2008 IEEE International Conference on Global Software Engineering

On the Suitability of Particular Software Development Roles to Global Software Development

Lane, Michael T.
University of Limerick and Lero – The Irish Software Engineering Research Centre
Michael.lane@ul.ie

Ågerfalk, Pär J.
Uppsala University, Sweden and Lero – The Irish Software Engineering Research Centre
par.agerfalk@dis.uu.se

Abstract

Global software development surfaces various challenges and benefits that are not always present in co-located teams. The purpose of this paper is to explore a set of propositions that address the suitability of four different software development roles to Global software development (GSD). A qualitative research approach was applied to collaborations undertaken between remote counterparts playing the same development role in various GSD projects. Specific development roles were considered: business analyst, designer, developer and development-support. A framework that details the benefits and challenges of GSD was used as a basis for this research. Suitability of a role to GSD is based upon the balance of challenges and benefits discovered in that role's case. Finally, opportunities for future research are presented.

1. Introduction

Globally distributed software development (GSD), a phenomenon that emerged in the 1990's, involves the production of software systems using a distributed set of staff. There are many configurations of GSD teams and examples range from remote sub-teams producing specific modules of a system to teams where different functional roles such as programming or business analysis are executed in different locations [1].

Teams that conduct GSD may gain certain advantages over those that are co-located. Conversely, distributed teams may encounter obstacles that are not major issues in co-located teams. A review of the published peer-reviewed literature on GSD case studies proposed a framework of the benefits and challenges related to this field [2]. This work and much of the literature that it summarized primarily used the GSD team as the unit of analysis. Given the importance of peer-to-peer interaction in GSD projects, our research used this framework to explore collaboration between individuals that had played the same development role. In addition to being a novel feature of this study, it also helped to focus the investigation.

Four particular roles were selected for investigation: development support, designer, developer and business analyst. These roles were selected due to their central influence on software engineering. Based on previous research [3] and the first author's industrial experience, a research proposition was presented and explored in relation to each role's suitability to GSD. Clearly, other roles could be reviewed in future research efforts. All GSD situations explored involved a team within a product development organization. The need for proximity to global markets and access to diverse global experiences makes GSD attractive to the producers of product software. Hence, concentration on one type of team narrowed the research parameters to teams whose needs are aligned with the assumed benefits of GSD and thus clearly identified a target audience for the study findings.

The paper proceeds as follows. First, a review of the software development roles and GSD teams is presented, followed by a discussion of the research design. Second, the analysis of the research findings is described. Finally, the relevance of this study to both industry and the research community is outlined followed by a presentation of future research opportunities.

2. Theoretical background

2.1. Development roles studied

Software development teams exhibit skills such as strong technical competence, programming, quality assurance, management and good domain knowledge [4]. All team members do not have to display these skills equally – the extent to which each skill is relied upon depends upon the role being played. Software development roles are defined to execute various activities of the software lifecycle and the process in use governs these activities.

978-0-7695-3280-6/08 $25.00 © 2008 IEEE
DOI 10.1109/ICGSE.2008.19

In a product software context, the requirements engineering process involves the elicitation, analysis, specification and validation of requirements [5]. Initial high-level requirements are generated and validated by a product manager working with marketing data and customers [6]. A *Business Analyst* performs the activities required to expand high-level needs into detailed requirements. Different requirements engineering situations may influence the elicitation techniques used or the type of software specifications produced [7], [8]. The specifications provided by the business analyst are used in the design process to produce a logical design that guides code development. Detailed design involves the specification of the software to be developed. Its primary motive is to direct code development. Many notations and design methods are available to develop and communicate designs. A *Designer* will need to be proficient in the tools and techniques used by his/her team. [9] Design is quite often a collaborative activity and as such is exposed to many of the issues around group interaction. These are further exacerbated in a GSD environment [10]. A *Developer* is responsible for the implementation of the design by production of code using some pre-defined programming language. The role of *Development Support* is responsible for the support of tools such as configuration management (CM) systems and computer aided software engineering (CASE) systems.

2.2. Packaged Teams

Two categories of software development teams can be distinguished [4]: "Information Systems" (IS) teams and "Packaged" teams. IS teams tend to build internal systems or provide services to build bespoke software based upon customer contracts. Packaged teams are responsible for the creation of software products that are sold to wider markets.

