians, on the other hand, accepted this distribution of tasks, as it allowed them to concentrate on the technical side of the development only.

However, in order to do so they were dependent on exhaustive specifications of features which the Germans found increasingly difficult to afford. As the initial distribution of labor turned out to be problematic, the German company had to learn that writing complete specifications (even for the standard software product) can be as time-consuming as the development itself (or even more). Instead of being self-explanatory and efficient, the disjunction of requirements-engineering and coding led to severe coordination problems which were caused by the necessary knowledge transfer and articulation work between the teams.

As the workload of the German team increased, the decision was taken to change the distribution of work while ensuring that control remained with the German team. The Russians accepted this change unwillingly. Despite their good technical knowledge, the Russians had difficulties with the task of writing specifications. As they lacked the necessary context knowledge, the effort of writing adequate documentation was very high, even more as they could not draw upon shared business experience with the customer. As a result, the Russians felt overstrained, and the amount of necessary requests, clarifications, and corrections increased—classical aspects of articulation work. At the same time, the dependency on ICT for articulation work created bottlenecks, which were further aggravated due to language issues between the teams. While the trust between the teams was still high, the German team attempted to improve the documentation by introducing standardized forms for specifications and bug descriptions. However, this attempt to support the Russian team in writing specifications did not work, as the necessary knowledge exchange was still insufficient. In contrast: the efforts to formalize the development turned out to be only new forms for informal articulation work, and for related uncertainties in the development process.

Despite the related increase of informal communication—which has been found to support knowledge exchange and even conflict resolution in distributed teams (Hinds & Mortensen 2005)—the company could not benefit from the change, because the Russians lacked the necessary context information which was paramount to successfully accomplish the task of writing proper specifications. After all, communication needs to support the underlying work structures of a team (Hinds & McGrath 2006). Instead of supporting the necessary communication work, the management of the holding—bound by the necessity to coordinate two organizations—reacted by intensifying control and formalization. This in turn was seen as an escalation and systematization of attempts to blame the Russians for the prevailing problems. The social capital which had formed the basis of the commitment of the Russian colleagues started to become eroded—despite the initial high level of trust between the sites, which rested on the personal relationship between the German project manager and the Russian team manager.

The Russians, unable to meet the expectations of the German team, began to neglect certain tasks which were regarded as being annoying and unnecessary, like tracking the progress of their work. The German company had to realize that it was dependent on the commitment of the Russian team and that formal methods of control cannot guarantee personal obligations—or even damage them (cf. Inslund 2003). Even worse, the German team was unable to solve this problem. In this phase of the cooperation, the still high level of social capital apparently hindered an open argument between the teams. The German management avoided blaming the Russians outrightly for not fulfilling their tasks, while the Russians avoided arguments with the German side by simply ignoring inconvenient tasks. In this regard, social capital apparently became a trap: the German manager understood the anger of the Russian team, but regarded the current division of labor as necessary. The Russians, on the other hand, accepted the decisions of the Germans, but felt unable to work under these conditions.

As more and more of the Russian developers left the company, the decision was taken to sell the offshore organization to the Russian team manager. While this decision was approved by both sides, it became the origin of further emerging conflicts, as the cooperation with the now legally independent Russian enterprise made it impossible for the Germans to use hierarchy in order to maintain their idea to substitute informal demands of articulation work by means of intensified formalization. The loss of competent developers was a significant drawback for the company, not only in terms of knowledge, but also in terms of social capital. While the initial cooperation had rested on the personal ties which were formed during the extended personal visit of the Russian team to Germany, the newly hired developers could not benefit from such relations. Instead, the social ties between the teams were mainly focused on the Russian team manager, who shifted his focus to acquire new customers instead of concentrating on the existing cooperation. As a result, the problems with the motivation of the Russian developers aggravated, as social capital as a means for motivating cooperation between the German team and the new Russian developers was weak.

