Global Software Development in Practice Lessons Learned

Rafael Prikladnicki
School of Computer Science
Pontifícia Universidade Católica
do Rio Grande do Sul - PUCRS
Avenida Ipiranga, 6681
Porto Alegre – RS – Brazil
+55 51 3320.3558
rafael@inf.pucrs.br

Jorge Luis Nicolas Audy
School of Computer Science
Pontifícia Universidade Católica
do Rio Grande do Sul - PUCRS
Avenida Ipiranga, 6681
Porto Alegre – RS – Brazil
+55 51 3320.3558
audy@inf.pucrs.br

Roberto Evaristo
College of Business
Administration
University of Illinois at Chicago
601 S. Morgan Street MC 294
Chicago, IL 60607, United States
+1 312 996 8415
evaristo@uic.edu

ABSTRACT
More than a decade ago organizations began to experiment with remotely located software development facilities seeking lower costs and access to skilled resources. This change is having a profound impact not only on marketing and distribution but also on the way products are conceived, designed, constructed, tested, and delivered to customers. The number of organizations distributing their software development processes worldwide keeps increasing. As a result, software development is becoming a multi-site, multicultural and globally distributed undertaking. More recently, attention has turned toward trying to understand the factors that enable multinationals and virtual corporations to operate successfully across geographic and cultural boundaries. Based on these factors, we present lessons learned from case studies in two software development units from multinational organizations located in Brazil.

Keywords
Software engineering, software development process, distributed software development, global software development, offshore outsourcing, offshore insourcing

1. INTRODUCTION
Software has become a vital component of almost every business. Success increasingly depends on using software as a competitive advantage (Carmel, 1999). More than a decade ago, many organizations began to experiment with remotely located software development facilities seeking lower costs and access to skilled resources. Economic forces are relentlessly turning national markets into global markets and spawning new forms of competition and cooperation that reach across national boundaries. Several factors have contributed to build this scenario (Herbsleb, 2001):

- The business market proximity advantages, including knowledge of customers and local conditions;
- Pressure to improve time-to-market by using time-zone differences in "round-the-clock" development;
- The need to have a global resource pool to successfully and cost-competitively have resources, wherever located.

This change is having a profound impact not only on marketing and distribution but also on the way products are conceived, designed, constructed, tested, and delivered to customers. The number of organizations distributing their software development processes worldwide aiming at heightened profit and productivity as well as cost reduction and quality improvements keeps increasing. As a result, software development is becoming a multi-site, multicultural, globally distributed undertaking. Engineers, managers, and executives face formidable challenges on many levels, from the technical to the social and cultural.

Considering these factors and the global scenario, it is reasonable to believe that there will be continuing pressure toward the adoption of globalized approaches to software development (Carmel, 1999). Although many organizations have faced difficulties in their experience with global software development (GSD), others experienced large benefits with geographic dispersion.

The objective of this study is to understand problems organizations have faced when going global in software development, as a result of a two-year-long study in multinational organizations in Brazil. Two case studies were conducted identifying some of the difficulties, solutions, and critical success factors of distributed software development.

Our contributions are the lessons learned from the case studies as a first step to propose a reference model aiming to minimize problems in global software development projects. Although the main title suggests global development, the topics in this paper apply also to most distributed software development environments, even those across town.

This paper has the following structure: section 2 presents the theoretical base; section 3 describes the research method; section 4 describes the case study; section 5 discuss the results found in the case study and presents the lessons learned; section 6 presents the final remarks, future studies and the research limitations.

2. THEORETICAL BASE

2.1 Global Software Development (GSD)

Software process is defined by a set of activities, methods, practices and technologies that people and companies use to develop and to keep related software and products (Pressman, 2001). The interest in the software process is based on the following premises:

- The software quality is strongly dependent on the quality of the process used in its preparation;
- The software process can be defined, managed, measured and improved.

It is not a simple task to develop software, even when using a well-defined development process. As part of the globalization efforts currently pervading society, software project teams have also become geographically distributed on a worldwide scale. This characterizes Global Software Development (GSD). Any software professional knows that even "normal" –let alone "global" – software development is fraught with difficulties (Carmel, 1999). Based on widely available and varying estimates, perhaps half of all system projects are failures.