Packaged teams tend to be less cost-conscious and have more access to resources than IS teams. They also tend to be more aware of duration pressure due to time-to-market considerations and exhibit a more solitary approach to development. IS teams tend to be more matrix-based whereas packaged teams are co-located. IS teams have more mature development environments and processes. This would be consistent with the earlier finding that packaged teams use more solitary or individualistic approaches to software development. [4]

The research conducted in this study focuses exclusively on packaged teams. As stated earlier, this research may be of value to packaged teams in building development groups tasked with the production of globally marketable software products. Given the differences between these team types, it is possible that any GSD impacts uncovered using research on packaged teams may differ if applied to IS teams.

2.3. Global Software Development

A global software team is typically separated by a national boundary while collaborating on a common project [1]. More specifically, global software development (GSD) can be defined as the execution of any software lifecycle activity (including maintenance) by a group of people who are geographically dispersed [2]. Essentially, GSD is the collaborative production of software across sites.

There are various motivations that have prompted the growth of GSD. These include the need to reduce costs, gain proximity to customers, exhibit a global image, reduce development project timelines and leverage specialized skill sets. These motivations are further supported by various improvements introduced by GSD teams. These include the encouragement of disciplined process to manage distance issues and the promotion of innovation caused by the diversity of team member's backgrounds. [1]

Another approach to the exploration of issues and benefits of GSD was provided by Carmel's treatment of the "centrifugal" and "centripetal" forces that influence this form of development. He lists distance, cultural differences, loss of "teamness" and impacts to communication, coordination and control mechanisms as issues that favour collocated development over GSD [1]. Complexities to design introduced by distance are further detailed by Rafii [11], while further reference to the reduction in coordination and control is provided by DeSouza's [12] views on "opportunistic interactions". The coordination issue of expertise identification and selection is highlighted by DeSouza [12] and is also prevalent in a review of coordination issues within different distribution configurations [13]. Another issue reported in that review was mistrust between team members due to lack of informal communication and this point supports the potential impact to "teamness" imposed by GSD.

Carmel [1] proposed a number of centripetal forces, or solutions, that help make GSD work. These included a strong telecommunication infrastructure, use of collaborative technologies and software development methodology. Certain team configurations may also support GSD depending upon the organization's resources and type of product being developed. These include modular structures, phase-based structures; functional expertise-based structures; customization-based structures and team configurations that are time zone based in order to transfer work through the 24-hour day ("follow the sun"). Managerial techniques to

tackle trust and team cohesion issues may be employed to help foster effective GSD. Techniques such as lateral communication may help push decision making to the parties with most information on a subject [14].

In an attempt to synthesize the published peer-reviewed literature, Ågerfalk et al [2] developed a framework listing benefits and issues related to GSD (see Table 1). As can be seen from the table, they focused on communication, coordination and control processes from the perspectives of geographical, temporal and socio-cultural influences. This framework was used in this study to drive data collection and analysis activities, as discussed in section 3.

Table 1: Overview of the Framework for GSD issues [2]

Process	*Dimension*		
	Temporal Distance	**Geographical Distance**	**Socio-Cultural Distance**
Commu-nication	Reduced opportunities for synchronous communication, introducing delayed feedback. Improved record of communications.	Potential for closer proximity to market, and utilization of remote skilled workforces. Increased cost and logistics of holding face to face meetings	Potential for stimulating innovation and sharing best practice, but also for misunderstandings.
Coordination	With appropriate division of work, coordination needs can be minimized. However, coordination costs typically increase with distance.	Increase in size and skills of labour pool can offer more flexible coordination planning. Reduced informal contact can lead to reduced trust and a lack of critical task awareness.	Potential for learning and access to richer skill set. Inconsistency in work practices can impinge on effective coordination, as can reduced cooperation through misunderstandings.
Control	Time zone effectiveness can be utilized for gaining efficient 24x7 working. Management of project artefacts may be subject to delays.	Difficult to convey vision and strategy. Communication channels often leave an audit trail, but can be threatened at key times.	Perceived threat from training low-cost 'rivals'. Different perceptions of authority/hierarchy can undermine morale. Managers must adapt to local regulations.

