

Divergence and Convergence in Global Software Development: Cultural Complexities as Social Worlds

Rasmus Eskild Jensen and Pernille Bjørn

IT University of Copenhagen, Technologies in Practice, Rued Langgaards Vej 7, 2300 Copenhagen S, Denmark, email: [raej, pbra]@itu.dk

Abstract This study reports the results of a workplace study of globally distributed software development projects in a global software company. We investigated cultural complexities as social worlds and sought to understand how differences in social worlds between geographically distributed developers become salient in their everyday interactions. By analysing both interviews and observations we identified two types of situations where social worlds become salient in the everyday interactions between developers working at different geographical locations: 1) the divergence of concept and meaning and 2) the convergence of concept but divergence of meaning. We argue that these situations are grounded in social worlds and pose a challenge to work practices in the form of miscommunication and misinterpretation of shared tasks.

1 Introduction

Working in globally distributed teams is increasingly becoming the norm for many large international organizations. Globally distributed work settings are malleable and allow work to be transferred across organizational, national, and cultural boundaries [1], which is attractive for organizations involved with flexible and transferable work like software development. Despite the numerous benefits of globally distributed work settings, there are challenges to coordinating work across sites, including establishing common ground [2], creating suitable work practices [3], and overcoming cultural differences [4]. One key challenge for global software development (GSD) concerns communicating and interpreting implicit knowledge, which is not easily shared out of context [5]. Implicit knowledge is socially embedded within work practices and it is not easily shared

across contexts. Communication requires the development of common ground, and common ground is established through grounded processes [6]. When people collaborate and communicate across social and geographical boundaries, such as languages, organizations, and national borders, the risk of misunderstanding and misinterpretation is high.

Investigating communication complexities in geographically distributed situations, CSCW researchers have examined how people with different national cultures interact with each other and apply different types of media [7-9]. Such studies refer to intercultural communication in terms of national culture, with many referring to Hofstede's different dimensions of understanding the disparities between, for example, Western and Asian countries [10]. However, this perspective on culture as a stable entity has recently come under fire, and it has been suggested that culture should instead be investigated as a social construction between people [11]. We join other CSCW researchers [5] in taking a practice-based approach to examining the cultural complexities in GSD by dealing with the issue of culture as part of practice rather than a stable construct based on nationality. GSD is an interesting area to investigate geographically dispersed collaboration because this type of work comprises closely coupled collaborative tasks and, as such, requires a lot of communication.

While we agree that culture beneficially can be understood as a social construct, we discovered particularly complex and pertinent aspects of communication in our empirical case. These aspects can be linked to the society in which the participants are situated and, as such, are related to culture. This observation made us wonder whether we could think about culture as part of practice without submitting to a national culture framework and yet still take the incidents related to society into account. In this study, we suggest that one way to capture the national perspective on culture without submitting to the categories of Hofstede is to think in terms of social worlds. Investigating the communication within GSD, we therefore asked: How can we identify situations where the differences in social worlds between geographically distributed developers become salient in their everyday interactions?

We report on an ethnographic study of GSD practices within an organization that develops IT systems for Danish customers, with developers dispersed across the Philippines and Denmark. The way people work together in GSD is not always apparent. Too often assumptions are made about the task without examining the underlying implicit knowledge embedded within the task. By making visible situations where differences between developers' social worlds are pertinent to the task at hand, we can better understand how social worlds affect collaboration within GSD.

In the following we start by presenting related work on GSD and communication in virtual teams. Then we present the methodology, our empirical case, and the study results. Finally we discuss our empirical observations and offer conclusions.

2 Theoretical Framework

Aside from cost-savings and despite increased collaborative challenges, the key motivators for developing software across geographical locations are interest in leveraging knowledge diversity, exploiting knowledge capabilities, and scaling activities [3]. Some of the collaborative challenges in GSD identified through ethnographic studies include awareness of distributed collaborators [12], coordination [13], and organizational learning [14]. While these are all important, several researchers have pointed to communicating and interpreting the requirement specifications of the IT system under development as a key challenge [e.g. 15,16].