The list of features that distinguish global software development from normal (centralized) is short and precise: distance (the distance of developers from each other and from their customers or end-users); time-zone differences (time zone is to a large extent a confounding factor with distance); and national culture (including language, national traditions, customs, and norms of behavior) (Carmel, 1999). Distance, time zone and national culture have effects on many levels:

- **Strategic Issues**: The decision whether a particular project should be developed by globally dispersed teams – and where it can be better developed, as well as how to divide it across sites – is difficult. Some analysis considering the risk and benefit of project dispersion can be necessary. Solutions are constrained by resources available at the sites, their levels of expertise in various technologies, infrastructure, etc. A number of models are possible and appropriate under different circumstances (Herbsleb, 2001).
- **Cultural Issues**: GSD require close cooperation of individuals with different cultural backgrounds. Cultures differ on many critical dimensions, such as national, ethnic, organizational, professional, technical, and team culture (Carmel, 1999). While many people find such differences enriching, they can also lead to serious and chronic misunderstanding. For instance, e-mail, from someone in a culture where communication tends to be direct might seem abrupt or even rude to someone from a different background. Culture differences often exacerbate communication problems (Marquardt, 2001).
- **Knowledge management**: without effective information and knowledge-sharing mechanisms, it is difficult to exploit the benefits of GSD. Poor documentation can also cause ineffective collaborative development. The resistance to documentation among developers is well known, but in GSD environments the value of documentation is very important to clarify unspoken assumptions and ambiguity and to support maintainability (Karolak, 1998).
- **Technical issues:** when teams are working across sites, the lack of synchronization can be particularly critical. There is a need to assure commonly defined milestones and clear entry and exit criteria for all tasks. Typically, risk management does not take into account possible impacts of the dispersion, diverse cultures, time and attitudes (Evaristo, 2003). The overhead of control and coordination associated with any software projects is astounding. Because of distance, people cannot coordinate by peeking around the cubicle wall, nor can managers control by strolling down the hall and visiting the team's office. Furthermore, time-zone differences impede a quick phone call to resolve an issue or clarify an algorithm on the fly. Since networks spanning globally dispersed locations are often slow and unreliable, tasks such as configuration management that involve transmission of critical data must be meticulously planned and executed. The need to control product changes and to ensure that all concerned hear about them is much greater in GSD (Carmel, 1999).

Tools and technological environments have been developed over the last few years to help in the control and coordination of the development teams working in distributed environments. Many of these tools are focused in supporting procedures of formal communication such as automated document elaboration, processes and other non-interactive communication channels.

The traditional problems and the existing challenges in GSD can be summarized as follows. Organizations search for competitive advantages in terms of cost, quality and flexibility in the area of software development (Prikladnicki, 2002),

looking for productivity increases as well as risk dilution (McConnel, 1996). Many times the search for these competitive advantages forces organizations to search for global solutions, where we have two main options currently under use: offshore outsourcing (contracting services with an external organization located in another country), and offshore insourcing (contracting with a wholly owned subsidiary also located in another country).

2.2 Related Work

Global software development causes a profound impact on the way the products are conceived, designed, constructed, tested, and delivered to customers (Herbsleb, 2001). Thus, the structure needed to support this kind of development is different from the one used in collocated environments. The literature mentions reference models for global software development with emphasis, on the global team. These studies consider both technical and non-technical factors. In the following sections we present three of these studies.

2.2.1 Karolak, 1998

Karolak (Karolak, 1998) proposes a model for global software development following the traditional project life cycle: pre-development, requirements, design, code, test, customer delivery and maintenance. It begins with the description of necessary steps to set up the development environment and project team, and how to effectively manage intellectual property, followed by a discussion of the differences between traditional (management of a solely in-house project) and virtual management. The model also includes requirements traceability, communication infrastructure, software configuration management, risk management, documentation, test, and software quality.

2.2.2 Carmel, 1999

Carmel (Carmel, 1999) proposes a model of software development for global teams. He sees software globalization as a centrifugal force that propels things outwards from the center as it disperses developers to the far corners of the world. A centrifugal force must be balanced by centripetal force, a counter force that is directed into the center. Centrifugal is derived from the Latin "to flee the center", and centripetal, from the Latin "to seek the center". The five centrifugal forces, or the problems, pull the global software team apart and inhibit its performance (Figure 1).