3. Research design

To support the need for data induction, a qualitative research approach was adopted. Preparation of the study resulted in the development of a conceptual framework (see Figure 1). This established that different cases representing each role would be investigated. It also defined the type of data to be gathered, focused the analysis, and promoted consistency across cases. This structured process helped to guide activities and avoid data overload [15]. It describes the different roles that were studied and presents aspects of the different frameworks used to guide the research. Processes and dimensions of the GSD issues framework produced challenges and benefits that were used to investigate the different GSD collaborations.

The research design approach adopted in this study was primarily influenced by the recommendations of Miles and Huberman [15] and Yin [16]. A three-stage iterative approach was followed, which allowed insights from one case to influence the research conducted in a subsequent case (see Figure 2). The creation of a case study protocol and a case study database has been suggested as mechanisms for introducing operational steps and traceability to a study [16]. Usage of a case study protocol reduced a lot of the effort in phase implementations – this was especially important for data analysis, as there were many steps and forms to be completed. "Prior instrumentation" was recommended as a mechanism to assist internal validity by assuring that "*a comparably measured response*" is obtained across informants [15]. Usage of the initial conceptual framework to direct a clearly defined interview format helped to control these investigations and could possibly be applied to future research.

A key component of case study design is the research propositions [16]. Each case dealt with a clear proposition. Specifically, it was proposed that the benefits of GSD might outweigh the challenges in the cases of business analysts and development support personnel. Conversely, for situations where collaborations were required between distributed

designers or distributed developers, the proposition was that the challenges imposed by GSD might outweigh the benefits. These initial propositions were based upon the first author's experience managing personnel playing these roles in various GSD situations.

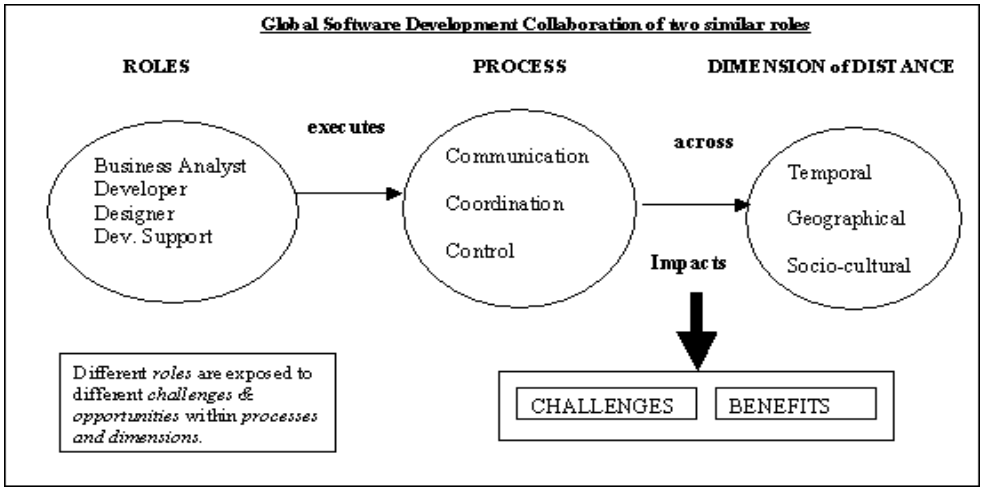

Figure 1: Conceptual Framework

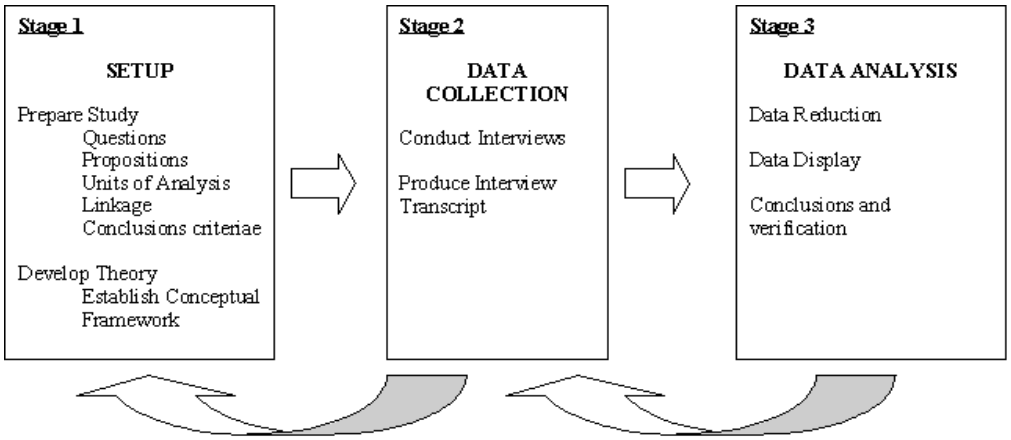

Figure 2: Iterative design approach (after [16])