One of the visions for GSD in any given project is to achieve a shared understanding of the system requirements across the various local sites that are part of the project. System requirements are a key artefact of collaborative practices in all types of software development, and they are used to guide, negotiate, coordinate, and communicate about the tasks shared between developers. However, communication is also considerably more difficult across geographical distances because many details need to be made explicit, yet knowledge is often taken for granted. Empirical workplace studies of GSD have shown that the interpretation of system requirements often causes problems and, in some cases, delays projects or even reduces the quality of the final product [16,17].

One strategy for solving this problem is process standardization, stipulating explicit and detailed requirements. Many large global IT companies (e.g., Infosys and TCS) have chosen this strategy. However, recent studies of GSD question the standardization approach because it restricts a company's flexibility and agility, which may be its core competencies [e.g. 3]. If standardization alone is not the solution, we need to find alternative strategies for dealing with the interpretation of inexplicit system requirements created using taken-for-granted knowledge and background assumptions.

2.1 From Culture to Social Worlds

To investigate how taken-for-granted knowledge and background assumptions affect communication concerning system requirement specifications in GSD, we need a theoretical perspective from which we can examine collaborative practices across cultural boundaries. We define culture as a "reference framework, which stipulates roles and interpretations, and which is dynamically negotiated by the actors in the course of their daily work" [5, p. 20]. This view of culture comprises multi-layered assemblages of intertwined practices, values, beliefs, and attitudes that cannot be isolated or directly examined. It includes lived experiences that guide people's behaviour and attitudes, which consist of unarticulated and taken-for-granted knowledge and beliefs [18, p. 229]. In this way, culture serves as implicit directions shaping the interpretation of events. In a collaborative situation

culture operates as a filter through which collaborators can observe and interpret the actions of others [19, p. 133]. While culture as an isolated factor in collaborative practice, we can study culture by examining the manifestations of culture in practice. This approach to studying culture entails investigating practices, artefacts, and activities as they emerge within GSD with the focus of identifying situations where particular cultural aspects are pertinent for interpreting the situation. However, the question remains how to address cultural aspects when investigating the practical circumstances in GSD.

In collaborative practices, cultural differences are most often invisible until a communication breakdown occurs. Communication breakdowns in geographically distributed teams appear to take place within a shared meaning context comprising three levels: work practice, organization practice, and life world practice [18, p. 231]. Importantly, although a communication breakdown may appear at the work practice level, it might be grounded in either the organizational or the life world level. Thus, we are focused on identifying situations in which communication challenges experienced as part of the work practice level are grounded in the life world level. We speculate that essential miscommunications derived from the life world level are grounded in the social worlds that the participants have grown up and lived their lives in. We propose that the meaning participants assign to particular situations or their understanding of a common task is dependent on the background knowledge they have internalized as part of living in particular social worlds [20]. Social worlds can be described as the institutions and notions about society that is shared among a larger population of people. The social worlds influence our perception and understanding of particular incidents we encounter, including communicating with colleagues who are globally dispersed from us.

By investigating situations where social worlds become salient in the everyday work practice we hope to conceptualize and understand how cultural differences affect communication between geographically dispersed participants involved in GSD.

3 Methods

To answer the research question, we chose workplace studies as our methodological approach [21]. Workplace studies seek to investigate and observe the world as it is and try to understand how people act in the world, making it a suitable approach for investigating communication between practitioners during their everyday interactions. The focus of our case study was GlobalSoft, a GSD company of Danish origin with 1500 employees. Typically clients contact GlobalSoft with particular needs for a new IT system; GlobalSoft also answers public calls for tender specifying the requirements for proposed IT systems. Regardless of how the connection between GlobalSoft and the client is created, all projects begin with key negotiations about the scope of the project. In most cases GlobalSoft negotiates directly with the client, with little or no involvement from offshore partners. Once

the project scope is defined, it is divided into tasks, some or all of which are sent to the Philippines, depending on the project. The Filipino department's only input on the project prior to this is when they are asked to do a task proposal estimating the number of hours required for at given task. One of the key documents in this process is the requirement specification, which is meant to define the scope of the entire software project.