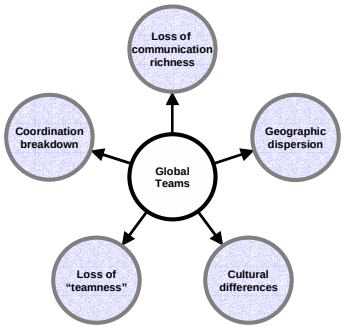

Figure 1. Centrifugal forces (Carmel, 1999)

The first centrifugal force is geographic dispersion. The next three forces build on the problem of distance: loss of communication richness, coordination breakdown, and loss of "teamness". The last force is cultural differences and culture breakdowns inside global teams. The six centripetal forces, or the solutions, pull the global software teams together and make it more effective (Figure 2).

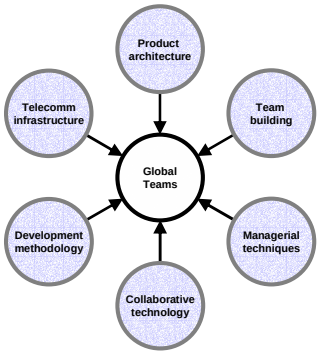

Figure 2. Centripetal forces (Carmel, 1999)

Telecommunication infrastructure is the foundation for all the other strategies. Collaborative technologies hold it all together. Development methodology and product architecture needs to be conceived carefully. Team building is the human resources effort and finally, managerial techniques are focused on global managers.

2.2.3 Evaristo, 2003

Evaristo (Evaristo, 2003) suggests dimensions to the concept of "distributedness" through a theory-based model. These dimensions are related not only to software development projects, but also to more general distributed projects (Figure 3).

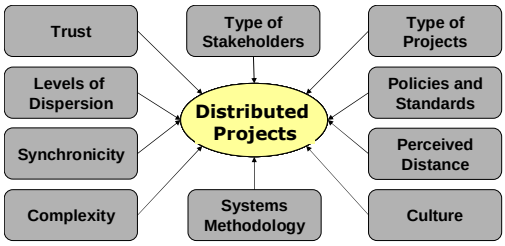

Figure 3. Dimensions for distributed projects (Evaristo, 2003)

The main objective is to understand what "distributed" means when discussing the management of distributed projects and to suggest better ways to manage distributed projects by finding out what the critical problems in "distributed" projects are.

2.2.4 Critical Analysis

The approaches described previously consider global projects in three different perspectives. While Karolak (Karolak, 1998) propose a specific model for global software projects, guided by traditional project life cycle, Carmel (Carmel, 1999) consider the global teams, their characteristics and the main challenges to have success in this scenario. Finally, Evaristo (Evaristo, 2003) discusses a more general type of project. These approaches present essential characteristics of global software development. Many other authors have been studied these characteristics (Damian et al., 2002; Sarma, 2002; Kiel, 2003; Lanubile, 2003) expanding the concepts and developing specific studies with each characteristic. Different models search for strategies to manage such projects and we can see many improvements in tools and methods over the last decades. In spite of that, organizations are experiencing many difficulties. The main motivation of this study is to try to understand these problems as well as the solutions adopted.

3. RESEARCH METHOD

This research is exploratory in nature and based on case studies. Qualitative methods are appropriate to study the system development process in its real context, with description and the understanding of the state of the art in those situations where practice precedes theory (Yin, 1994).

The case studies were developed in two software development units, each one owned by a multinational organization with worldwide units. The organizations were selected considering their size, the existence of a formal and documented process and their recognition as a SW-CMM[1] level 2 organization.

The first organization works mainly in consulting, software development projects and training. It has nine software development units located in Brazil. The organization also has offices located in Brazil and Latin America as well as in the U.S. and Europe. Its headquarters are located in Brazil. The second organization supports and manufactures computers. It has three software development units located in two continents that are responsible for **internal client demand** worldwide. Its headquarters are located in the U.S.

The data collection was constituted of primary and secondary sources. We conducted 22 individual interviews (11 in each organization). Secondary sources were also used: document reviews, strategic mission analysis, business process, meetings minutes, and software development process description beyond access to the homepage of both organizations. For data analysis a content analysis was developed, with stability test (Krippendorff, 1980).