The critical incident technique [17], cited in [18] was used to help guide research questions and promote interpretive consideration of experiences by the interviewee. The key to this technique is that an interviewee can clearly identify the objective of the incident and understand its impact. This allows them to form an opinion of the workings of an organization based upon their interpretation of the consequences of the incident [19]. This approach was also motivated by the need to avoid the proposition unduly influencing the selection of cases under study.

The main method of data collection was the focused interview [20]. The framework of GSD issues (Table 1) was used as a basis for deriving an interview guide [21]. Data analysis was then achieved by reducing the data using categorization techniques, presenting the data in clear manageable formats and the creation and verification of various conclusions. To facilitate research into the advantages and disadvantages of GSD, the framework overview shown in Table 1 was expanded into a more detailed presentation of specific challenges and benefits related to each of its processes and dimensions [2]. This study tailored that elaborate

framework in order to interpret and measure the data collected. Research of role suitability to GSD required interpretation of the findings to either confirm or refute each proposition. This interpretation was performed in each case by the identification of challenges and benefits, supplemented by a subjective assessment of their impact on the case. A rating for each benefit and challenge was generated from the assigned impact. The interviewee's opinion was also solicited on the overall subject as this high-level question had the potential to induce viewpoints missed by the framework-based questions. The tailored framework and its contents are described in greater detail in the conclusions section of this paper.

4. Research findings

4.1. Developer case

This case supported the proposition that GSD challenges outweigh its benefits. It dealt with collaboration between a distributed team of software developers from a project that employed a module-based distribution configuration. This team was responsible for the extension of a framework that enabled the generation of the presentation layer of a supply chain management product. A major obstacle faced by the team was that the existing framework was unstable, immature and poorly documented. A further complication to this effort was the pressures to use the framework to produce user interfaces for parallel domain layer development projects. All developers in the user interface team were originally co-located in California and had a basic knowledge of the framework. This team was extended to add members from module-based locations including Ireland, Minnesota and California.

The critical incident used by the developer occurred in November 2001. It referred to a situation when the team's remote project manager re-prioritized work activities in order to exert more control over the work effort. The introduction of clear work breakdown schedules and extensive task estimation exposed major deficiencies in the team's awareness of critical tasks and their estimation of future development.

Examples of challenges that had a large impact included coordination complexity related to both temporal and geographical distance. One of the key aspects of this challenge is that tasks may need to be managed closely.

"Work was not broken down to coordinate tasks. Even the project manager was exceptionally remote from the project. If she was there to guide meetings she would have got the team to be more attentive". (Developer)

The case also revealed a lack of trust between remote developers. The distributed team used an internally developed framework to build the application. One remote developer experienced difficulties understanding various poorly documented framework patterns and extension mechanisms. However he was unable to engage in synchronous communications with his more experienced remote colleagues due to the lack of overlapping working hours. Not only did this temporal obstacle lead to reduced productivity, it also provoked doubts within the wider team about the remote developer's ability. Time zone issues did not allow him to clarify misunderstandings. The remote developer had joined the wider team from an office that espoused a strong culture of process innovation. However, the developer found it very difficult to influence any team-wide improvements. The aforementioned lack of trust impacted the group's ability to realize any benefit from cultural diversity.

"Being new on the team you do go through your learning curve and make mistakes. I could check in stuff that would break something else and they would have to work through code to find out what happened. If I was there I could tell them immediately of what I had done, but not being there it caused them a lot more work and that happened a few times". (Developer)

Only one benefit was noted. It recounted certain instances when time zone differences led to the efficient use of time to reduce project duration. There was little success with coding activities but there appears to have been some positive impacts when working with the test group.

"*For standard development – there were no huge advantages. When it came to QA of work and discovering issues, when you would eventually get to work with somebody 8 hours away, they could continue to debug it during my evening and provide some feedback for me. When it worked, it worked well*"

4.2. Designer case

This case involved collaborative situations between designers from a packaged team that had responsibility for the development of a recently acquired warehouse management product. The team was formed to extend the product and also ensure it conformed to the corporation's standards and processes. The team was distributed over four locations: Barcelona, Brussels, Leeds and Limerick.