As part of a large research project on GSD, we initiated a workplace study with GlobalSoft in November 2010 (study ongoing). In total, three researchers conducted 22 audio-recorded interviews (14 in Denmark, 8 in Philippines) lasting 30–60 minutes (average 50 minutes). Practices were observed in Manila, Philippines, for approximately 180 hours over three periods (December 2010, July 2011, November 2011). In Copenhagen, Denmark, observations of a particular project were conducted for approximately 80 hours. Four workshops (2 in Denmark, 2 in Philippines) were conducted, and various documents and presentations were collected. Employees at many different organizational levels were interviewed, allowing us to compare perceptions of the corporate vice president of GSD with those of the developers.

4 Results

We documented several incidents where the challenges developers experienced could not be explained by normal communication issues like trust [2] or social context [22]. Instead, these incidents were related to the local social worlds of the different participants. Here we present four examples of situations where the differences in social worlds between geographically distributed developers become salient in their everyday interactions: Prescriptions and pharmacy, CVR and CPR, food and health inspections, and retirement plans. We also present two examples of social worlds at work: Inventory facility management system and children with special needs.

4.1 Diversities in Social Worlds

4.1.1 Prescription and Pharmacy

Our data suggest that the Danish employees often experienced their Filipino colleagues misinterpreting or misunderstanding the intended meaning of the requirement specification. The requirement specification contains overall descriptions of all the tasks for a given project. But the descriptions of the tasks are often part of a

predefined context, which is the result of assumptions that are embedded in the requirement specification and which can cause misinterpretations and misunderstandings. A Danish manager from GlobalSoft explained the situation:

[The Danes] should also understand that they [the Filipinos] may not recognize everything. That they spend time talking about what a prescription is. And what a pharmacist is. (Manager)

The manager quoted above spoke about how different understandings of a concept can influence the project. In this example he mentioned prescriptions and pharmacists as concepts that were perceived differently by the Filipinos. In Denmark, all pharmacists have undergone 5 years of university training and are strictly governed by official authorities. All prescriptions are sorted in IT systems and are efficiently monitored by the authorities. Doctors authorize prescriptions after patient consultations and submit them to a general database that all pharmacists can access. Patients can then go to the nearest pharmacy and collect the medicine. The Danish manager recognized that the Filipino employees might not fully comprehend the complexities of how prescriptions and pharmacies are integrated and administrated by the officials in the Danish system. It therefore became essential to talk about what a prescription or a pharmacist was to develop a common understanding of these concepts. The manager was describing the challenge of understanding the local context of Danish pharmacies and prescriptions - a challenge grounded in the social worlds between the Danish and Filipino employees.

4.1.2 CVR and CPR

A Danish IT architect described the challenge of communicating possible differences in social worlds:

But there are also some things we take for granted. I do not need to tell a Danish programmer what a CPR number is, or a CVR number, or many other things, because we take it for granted.... But when we are speaking with the Filipinos... then it is not certain that they have the same knowledge. Such cultural issues, which are something we [the Danes] all know about, are not known outside the borders of the country. And that can easily cause misunderstandings. (IT-architect)

The IT architect described the Danish Central Personal Registry number (CPR), which is unique to Denmark. Every Danish citizen is given a CPR number at birth used as identification for every Danish citizen. In some ways it is equivalent to a social security number, except that it is not optional; everyone must have one. All interactions between the public and the municipalities or the government such as healthcare, taxes, day care, and education, are managed through a CPR number. Thus, the CPR number is an integral part of the social world in Denmark. However, the extensive use of the CPR registration is viewed as controversial in other countries, such as United States, where it is often perceived as unnecessary governmental control of citizens in a democratic country. This example illustrates that outside of Denmark, the concept of the CPR is not fully comprehended, yet the

development of IT systems for Danish institutions will require a comprehensive understanding of the concept and the criteria surrounding it. The Central Company Registry number (CVR) is used for registered companies of a certain size. Companies with revenue of 50,000 Danish kroner or more have to register for a CVR number in a centralized database called Virk.dk and cannot function legally without one. According to the IT architect, there is considerable taken-for-granted knowledge about Danish society that is not easily communicated. This became evident when the Danes were trying to communicate the meaning of the CPR number to their Filipino colleagues, because they lack a shared context.