The German team—discontent with the development of the cooperation with Russia—felt trapped: since none of the teams was able to work efficiently without the other team, but every team had the possibility to jam shared projects (by ignoring or by misinterpreting cooperation demands), successful cooperation became unlikely. Collaborative demands on articulation work—considered to be substitutable by formalization and control by the holding management—emerged again on each level of conflict resolution and turned out additional strategic options for constraining the cooperation afterwards.

In this regard, the company also suffered from hard-to-anticipate indirect effects, as, for example, the rising salaries in the region of Saint Petersburg, which were partly connected to the boom of investments in the area and contributed to cost the inefficiency of the cooperation and its termination. Moreover, the change

of infrastructure on behalf of the Russians was hard to foresee. The German team accepted these changes within certain limits because they feared to further discourage the offshore team members.

Apparently, the management of the holding overestimated the possibilities of formal control, and neglected conflict dynamics and social capital issues. As a result, conflicts manifested when coordination necessities emerged on the basis of inter-dependencies in the work constellation which could not be settled by controlling and formalizing the software development. When people tried to solve the problems by means of formalizing articulation work, the situation did not improve, but deteriorated—and was further aggravated by the structural circumstances like rising costs, decreasing social capital and the organizational consequences of the divestment of the branch office.

Conclusion

Our case study illustrates the endeavors of a small German enterprise to keep its offshoring project running. Looking at the related failure story from a long-term perspective, complex and inter-related conflicts within the field of articulation work become visible. The German management, for instance, tried to take advantage of the relatively low wage levels of the Russian partners whom they sought to control by determining their work (by means of the division of labor and tool usages).

The Russian partners turned this claim around by showing that, if control was that important, there was a lack of it in the whole collaboration. How could the Russians work well if the requirement delivered to them were not controlled (for instance, if they complied with international standards)? The counter-reaction of the German management was, again, a turn-around: if the requirements were that difficult to handle, the Russians should write them themselves.

This shows that decisions were taken to shift responsibilities between the offshoring partners. Therefore, articulation work obviously was not only a contingent dimension of decision making, but (in the given case) even attributed to a history of its own in regard of work regulations. The related ping-pong effect of control-based arguments shows that both partners shared related convictions or, at least, did not want to question them. Hence, the changes to the work arrangements were partly the result of continued shared convictions about the necessities of control and formalization. But those were not the only continuities.

While it became apparent that any new regulation led to new areas of conflict, it has to be noted that this did not diminish the mutual appreciation among the actors. Their mutual trust remained through all these conflicts. But what about social capital? It was defined before as “network ties of goodwill, mutual support, shared language, shared norms, social trust, and a sense of mutual obligation that people can derive value from” (Huysman & Wulf 2004). Have these ties declined

throughout the diverse conflicts? The astonishing fact is that partners from both sides still would have liked to collaborate even after the termination of the offshoring project which was seen as a salvation on both sides.

Obviously the actors differentiated between the personality of their partners and the offshoring situation as a whole (cf. Imsland 2005). This implies that the management partly understood the strategies of the Russians to take advantage of higher salaries, or at least did not see it as personally insulting. In contrast, the Russians obviously understood the role of the holding as a limiting factor for the German project manager. Insofar, the trust—with regard to mutual goodwill—among the actors prevailed even through disappointing collaborative experiences regarding opportunistic and sometimes unpredictable behavior. The same was true for social capital in the mentioned sense as an accumulative value.

However, the social capital, which contributed to motivation at the start of the offshoring project, also turned out to be a hindrance at its end, as the assumed knowledge about the personalities of the partners made it easier to detect structural limitations of the situation, and as it apparently hindered an open argument about the prevailing conflicts. This means that social capital can really be an asset in the sense that collaboration would not be possible without it. Nevertheless, it can become dysfunctional, a mis-investment in terms of the capital metaphor. Social capital is not only about cognition, inter-action, and shared perceptions: it also relates to fallible investment of efforts.