The critical incident commenced in September 2004 and was completed by November of that year. Its primary focus was the migration of product development from individual desktop/laptop configurations to a controlled development

environment. This task required assistance from a dedicated development-support technician located in California. This work was performed in parallel with ongoing design activities due to existing customer commitments.

The general pattern of findings from the designer case was that there were many obstacles to effective design between remote parties. The lack of informal communication may have resulted in erosion of trust. During the interview, the designer referred to his uncertainty in relation to the motives of his counterpart.

"Because there was no rapport built up – you assume the good of everybody but at the same time you don't have anything to go on".

Another coordination issue dealt with the reduced team spirit and increased frustration caused by the short overlap in collaborative time. This issue was foremost at the outset of the incident as the need for group work was greater. Once the project and team had built relationships and gained some momentum in their application of processes, there was a better opportunity to stagger tasks.

"*On reflection, I don't think that productivity was as high as it could be. It was due to the geographical nature. There was a department that served people as best they could, but with a time difference, at the early stage the set up was very frustrating due to there being a lot to be done in a small amount of time. You reach a stage where you can put tasks in sequence or parallel and take benefit from staggered days, however at the start or set-up, you need everybody to work together ...*"

However, the challenge of coordination across geographical distance was seen to have the highest negative impact on this case. Without the context supplied by face-to-face communication, it seems to have been difficult to convey effective abstractions or arrive at conclusive agreements. These difficulties could prove a major obstacle given the collaborative nature of design activities.

"In my experience, over a number of projects – I never saw an advantage to doing a design globally. Where I saw a benefit was when a very clear description of what is to be done is provided to a resource that can do it and they can turn it around in a different time zone. But if you are trying to abstract something with global meetings, it can be very time-consuming and ineffective."

Although the above designer case presented a negative view of GSD due to different challenges encountered, a separate critical incident provided evidence of a benefit that illustrated the power of global development when it works effectively. The designer being interviewed outlined an incident that occurred in February 2002 that involved interactions between two remote designers developing a CRM product. These designers were members of a distributed team based in various US states, Ireland and France. At the outset of the incident, both designers spent a number of weeks co-located. During this time they built a strong relationship and appreciation of each other's approach to work. They created a collaboration process for use when one party had to return to his home location. It is likely that the emphasis on up-front organization of the collaboration led to the success of this incident (cf. [11], [19]).

"We discussed (a design issue), documented it in email and proceeded. I would always speak at the end of my day, and would always have an email sent to him before the start of his day. He always sent an email at the end of his day. Also, reviews were fully threshed out between us and prepared so larger reviews went much more smoothly".

4.3. Development-support case

This case involved incidents of collaboration between development-support personnel. These employees formed a distributed team responsible for the support of development environments used in the creation of software products. The corporation had many products under development throughout the globe. Team members were located at various offices throughout the world. The corporation had 24 offices worldwide including R&D facilities in California, Ireland, China and Australia. The development-support team had members based in each of the R&D facilities and also at other strategic locations where they provided technical support to various development groups in their assigned region.

The critical incidents involved in this case dealt with support of development environments for two products (A and B). Each of these products was considered to be the responsibility of the Limerick, Ireland facility and was developed by a distributed team that contained employees from India, multiple European countries and US states. Product A was developed using a leading-edge framework. It was quite immature, exhibited many stability issues and offered a steep learning curve to developers. Product B was developed using a mature technology that was familiar to most developers in the organization.

Coordination difficulties can arise due to geographical and temporal distance. The development-support group sometimes required multiple people with different experiences to solve a very difficult issue. However, situations arose where critical tasks handed off to colleagues in a different location were suspended due to re-deployment of that colleague to other work.

"I'm assigned to a subset of projects that exist within the company. What is a hot issue for me that I involve one of my colleagues to help with, may not necessarily be their hot topic. Its very probably they will get called off to do something else …"

Collaborations between remote development-support personnel served to share the rationales behind usage of standard configurations and product development environments. This promotion of standard practices across sites reduced support costs and maximized potential deployment of developers to different teams. Proximity to the remote development groups allowed them to interpret developer needs and react quickly to issues. It is probable that the remote development-support person acted as a cultural liaison to the wider group when representing the needs of their assigned development group.