4.1.3 Food and Health Inspections

Another challenge we documented involved communicating the use context for a particular IT system. In 2001, the Danish government initiated a public food inspection program in an attempt to secure the health of Danish citizens. Inspectors travel the country regularly visiting restaurants and giving them a general hygiene rating. These ratings correspond to various smiley faces, where the cleanest restaurants get a full smiley face and the less clean get a sad face. A Danish project manager had the following experience when working with his Filipino colleagues on a scheduling system for governmental food and health inspectors:

Yes it [the project] is about route schedules. In this case, an inspector who should visit two restaurants in the course of one day, and each visit should take approximately one hour. To a Dane, this is clearly a mistake, because what was the inspector going to do for the rest of the day? But for the Filipinos, (...) well, they did not relate it [the product] to the application. (Project manager)

The project manager quoted above presented an example of how different perceptions of a concept caused a misunderstanding. The developers in the Philippines were coding a scheduling system where data (i.e., number of inspectors and number of restaurants to be checked) were entered and the system generated route schedules. However, at some point an error occurred in the system, resulting in inspectors being assigned to only two visits a day. The Filipino developers tested the IT system using various use-case scenarios and did not find this error. They did not realize that two inspections a day translates into a two-hour workday, which, by Danish standards, is not a sufficient use of resources. To the Danes, this was clearly a mistake. But, according to the Danish project manager, the mistake was grounded not in the Filipino developers' lack of ability but rather in a fundamental lack of understanding of the IT system and how it would be used in Danish society. We suggest that the Danish developers identified the error not based on a superior understanding of the requirement specification but due to a fundamental understanding of the Danish society. As the manager stated, the Filipinos did not understand the use context and therefore could not relate the IT system to the use situation. Not being able to relate the task to a given context is a challenge that both the Filipinos and the Danes are aware of. Yet it still remains a source of miscommunication.

4.1.4 Retirement Plans

The following example illustrates the challenge of interpreting system requirements with unknown terminology, in this case, while developing an IT system for retirement plans. A Danish manager explained:

We had a couple of discussions regarding the retirement concept (efterløn) and your public pension age (folkepensionsalder), which was misunderstood. They [the Filipinos] had understood it in one way and we had another idea of the concept. And this meant that our testing did not match, and at some point the client got involved, because the correction of one error resulted in new errors, basically because they had corrected more than they were supposed to. And it was all caused by this confusion of concepts.
(Advanced Project Manager)

In Denmark, everyone gets a public pension at the age of 65, but many also have a privately funded pension. On top of that, many Danes are part of a public retirement fund called “efterlønsordningen,” which is a supplement to the national pension plan aimed specifically at blue collar workers. This voluntary plan is intended to retire the older generation and create demand for younger workers by allowing workers with physically demanding jobs to retire at 60 instead of 65. In the Philippines they have a social security system where the employer and employee each make monthly contributions based on the employee’s monthly wage. The contributions depend on the salary, and the employer matches the amount contributed by the employee. This is a mandatory minimum that legally obliges Filipino workers and companies to create retirement funds. Workers become eligible for their pension around the age of 55–60, and there is no public pension plan. Furthermore, Filipino law requires companies to pay one month of salary for every year of service to employees who have been with the company for at least 10 years when they turn 55.

These very different retirement plans are built on the particular social worlds of their respective countries. Thus, constructing an IT system to manage retirement plans requires significant knowledge of the social systems in the given country. This knowledge is not easily transferred between developers; it requires considerable communication not only about the system requirements but also about the society in which the IT system will be implemented.