This fallible characteristic of social capital is not covered in the necessary detail by Putnam's tradition of social capital as "goodwill, mutual support, shared language, shared norms, social trust, and a sense of mutual obligation". Hence, it seems to be fruitful to expand the given understanding of social capital by referring to Bourdieu's (1986) notion of social capital as a means to reconstruct the risky decisions of individuals when attempting to establish profitable value chains, which can explain why social capital apparently can change from an asset to a hindrance.

In relation to offshoring, it was found that there seems to be something like a tendency of trust to prevail (Zolin et al. 2004). We came to similar results for social capital, but our results also question the concept of social capital as a merely positive factor for global software engineering. Like for trust, it seems we need a much more differentiated understanding of social capital in the context of GSE. Without social capital, GSE as a complex form of distributed collaboration will hardly be possible. On the other hand, formalization and social capital are no guarantee for successful performance. As we have seen, impacts of the international environment and contingencies of articulation work make it very likely in GSE that a given arrangement changes quickly. Therefore, it seems to be a major challenge for GSE to develop forms of making articulation work reflexive and operative, for example, through globally distributed organizational learning.

Acknowledgements

This paper was supported by the German Research Foundation (DFG) within the project “ARTOS – Articulation work in offshoring projects of small and medium-sized enterprises of the software branch”.

References

- Argyris, C., Putnam, R., and Smith, D. M. (1985). *Action science*. Jossey-Bass, San Francisco.
- Billings, M. & Watts, L. (2007). ‘A Safe Space to Vent: Conciliation and Conflict in Distributed Teams’. *European Conference on Computer Supported Cooperative Work*, Limerick, Ireland.
- Boden, A., Nett, B., and Wulf, V. (2008). ‘Articulation work in small-scale offshore software development projects’ *International workshop on Cooperative and human aspects of software engineering*, Leipzig, pp. 21-24.
- Bourdieu, P. (1986). ‘The form of capital’, in J.G. Richardson (ed.), *Handbook of theory and research for the sociology of education*, Greenwood Press, New York.
- Cataldo, M.; Wagstrom, P. A.; Herbsleb, J. D. and Carley, K. M. (2006). ‘Identification of Coordination Requirements: Implications for the Design of Collaboration and Awareness Tools’, *International Conference on Computer Supported Cooperative Work*, pp. 353-362.
- Cohen, D. and L. Prusak (2001). *In good company: how social capital makes organizations work*, Harvard Business School Press, Boston.
- Cramton, C. D. and Hinds, P. J. (2007). ‘Intercultural Interaction in Distributed Teams: Salience of and Adaptations to Cultural Differences’, in G. Salomon (ed.), *Proceedings of the Academy of Management Annual Meeting, Best Papers*, Philadelphia.
- Faergemann, L., Schilder-Knudson, T. and Carstensen, P. H. (2005). ‘The Duality of Articulation Work in Large Heterogeneous Settings – a Study in Health Care’. *European Conference on Computer-Supported-Cooperative-Work*, pp. 163-183.
- Fisher, J., Hirschheim, R., and Jacobs, R. (2008). ‘Understanding the outsourcing learning curve: A longitudinal analysis of a large Australian company’. *Information Systems Frontiers*, 10(2), pp. 165-178.
- Gutwin, C.; Penner, R. and Schneider, K. (2004) ‘Group Awareness in Distributed Software Development’, *International Conference on Computer Supported Cooperative Work*, pp. 72-81.
- Herbsleb, J. D.; Finholt, T. A. and Grinter, R. E. (2001). ‘An Empirical Study of Global Software Development: Distance and Speed’, *International Conference on Software Engineering*, pp. 81-90.
- Herbsleb, J. D.; Mockus, A.; Finholt, T. A. and Grinter, R. E. (2000). ‘Distance, dependencies, and delay in a global collaboratio’, *International Conference on Computer Supported Cooperative Work*, pp. 319-328.
- Herbsleb, J. D.; Paulish, D. J. and Bass, M. (2005). ‘Global Software Development at Siemens: Experience from Nine Projects’, *International Conference on Software Engineering*, pp. 524-533.
- Hinds, P. & McGrath, C. (2006). ‘Structures that work: Social structure, work structure, and performance in geographically distributed teams’. *International Conference on Computer Supported Cooperative Work (CSCW)*, Banff, Canada.