Diverse technology stacks had been introduced to the company due to the needs of new product development. Access to a wide mixture of backgrounds and experiences meant that the development-support organization was better equipped to support this broad technology base. Remote communication issues encouraged the discipline of using clear traceable processes. The distributed team promoted the infrastructure and culture of a learning environment.

"It has encouraged me to get documentation in place and pushed out to all members of the team and to proactively push the information out regularly."

Time zone effectiveness was of great benefit to the development-support role. The 24-hour day could be used to monitor script executions that were sensitive to failure from trivial errors.

"Its very convenient at the end of my day to ask folks from another time zone to check the job at some time during their day". (Development Support)

4.4. Business Analyst case

This case involved collaborations between business analysts from a packaged team that had responsibility for the development of a globally marketed software product. The business analysis team was responsible for the creation of detailed use-cases that served as specifications for design, development and test activities. Team members were located in Ireland, India and Southern California.

The critical incident under discussion occurred in March 2004 and lasted for eight weeks. It reflected events within the team when the lead business analyst left the company. This person had dictated the format and style used in the production of artefacts and also was acknowledged as the "product visionary". As such, many analysis discussions were heavily influenced by his participation

Findings from the business analyst case resulted in the challenges and benefits appearing to balance one another. It was evident that certain operational aspects of the role are more easily performed in a co-located team. However, the overarching view was that market proximity and customer knowledge were vital to effective business analysis in global product development. These key elements were best achieved by using business analysts with strong experience and exposure to the target locations of the product. This factor led to the conclusion that although the findings were finely balanced, the benefits of GSD outweighed its challenges. The distributed team benefited from the richer base of information provided by the diverse experiences of different team members. However, establishing shared understandings among the team posed a serious challenge to their productivity. Geographical distance reduced opportunities for face-to-face communication. Cultural nuances impacted on the ability to determine whether remote parties clearly understood communications. Although the distributed team benefited from a richer base of information, various costs of accessing the data were noted:

- Cost of preparation of communication to help prevent misunderstandings.
- Cost of delays due to misunderstandings.
- Cost of lost productivity due to other party working on incorrect data.
- Cost of relationship due to frustrations that could emerge from either misunderstandings or because of over elaboration of message.
- Cost of travel dealing with certain complex issues that required the richer context provided by face-to-face communication.

"You explain something in an email or phone call … you don't have any eye contact or body language to indicate that the message wasn't understood … certain cultures have a problem telling you to clarify an issue, especially in a group setting. Definitely they had issues in asking me to clarify something twice. I would outline a problem and they would go away and come up with a solution that would not be related to the problem."

Furthermore, challenges were noted in relation to communication delays caused by time zone differences.

"I would often send a request to people as my last task of the day and I would let them know that I am stuck. I would ask them to review the issue and set me up to continue working … There were sporadic incidences where there was no reply … This would initially annoy me – I would then have to reorganize my day. It was more usual that I got an irrelevant response or I was asked a clarifying question. …".

A key benefit acknowledged by the business analyst was the manner in which work was distributed. The modularization of work motivated by the temporal and geographical coordination issues was very effective. It promoted independent thought for the analyst and his development team and also encouraged the establishment of clear interfaces to other modules thus supporting subsequent integration activities.

"The discipline amongst the group was very good. It was very much that each BA had an area they were responsible for and each individual created artefacts that were reviewed by the team"

The tipping-point in declaring that benefits outweighed challenges for this role was the discovery of the criticality of diverse experiences to job success.

"Access to diverse specialities was critical in relation to global software packages … dealing with tax restrictions, regulatory issues etc. We had a Business Analyst who was a CPA in the US, and other automotive experts. You are doing GSD because you propose to sell your product to the globe. In terms of how the software must deal with tax, regulatory approval, etc. you must deal with this".

5. Conclusions and future research

This study addressed the impact of GSD on collaborations between remote parties playing particular development roles. To this end, a separate specific case appropriate to each role was studied. Table 3 summarizes the findings. It was found that the challenges of GSD outweighed the benefits for developers and designers while the benefits outweighed the challenges in the case of development support. The business analyst case revealed an even balance between the advantages and disadvantages of GSD. However, benefits were deemed to outweigh challenges due to the impact of one particular benefit, "diverse knowledge and market proximity". It was found that this benefit was critical to the successful performance of the role.