According to the Danish manager, the differing pension systems led to different interpretations of the concept, which led to errors in the IT system that were eventually detected by the Danish testers. This example illustrates the invisibility of different background assumptions and taken-for-granted knowledge. The manager saw a strong relationship between the problems they experienced in the project and the lack of a common vocabulary for the project because the developers shared the concept, but not the meaning behind the concept.

4.2 Social Worlds at Work

4.2.1 Inventory Facility Management System

When the Filipino employees are given a task, it is often in the form of a paper-based requirement specification. We observed two Filipino project managers discussing a task proposal sent from Denmark. They discussed how to determine the scope of the task and how to estimate the number of hours required to complete the task. During the discussion, the developers turned to us and reflected on how difficult and prone to misinterpretation this activity is:

Project Manager: Then there are these requirements like: There should be a...

Interviewer: Bruttoliste?

Project Manager: Yeah, see, we can't understand that. What is that? So what kind of list is that? My assumption is, like, gross list. I am not sure, but based on our assumptions, bruttoliste is like a contract list. So, I am not sure how we can use that term.

The Filipino project manager did not understand the term “bruttoliste”, a Danish word related to calculating inventory. If the word is directly translated, it means a gross list, but, according to the Filipino manager, it could be interpreted as a contract list. The project manager tried to relate the concept to its meaning, which is key to understanding the IT system they are going to create. The challenge was further complicated in this case because the Danish developers did not translate the word bruttoliste. Forgetting to translate is a common mistake in GlobalSoft, despite assumptions about the concept being used to create a sense of meaning for the proposed IT system. To overcome this challenge, the Filipino project managers created a list of assumptions showing how they have interpreted the task description.

There are a lot of assumptions in this project. There are, like, 16 assumptions. Yes, it is quite a lot. Because of the requirements. Did you see the requirements? How would we work on that? (Project manager)

The Filipino project manager was clearly frustrated with the difficulty making sense of the requirements. Creating assumptions was their way of trying to overcome the uncertainties in the task proposal. In this case, they had to create a list of 16 assumptions to estimate the number of work hours for the project. If any of these assumptions are wrong, then a new estimation of time and resources is required. A Filipino project manager continued discussing the gross list IT system:

(...) this facility asset thing...we are not used to those kinds of systems. So, as mentioned before, it basically comes down to the domain knowledge frustration. So, we don't have any domain knowledge about this system. (Project manager)

The manager explained they do not always fully understand the requirement specification. She referred to this problem as “domain knowledge frustration.”

4.2.2 Children with Special Needs

It is not only the lack of a common vocabulary that creates challenges for communication. In GlobalSoft, we also saw examples where differences in social worlds became pertinent for the collaboration. While working to create an IT system for children and youth with special needs, a Danish team leader experienced both a lack of understanding and scepticism as a result of different social worlds:

We had a project about a system for handling children and youth with special needs. And we had negotiated the scope, but they [the Filipinos] never realized that this was a big project of great importance because, in their eyes, they believed that, frankly, you cannot allocate that much money for these activities. They [the children] should be able to look after themselves. Because this is how they do it in their [Filipino] society. (Team Leader)

In Denmark, the social welfare system is investing heavily in children and youth with special needs ranging from learning difficulties to severe physical disabilities. It is a high-priority issue that has general support from most political parties. However, the Danish team leader indicated that the Filipino developers could not grasp the importance of the project. In this example, social worlds became salient. Even after the scope of the project was negotiated, the project leader found it difficult to convince the Filipino developers why the project was important. They showed disbelief that so much money could be allocated to children with special needs. In Filipino society, such children have to look after themselves and would not be supported by the government. The team leader described further:

So it was linked as a central solution to an important project and we had money prioritized for these things, right? But they [the Filipinos] never really took it seriously. Because, in their context, this seems like a completely ridiculous way to spend money. (Team Leader)

This quote illustrates how social worlds become salient in global work. The Danes felt they had a solid project with a straightforward solution and that overall the project was important and highly prioritized by the client—the Danish government. But the Filipino developers were sceptical about the project. They struggled to understand the willingness to spend so much money on children with special needs. Because the Filipino employees had trouble relating this project to a meaningful situation, they were, according to the manager, not able to collaborate in a serious manner.