We developed two questionnaires (Appendix 1), each considering a specific dimension to be explored: "organizational," containing information about the organization as a whole, and "project," with information on the four projects included in this study. One development manager was interviewed in each organization, whereas five interviews were conducted for each project. Our convenience sample was not probabilistic although we looked for a good representation of all groups involved (Table 1).

[1] SW-CMM is one of the CMM models used for software engineering organizations (http://www.sei.cmu.edu).

Table 1. Interviewee profile

Role	Qty	Dimension	Org.
Development Manager	2	Organizational, Project	1, 2
Project Manager	4	Project	1, 2
Technical Leader	4	Project	1, 2
Developer	4	Project	1, 2
SQA representative	1	Project	1
SEPG representative	1	Project	2

The respondents were chosen according to the unit of analysis and the study purpose. Therefore, we interviewed project team members, development managers, quality assurance team members (SQA), software process improvement responsible (SEPG[2]) and individuals representing the organization strategic level, all of them located in Brazil.

4. CASE STUDY

The case studies were conducted through 11 interviews in each organization. Document review was performed, and other interviews were conducted as needed to clarify ambiguities. In the next sections we present specific details of each organization as well as the consolidated results.

4.1 Organization 1

This case study was developed in the organization headquarters, in a city located in the southeast of Brazil, where the main software development unit is also located. It has 80 collaborators working in software development and all clients are external to the organization. Their software development process is based on RUP (Rational Unified Process) and PMI (Project Management Institute). The unit studied is certified as ISO 9001[3] since 1996 and recognized as a SW-CMM level 2 organization since 2002.

When asked about the reasons to invest in distributed and global software development, the individual representing the strategic level of this organization pointed out the following items:

- Cost reduction, quality focus, competitiveness;
- The creation of centers of competence, having each unit with specialized in specific skills and/or technology;
- Global standard of software development;
- To have competitive costs, wherever located.

4.1.1 Defining the two projects evaluated

The purpose of **Project A** was to develop an application for a large company located in the U.S. The project was managed both by the company in the U.S. and by the branch office located in Brazil. According to the classification proposed by Prikladnicki, Audy, and Evaristo (Prikladnicki, 2003), project team, customers and users were related in the following way:

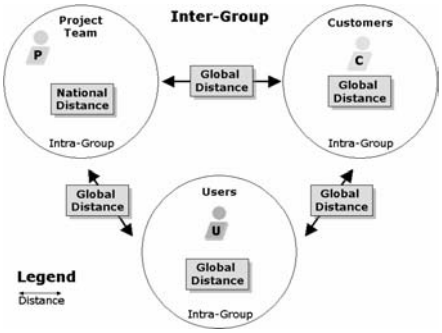

Figure 4. Project A

[2] The SEPG is a group of specialists who facilitate the definition, maintenance, and improvement of the software process used by the organization.

[3] ISO 9001 is the international standard for assessing quality systems to ensure process consistency and predictability (http://www.iso.ch).

The project team was located in two different offices in Brazil, while customers were located both in Brazil and in the U.S. The users were all company employees.

The objective of **Project B** was to develop an application for a bank in São Paulo. The bank contracted the job to a third-party company, which in turn subcontracted the software development to the unit that we studied. Therefore, the bank was the user, and the third-party company contracted by the bank was the customer, acting sometimes as part of the project team. And the unit studied was the project team. According to the classification proposed (Prikladnicki, 2003), the stakeholders were related this way:

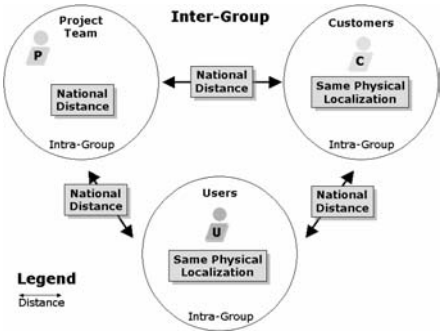

Figure 5. Project B

The project team, users and customers were located in Brazil. But the project team was distributed in two different cities, while customers and users were in the same physical location. Therefore, they were separated from part of the project team. The other part of the project team was inside the bank.