As stated above, measurements of the different challenges and benefits found in the various cases were collated in a table that was built from the GSD framework (Table 3). Each challenge discovered was initially given a rating of –1 and this was multiplied by a weighting factor of 1 to 5 depending upon the impact of the challenge to the case. Conversely, each benefit was assigned the product of +1 and its related impact weighting.

The findings from this study and the data that they are based upon provide novel contributions to the general body of GSD research. A notable distinction of this research into the field of GSD is its focus on specific roles. Contrary to current views that tend to approach GSD issues such as temporal distance from a general perspective, this study proposed that treatment of these issues needs to acknowledge the diverse situational needs of different software development roles. It was found that particular roles were more suited to GSD than others in the studied cases. An analytic framework constructed from published peer-reviewed case studies [2] was used to investigate specific industry cases. This use of the framework ratified it as an appropriate vehicle for research. Although it was not the intention of this study to test the framework, it is a notable side effect that inquiries generated from the framework structure resulted in consistent useful findings.

In industrial contexts, the findings could be used to influence distributed team configuration strategies. Team structures could be built to facilitate the co-location of developers and designers. Matrix organizations could ensure that business analysts and development-support personnel realize the benefits of distributed teams. Organizations could focus on role-based support systems in order to counteract challenges that specifically reduce the productivity of certain roles. An example of a crucial task that impacts the designer role is sharing their comprehension of complex issues. Realization of this challenge could lead to it being proactively tackled by mechanisms such as training, work practices or team configurations. Certainly, there are a number of potential extensions to this research effort. Such activities could leverage the case study protocol and database. Findings could be further validated using replication theory if more cases from different organizations were incorporated into the study [16]. Future research could widen the set of cases to incorporate additional roles and cross-role collaborations. Another approach could be to explore situations within a common project that impacted upon multiple roles. While this study concentrated on four specific roles performing same-role collaborations, many further avenues could be explored.

One particular approach may be to extend the research conducted by Flor [24] in his investigation of properties of collaborations. He proposed that properties present in co-located pair programming situations could be used to devise infrastructural tools to facilitate effective globally distributed pair programming. The role-specific approach of this study could be elaborated to explore different properties present in collaborations between distributed parties. This information could also be used in the development of infrastructural tools for all same-role collaborations.

Table 3: Outline of findings from case studies using a tailored version of the GSD framework by [2]

Framework Aspect	CHALLENGES & RATINGS					BENEFITS & RATINGS				
	Specific challenge	BA	DES	DEV	DEVT. SUPP.	Specific Benefit	BA	DES	DEV	DEVT. SUPP.
Communication (Temporal)	Delayed Communication	-2	-2		-2	Time zone effectiveness	1	3		3
	Delayed Feedback			-2						
Communication (Geographical)	Lack of informal communication		-2		-1	Proximity to market/customer	5			3
	Dependency on ICT		-2							
Communication (Cultural)	Language Differences and misunderstandings		-1							
Coordination (Temporal)	Coordination complexity	-2		-3		Modularization of work	3			
Coordination (Geographical)	Reduced Trust	-1		-1		Access to large labour pool	1			3
	Lack of awareness / team spirit		-2							
	Lack of mechanisms for creating shared understandings		-3	-2		Standardization in work practices				3
	Coordination complexity			-3						
Coordination (Cultural)	Lack of mechanisms for creating shared understandings	-3				Mix of skills & experience	5			
	Lack of awareness / team spirit			-2	-2					
Control (Temporal)						Time zone effectiveness	1		2	2
Control (Geographical)	Lack of concurrent engineering principles	-3		-3		Allocation of roles & team structure				2
Control (Cultural)	Perceived threat from low-cost alternatives	-2								
	Adapting to local formalized norm structures			-2						
Totals		-13	-12	-18	-5		16	3	2	16

Findings from these research efforts could lead to a greater understanding of how to overcome obstacles of distributed development and leverage its benefits. This may also lead to additional appreciation of different roles and their training and development needs in order to equip software engineering professionals to deal with the demands for increased flexibility and global awareness in the performance of their duties.

References

[1] E. Carmel, *Global Software Teams: Collaborating Across Borders and Time Zones,* Upper Saddle River: NJ, USA:Prentice Hall, 1999.