No, this can be a real challenge because sometimes they [the Filipinos] find it very hard to understand...to understand what really concerns people in Danish society, and why many things can be important in Denmark when they do not understand them at all. (Team Leader)

According to the team leader, the real challenge is a basic lack of insight into the different social worlds embedded in the geographically distributed teams. The Filipino developers have trouble understanding the social contexts in which the IT systems will be applied. This lack of contextual knowledge is an obstacle for the collaboration and may increase the risk of communication breakdowns.

5 Discussion

We have been investigating situations in practice where differences in social worlds between Danish and Filipino developers became salient. Our empirical observations demonstrated how social worlds became a pertinent part of collaborative work in relation to concept and meaning. We set out to identify the types of situations in which the differences between the developers' social worlds were visible and affecting the collaboration. We propose two general situations where social worlds become salient: 1) divergence in concept and meaning and 2) convergence in concept but divergence in meaning. We do acknowledge that the categories "meaning" and "concept" might be a simplistic model for illustrating complex situations of matches and mismatches, but we propose it as a way to unpack the concept of culture.

5.1 Divergence in Concept and Meaning

We saw several examples where concepts essential for the design of the IT system were not part of the social world of the remote site, including the CPR number, a key part of the structure of Danish society that is thus crucial for IT systems designed for the Danish government. Social security numbers are used in the Philippines, but there are clearly differences between Danish CPR numbers and Filipino social security numbers. The divergence between social worlds in Denmark and the Philippines is quite evident, so the parties are aware of the differences. They know they do not share knowledge and must therefore make extra efforts to explain the concept as well as the meaning behind the concept to the remote site. In situations where concepts relevant for the interpretation of the system requirements are localized in the social world of one location, and where the concept is not part of the shared meaning context of the other location, identifying the issue and resolving communication breakdowns is a process of explaining, negotiating, and creating a shared meaning context [18].

In the food and health inspector example, we observed the Filipino developers' failure to relate the IT system to the use context, leading to errors that were easily identified by the Danish testers. In such a situation it is crucial that the remote party identify and question the assumptions embedded within the system requirements, as others have argued [17,16]. We suggest that situations of divergence in concept make it easier to identify the divergence in meaning of the concept because in the remote location, the concept does not exist. In these situations, reinterpretation of the meaning behind the concept is not needed.

5.2 Convergence in Concept but Divergence in Meaning

As we have shown, high diversity in domain vocabulary across sites creates extra work that is critical to creating a shared meaning context for the project. In high diversity situations, developers working across sites might be aware of the risk of misunderstandings, and thus will use considerable resources translating domain-specific knowledge in documents like the requirement specification and the product description. It is relatively easy to identify instances of divergence in concept because the lack of a shared vocabulary is obvious. However, our analysis revealed a different type of situation where the local social world was evident—situations where the concept is shared across locations but has different meanings for the different social worlds, such as the case of the retirement system. The retirement systems in Denmark and the Philippines appeared quite different and were dependent on the social world of each country. We label this a situation where there is convergence in concept but divergence in meaning across different social worlds. Differences in social worlds prove challenging for the design and development of IT systems in cases of a divergence in meaning of key concepts used in both local contexts. Both Denmark and the Philippines have an interpretation of the concept of a pension plan, and the concepts have similarities across locations, such as similar retirement age. There are, however, distinct differences in how the two societies have structured their pension systems, but these differences may not be immediately apparent to the development teams.