- Hinds, P. & Mortensen, M. (2005). 'Understanding conflict in geographically distributed teams: An empirical investigation'. *Organization Science*, 16, 290-307.
- Huysman, M. and D. de Wit (2004). 'Practise of Managing Knowledge Sharing: Towards a Second Wave of Knowledge Management', *Knowledge and Process Management*, vol. 11, No 2, pp. 81-92.
- Huysman, M. and V. Wulf (2004). *Social capital and information technology*, MIT Press, Cambridge, Mass.
- Imsland, V. (2003): *The Role of Trust in Global Software Outsourcing Relationships*. Ph.D. thesis, Univ. of Oslo.
- Jarvenpaa, S. L. and Leidner, D. E. (1995). 'Communication and Trust in Global Virtual Teams', *Journal on Computer-Mediated Communication*, vol. 3 (4).
- King, W. R., and Torkzadeth, G. (2008). 'Information Systems Offshoring: Research Status and Issues', *MIS Quarterly*, vol. 32/2.
- Lee, J., Huynh, M., & Hirschheim, R. (2008). 'An integrative model of trust on IT outsourcing: Examining a bilateral perspective', *Information Systems Frontiers*, vol. 10(2), pp. 145-163.
- Levina, N., & Vaast, E. (2008). 'Innovating or Doing as Told? Status Differences and Overlapping Boundaries in Offshore Collaboration', *MIS Quarterly*, vol. 32 (2).
- Olson, G. M. and Olson, J. S. (2000). 'Distance Matters'. *Human-Computer Interaction*, vol. 15, pp. 139-178.
- Putnam, R. D. (2000). *Bowling Alone: The Collapse and Revival of the American Community*. Simon & Schuster, New York.
- Ramesh, B.; Cao, L.; Mohan, K. and Xu, P. (2006). 'Can distributed software development be agile?', *Communications of the ACM*, vol. 49, pp. 41-46.
- Rottman, J., and Lacity, M. (2008). 'A US Client's learning from outsourcing IT work offshore', *Information Systems Frontiers*, vol. 10(2), pp. 259-275.
- Schmidt, K. and Bannon, L.(1992). 'Taking CSCW Seriously: Supporting Articulation Work', *Computer Supported Cooperative Work*, vol. 1 (1992), pp. 7-40.
- Schmidt, K. and Simone, C. (1996). 'Coordinaton Mechanisms: Towards a Conceptual Foundation of CSCW Systems Design', *Computer Supported Cooperative Work*, vol 5, pp. 155-200.
- Sole, D. and Edmondson, A. J. (2000). 'Knowledge Sharing in Virtual Teams', *Harvard Business School Working Paper*
- Star, S. L. and Strauss, A. L. (1999). 'Layers of Silence, Areas of Voice: The Ecology of Visible and Invisible Work', *Computer Supported Cooperative Work*, vol. 8, pp. 9-30.
- Strauss, A. L. (1988). 'The Articulation of Project Work: An Organizational Process', *The Sociological Quarterly*, vol. 29 (2), pp. 163-178.
- Strauss, A. L. and Corbin, J. M. (1998). *Basics of qualitative research: techniques and procedures for developing grounded theory*. Sage Publications, Newbury Park.
- Suchman, L. A. (1987). *Plans and situated actions: the problem of human-machine communication*. University Press, Cambridge.
- Zolin, R., Hinds, P., Fruchter, R. & Levitt, R. (2004). 'Interpersonal trust in cross-functional, geographically distributed work: A longitudinal study', *Information & Organizations*, vol. 14, pp. 1-26.