[2] P. J. Ågerfalk, B. Fitzgerald, H. Holmstrom, B. Lings, B. Lundell and E. Ó Conchuir, "A framework for considering opportunities and threats in distributed software development," *Proc. International Workshop on DistributedSoftware Development,* Paris, France: Austrian Computer Society, 2005, pp. 47-61.

[3] E. Carmel and P. Tjia, *Offshoring Information Technology: Sourcing and Outsourcing to a Global Workforce,* Cambridge, NY: Cambridge University Press, 2005.

[4] E. Carmel and S. Sawyer, "Packaged software development teams: what makes them different," *Information Technology & People,* vol. 11, pp. 7-19, 1998.

[5] I. Sommerville, *Software Engineering.* ,8th ed.Harlow, Essex, UK: Addison-Wesley, 2006.

[6] R. Stevens, P. Brook, K. Jackson and S. Arnold, *Systems Engineering Coping with Complexity* London, U.K.: Prentice Hall, 1998.

[7] S. Lauesen, *Software Requirements – Styles and Techniques.,* 2nd ed. Reading, MA.: Addison-Wesley, 2002.

[8] N. Power and T. Moynihan, " A theory of requirements documentation situated in practice," *Proceedings of the 21st Annual International Conference on Documentation,* ACM Press, New York, 2003, pp. 86-92.

[9] A. I. Wasserman, "Toward a Discipline of Software Engineering," *IEEE Software,* vol. 13, pp. 23-31, 1996.

[10] S. L. Pfleeger and J. M. Atlee, *Software Engineering Theory and Practice.*, 3rd ed. Upper Saddle River, NJ.: Pearson Education Limited, 2005.

[11] F. Rafii, "How Important is Physical Collocation to Product Development Success?" *Business Horizons,* pp. 78-84, January-February 1995.

[12] C. R. B. De Souza, (2001) "Global Software Development: Challenges and Perspectives " [online] 2001. Available: http://citeseer.ist.psu.edu/cache/papers/cs/22967/http:zSzzSzwww.ics.uci.eduzSz~cdesouzazSzCourseszSzGlobalSoftwareDevelopment.pdf/global-software-development-challenges.pdf [Accessed 1/3/2006]

[13] R.E. Grinter, J.D. Herbsleb and D.E. Perry, "The Geography of Coordination: Dealing with Distance in R&D Work," *Proc. Int'l ACM SIGGROUP Conf. Supporting Group Work (GROUP '99),* ACM Press, New York, 1999, pp. 306-315.

[14] J. R. Galbraith, *Organization Design* Reading, Ma.: Addison-Wesley, 1977.

[15] M. B. Miles and A. M. Huberman, *Qualitative Data Analysis: An Expanded Sourcebook.*, 2nd ed. Thousand Oaks, CA.: Sage, 1994.

[16] R. K. Yin, *Case Study Research: Design and Methods*, 2nd ed. Thousand Oaks, CA.: Sage Publications Inc., 1994.

[17] J. C. Flanagan, "The critical incident technique," *Psychological Bulletin,* vol. 51, pp. 327-355, 1954.

[18] C. Koh, S. Ang and D. W. Straub, "IT Outsourcing Success: A Psychological Contract Perspective," *Information Systems Research,* vol. 15, pp. 356-373, 2004.

[19] L. Gundry and D. M. Rousseau, "Communicating culture to newcomers," *Human Relations,* vol. 47, no. 9, pp. 1065-1088, 1994.

[20] R. K. Merton and P. L. Kendall, "The Focused Interview " *The American Journal of Sociology,* vol. 51, pp. 541-557, 1946.

[21] M.Q. Patton, *Qualitative Evaluation and Research Methods*, 2nd ed. Newbury Park, CA.: SAGE, 1990.

[22] J. D. Herbsleb and R. E. Grinter, "Splitting the Organization and Integrating the Code: Conway's Law Revisited," *Proc. 21st Int'l Conf. Software Eng. (ICSE '99),* ACM Press, New York, 1999, pp. 85-95.

[23] J. Pyysiäinen, J. ''Building Trust in Global Inter-Organizational Software Development Projects: Problems and Practices," in *International Workshop on Global Software Development (co-located with ICSE 2003, International Conference on Software Engineering),* Portland, OR., May, 2003, pp. 69-74.

[24] N. Flor, "Globally distributed software development and pair programming," *Commun. ACM* vol. 49, no. 10, pp. 57-58, Oct 2006.