The lack of visibility of different parties' interpretations of a common concept makes it difficult to detect this source of misunderstanding. We saw this in the retirement system example, where crucial errors were not detected early by the Filipino testers and were only later discovered by the Danish testers. This suggests that communication breakdowns caused by diversity in meaning of shared concepts are more likely to happen at later stages because participants may perceive a "false" sense of common ground, making the lack of shared understanding harder to identify. Common ground occurs when parties share knowledge and know they share it [2]; however, developing common ground requires a grounding process where the parties rearrange their knowledge according to each other's utterances [6]. In cases of convergence in concept but divergence in meaning, detecting a lack of common ground is difficult and, in some situations, even impossible. In many cases, the lack of common ground will not appear until technical decisions based on false assumptions become manifested in the IT system. This kind of miscommunication is more costly for the company because the cost of fixing errors rises exponentially as the product reaches the delivery deadline. It is essential that the remote party not only identify and question the assumptions embedded within the system requirements, but also that they are aware of the divergence in meaning of what might seem to be a shared understanding of key concepts. We believe identifying situations of convergence in concept but divergence in meaning is difficult but critical to reducing the risk of miscommunication throughout the development process. We argue that re-interpreting the meaning behind the concept is required for developers to establish a shared understanding of the development of

IT systems. Yet, because of the initial shared understanding of the concept, development teams tend not to allocate resources to communicating its meaning and thus fail to address the challenge of divergence in meaning.

You may also wonder about situations of convergence in concept and meaning. We detected no such cases when examining our data for situations where differences in social worlds between sites became salient for communication. We might assume that in situations of convergence in both concept and meaning, no differences in social worlds exist, and thus they are not problematic. Perhaps our empirical material revealed no such situations because they do not result in communication breakdowns, and thus our respondents did not identify these instances as challenges for collaboration in GSD.

We have argued that social worlds can become salient as either divergent in concept and meaning or convergent in concept but divergent in meaning. We propose that these social worlds pose a risk for miscommunication between developers, and we have observed examples of these miscommunications in practice. In the inventory facility management example, where the Filipino project leaders found it challenging to comprehend the IT system's use context, these challenges lead to frustration because a list of assumptions had to be created as a way to address these challenges. Workaround creation in GSD processes has also been noted by other researchers [3]. In such situations, social worlds become salient and project leaders create ways to work around them.

The development of an IT system for children with special needs was another example of social worlds at work. In this case, differences in social worlds led to scepticism among the developers across locations and created a challenge for the collaboration. Thus, the lack of implicit directions for shaping the interpretation of events hindered the developers' ability to enact shared meaning [18].

Based on our observations of social worlds and their impact on collaboration, we suggest the need for further research into understanding how divergence and convergence in concept and meaning affect collaborative work in GSD. We propose that divergence in concept and meaning is easier for participants to identify and comprehend and will therefore often be present early in the development process. In contrast, convergence in concept but divergence in meaning is harder to identify and more complicated to comprehend because it requires reinterpretation of familiar concepts and we suggest that these situations will occur later the development process.

6 Conclusion

In this study, we present an analysis of a work place study in a global software development company. We set out to investigate how to identify situations where the differences in social worlds between geographically distributed developers become salient in their everyday interactions. In our analysis, we identified two

types of situations where social worlds became pertinent: 1) situations of divergence in concept and meaning and 2) situations of convergence in concept but divergence in meaning. While we acknowledge that the conceptualization of concept and meaning might be somewhat simplistic, we believe this to be a first step into unpacking culture as part of collaboration. Based on our empirical findings we have argued that by identifying and describing these situations, we can better understand how and why communication breakdowns occur in intercultural collaborative work practices.

Acknowledgements

This research has been funded by the Danish Agency for Science, Technology and Innovation under the project "Next Generation Technology for Global Software Development", #10-092313.