4.2 Organization 2

The case study was developed in the software development unit in a city located in the south of Brazil. This center aims to perform worldwide technological development for the organization. Almost all projects are distributed, mainly global, since customers and users are located in offices around the world. It has 180 collaborators working in software development and all clients are internal to the organization. The software development process is based on the MSF (Microsoft Solutions Framework) and also on known methodologies, like RUP, and PMI. The unit was SW-CMM certified on level 2 since 2003.

The individual representing the strategic level of this organization presented the following reasons to invest in distributed and global software development:

- Cost reduction;
- Expanding strategy to global markets;
- Consolidate the organization trademark outside the U.S.;
- Global standard of software development.

4.2.1 Defining the two projects evaluated

Project C involved the development of an application to manage talent to be used by the global human resources department. According to the classification proposed (Prikladnicki, 2003), the stakeholders were related this way:

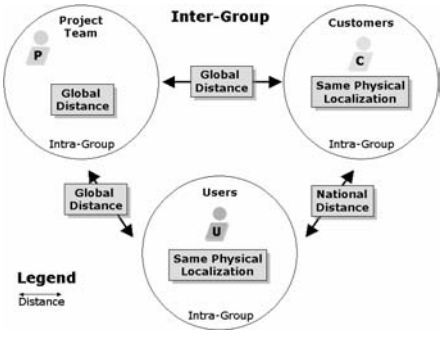

Figure 6. Project C

The project team was located in Brazil and in the U.S., while customers (human resources department) were located in the U.S. unit in the same physical location. The users were also located in the U.S., in the same physical location, but non-collocated with the customers.

The purpose of **Project D** was to develop an application for the organization manufacturing area. According to the classification proposed (Prikladnicki, 2003), the stakeholders were related this way:

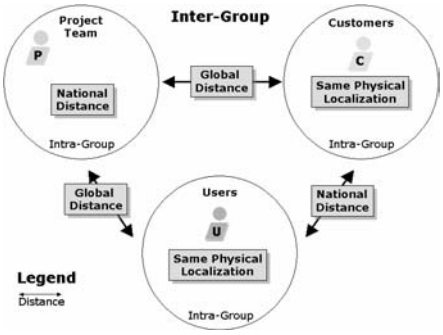

Figure 7. Project D

The project team was dispersed, but located in Brazil. Customers and users were located in the U.S., each one in the same physical location, but dispersed.

4.3 Case Study Results

Both organizations were involved with globally distributed projects. We found empirically factors that were theoretically predicted, and consolidated the combined learning under "lessons" below. We concluded that difficulties, solutions and critical success factors involve three dimensions: technical, non-technical and hybrid (both technical and non-technical). Technical factors are those related to technical knowledge used in the software development, whereas non-technical (social, cultural, languages, behavior, and so on) the ones that are composed by knowledge of related areas needed to the software development activity. The hybrid factors include both technical and non-technical knowledge. The next sections present the consolidated case study results (we will use CS1 for the case study in the organization 1 and CS2 for the one in the organization 2).

4.3.1 GSD difficulties

According to the interviews, the difficulties with GSD are related to the lack of standards in the activities between distributed teams, the difficulty of share information and the lack of a well-defined software development process. As corroborated by Carmel (Carmel, 1999), and Evaristo (Evaristo, 2003), difficulties concerning language barriers and communication, cultural differences, context sharing and trust acquisition between teams (Table 1) were also found.

Table 2. GSD difficulties found

GSD difficulties	Dimension	Source
Requirements engineering	Technical	CS1, CS2
Software development process	Technical	CS1, CS2
Software configuration	Technical	CS1
Knowledge management	Technical	CS1
Communication and language	Non-technical	CS1, CS2
Culture and context sharing	Non-technical	CS2
Trust	Non-technical	CS1, CS2

Requirements engineering was a consistent difficulty, involving requirements elicitation, analysis, specification, validation, and management. Some interviewees mentioned the need to have a very detailed set of requirements to compensate the distribution across many sites.

The software development process was considered a large difficulty since sometime distributed teams are not using the same process. Software configuration is also a critical issue and the source of many problems (artifacts with different versions and content in each site).