References

1. Majchrzak A, Rice RE, Malhotra A, King N, Ba S (2000) Technology Adaptation: The Case of a Computer-Supported Inter-Organizational Virtual Team. *MISQ* 24 (4):569-600
2. Olson GM, Olson JS (2000) Distance Matters. *Human-Computer Interaction* 15:139-178
3. Avram G, Bannon L, Bowers J, Sheehan A (2009) Bridging, patching and keeping the work flowing: Defect resolution in distributed software development. *Comput Support Coop Work* 18:477-507
4. Krishna S, Sahay S, Walsham G (2004) Managing cross-cultural issues in global software outsourcing. *CACM* 47 (4):62-66
5. Boden A, Avram G, Bannon L, Wulf V (2009) Knowledge management in distributed software development teams: Does culture matter? Paper presented at the International conference on Global Software Engineering (ICGSE), Limerick, Ireland, 13-16 July
6. Clark H, Brennan S (1991) Grounding in Communication. In: Resnick L, Levine J, Teasley S (eds) *Perspectives on Social Shared Cognition*. American Psychological Association, Washington DC, pp 127-149
7. Massey AP, Hung Y-TC, Montoya-Weiss M, Ramesh V (21) When culture and style aren't about clothes: Perceptions of task-technology "fit" in global virtual teams. In: *GROUP*, Boulder, Colorado USA, 2001. ACM, pp 207-213
8. Nardi BA, Whittaker S, Bradner E (2000) Interaction and outercation: instant messaging in action. Paper presented at the Proceedings of the 2000 ACM conference on Computer supported cooperative work, Philadelphia, Pennsylvania, United States
9. Setlock L, Fussell S (2010) What's it worth to you? The costs and affordances of CMC Tools to Asian and American Users. Paper presented at the Computer Supported Cooperative Work (CSCW), Savannah, Georgia, US
10. Duarte D, Snyder N (2006) Mastering virtual teams: Strategies, tools, and technoques that succeed. John Wiles & Sons Inc, San Francisco, California
11. Söderberg A-M, Holden N (2002) Rethinking cross cultural management in a globalized business world. *International Journal of Cross Cultural Management (CCM)* 2 (1):103-121

12. Souza Cd, Redmiles D (2007) The awareness network: To whom should I display my action? And, whose actions should I monitor. Paper presented at the European Conference on Computer Supported Cooperative Work (ECSCW), Limerick, Ireland
13. Espinosa A, Slaughter S, Kraut R, Herbsleb J (2007) Team knowlegde and coordination in geographically distributed software development. *Journal of Management Information Systems* 24 (1):135-169
14. Boden A, Nett B, Wulf V (2010) Operational and strategic learning in global software development: Implications from two offshoring case studies in small enterprises. *IEEE Software* 27 (6):58-65
15. Herbsleb J (2007) Global software engineering: The future of socio-technical coordination. Paper presented at the Future of Software Engineering (FOSE), Washington, DC, USA,
16. Boden A, Nett B, Wulf V (2009) Trust and social capital: Revisiting an offshoring failure story of a small German software company. Paper presented at the European Conference Computer Supported Cooperative Work (ECSCW'09), Vienna, Austria, September 7-11
17. Herbsleb J, Paulish D, Bass M Global software development at Siemens: Experience from nine projects. In: International conference on software development (ICSE), St. Louis, Missouri, USA, 2005. ACM, IEEE Xplore, pp 524-533
18. Bjørn P, Ngwenyama O (2009) Virtual Team Collaboration: Building Shared Meaning, Resolving Breakdowns and Creating Translucence. *Information Systems Journal* 19 (3):227-253
19. Ngwenyama O, Klein H (1994) An Exploration of the Expertise of Knowledge Workers: Towards a Definition of the Universe of Discourse for Knowledge Acquisition. *Information Systems Journal* 4:129-140
20. Star SL, Griesemer J (1989) Institutional ecology, translations and boundary objects: Amateurs and professionals in Berkeleys museum of Vertebrate Zoology, 1907-39. *Social Studies of Science* 19:387-420
21. Randall D, Harper R, Rouncefield M (2007) *Fieldwork for design: Theory and practice*. Springer, London
22. Jarvenpaa SL, Leidner DE (1999) Communication and Trust in Global Virtual Teams. *Organ Scien* 10 (6):791-815