The case studies identified the lack of a formal and consistent knowledge management process in both organizations, emerging as a great obstacle to sharing information. Trust was also a problem, mainly the necessity of a distributed trust

acquisition. Finally, communication and language misunderstandings were motivated by the cultural differences between the dispersed stakeholders.

4.3.2 Solutions to GSD difficulties

Although there are many possible solutions for each problem identified, the organizations focused their solutions mainly on the need for work standardization, investment in planning, process engagement and continuous risk management. Other solutions mentioned included the integration and ways to increase trust between global teams and continuous training, also mentioned by Evaristo (Evaristo, 2003). A formal software development process and an effective requirements engineering was mentioned and are also supported by Carmel (Carmel, 1999), Karolak, (Karolak, 1998), and Damian (Damian, 2002) (Table 2):

Table 3. Solutions implemented

Solutions	Dimension	Source
Formal Planning and engagement	Hybrid	CS1, CS2
Training	Hybrid	CS1, CS2
Standardization	Technical	CS1, CS2
Risk Management process	Technical	CS1
Software development process definition	Technical	CS1, CS2
Face-to-face requirements elicitation	Technical	CS1, CS2
Trust acquisition and integration	Non-technical	CS1, CS2

Initial planning was a necessity to select the projects to be distributed, evaluating its characteristics and the unit availability to work on them. Moreover, the process engagement plays an important role to start the interaction between distributed teams.

Another solution implemented was training in soft skills (non-technical factors). Some topics explored were leadership, communication, culture, context sharing, project management, and technical training. Standardization was adopted when the distributed teams were not using the same process. Three strategies were considered: forcing standardization; blending methodological components from the various sites into one "new" methodology; and imposing high-level guidelines.

Risk management was growing in both organizations. Some groups practiced traditional risk management, not taking into account the possible impacts of dispersion, the diverse cultures, time and attitudes, leading to the recommendation of the need for a distributed risk management process.

Both organizations are investing in face-to-face requirements elicitation, something dependent on project characteristics and travel limitations. There was a large effort to create formal approvals for artifacts in every project. Finally, integration activities are being conducted aiming at trust acquisition. Some of these activities are developed virtually, but most of them occur when teams (or part of it) meet face-to-face.

4.3.3 GSD Critical Success Factors

The critical success factors (CSF) identified are directly related to the organizational "modus operandi". For the same activity we can have different CSF related to the strategies adopted by each organization. Consolidating the results, we identified the following critical success factors (Table 3):

Table 4. Critical Success Factors

Critical Success Factors	Dimension	Source
Software Development Process	Technical	CS1, CS2
Training	Hybrid	CS1, CS2
Planning and Engagement	Hybrid	CS1, CS2
Infra-Structure	Technical	CS2
Team integration	Non-technical	CS1, CS2
Communication and Feedback	Non-technical	CS1, CS2

The existence of a formal software development process was considered one of the most important success factors for distributed projects. Other CSF included large investments in training resulting in improved relationships. The initial planning was important to evaluate distributed projects correctly and to select the proper unit to assign each project. Process engagement was considered a success factor because it was the first contact between teams in distributed projects.

Integration activities improved soft skills of individuals, increasing trust and minimizing cultural differences. Finally, integration improved the communication and feedback.

5. LESSONS LEARNED

The study conducted in both organizations shows many characteristics of GSD (section 4). In this section we will present the lessons learned.

Lesson 1: Project management and in particular risk management need additional effort and steps

According to the Project Management Institute (Project Management Institute, 2000), project management is the application of knowledge, abilities and techniques to plan activities that can reach the needs and expectations of all stakeholders involved in a project. Bad project management can mean the loss of the project and the resources involved. Therefore, risk management is one of the most important activities in a project, involving the identification, treatment and elimination of risk sources before it becomes a concrete threat for the project. Risks can also be treated in different levels.

It was found that all activities involving project management and risk management have extreme importance for distributed projects and almost all project managers interviewed said that in distributed projects these activities take longer than in traditional collocated projects, requiring larger effort and additional steps as compared to traditional models. For instance, one of the development managers interviewed said that "the culture is a risk for this kind of projects and can have a huge impact on attitudes and team behaviours. It is easier to notice and clarify misunderstandings in co-located situations, but in the case of distributed projects, however, cultural differences that may pass unnoticed magnify any problems that may occur because of the inherent difficulty in distributed environments".

Lesson 2: The existence of a well-defined software development process is responsible for many advantages in distributed projects

According to Pressman (Pressman, 2001), a well-defined process is a process with good documentation, detailing what is being done (product), when (steps), for whom (actors), the artifacts used (input) and the developed artifacts (output/results). Moreover, a life cycle must be selected as the starting point for any project.

The study showed that in both organizations all projects without a well-defined process experienced many problems, some of them related to process (requirements, configuration management, testing, etc.), and others inherited from previous problems with process, such as communication, synchronization and trust. Thus, a single and well-defined process in accordance with the project environment can be the solution for many difficulties in global development. For this reason, when cross-border sites are consolidated into a team, a good way is to consider one of three strategies: forcing standardization; blending methodological components from the various sites into one "new" methodology; or imposing high-level guidelines.

One of the technical leaders interviewed mentioned that "there are a lot of issues that need to be considered from the beginning of distributed projects. Things like coding standards, project life cycle, who will be the code owner, how the development will be conducted, and if the distributed team will work in the same or in different modules".

Also, a developer said that "even if the each team has a different process, an internal process must be created specifically for that project, defining standard tools to be used, activities flow, coding standards, etc, in order to manage the team expectation related to the process to be used".

Lesson 3: Knowledge management stimulates the information sharing and stimulates the learning from experience

The purpose of knowledge management is to absorb the organizational intellectual capital to use in the future Lindvall (Lindvall, 2002). But the main problem is that the intellectual capital is intrinsic to the human being. In software development organizations, this concept has been applied with the purpose of investing in learning from experience, that is, an individual can learn based on the experiences lived by other individuals, since all experiences are documented in a systematic way. Knowledge management can help in the decision process, increasing the quality and decreasing costs and project time (reuse), stimulating the developing of a consistent information repository to be used in the future.

Knowledge management also relates to information collection from projects. Projects generate many types of information that, if shared, can bring benefits for the teams and for the organization (Desouza, 2003). One of the consequences for distributed projects is the stimulation of learning from experiences shared between distributed teams. The interviews conducted indicated that such investment in knowledge management (tools or activities that stimulate information sharing) minimized many obstacles to GSD.

One of the project managers interviewed said that "despite of the existence of historical data from all projects in the organization, there is not a culture of using this information in an efficient way, because people are not able to use, don't want to use or simply are not trained to use the benefits of an information repository".

Lesson 4: Requirements engineering is the main challenge for the software development process point of view

Requirements engineering plays an important role in the software development. A requirement is the condition or capacity that a system that is being developed must satisfy (Oberg, 2000). Therefore, the compliance with requirements determines the project success or failure. Requirements are identified, registered, organized and verified during the project development, and are essential to keep the "contract" among project team, users and customers.

The problems related with requirements engineering are one of the main reasons for software projects failures (Oberg, 2000). Research has identified (Oberg, 2000) that 70% of the requirements were difficult to identify and 54% were not clear and well organized. Therefore, it is not difficult to find errors in requirement specifications with a resulting large impact in the project costs. It is clear that the earlier a problem is detected and solved (especially during the requirements phase) the earlier other problems are minimized in the following phases (Damian, 2002). Damian, and Zowgui (Damian et al., 2002) conducted a study where the main challenges on managing requirements in a multi-site organization were identified. The study identified four major problems of geographical distribution (inadequate communication, knowledge management, cultural diversity, and time differences). These problems created specific difficulties conducting requirements engineering, like managing conflict, common understanding of requirements, effective meetings, delay, etc.

In the same lines as Damian's study, almost all project managers and technical leaders interviewed in our research pointed out difficulties related to requirements engineering activities. The main problem of one of the projects was requirements instability, mainly because the distance between teams, compromising understanding and agreement between parties. In all projects requirements identification was a challenge, involving activities like meetings, requirements documentation as soon as defined, traceability, requirements control and management. One of the development managers interviewed said that "a good approach to minimize problems with the distributed requirements engineering was to conduct as many meetings as necessary to understand all requirements, and to document all meetings in detail, in order to get the acceptance of all people involved".

Lesson 5: The planning phase is important to organize and manage the distributed projects properly

The definition of strategies in information systems based on a formal planning process is a challenge (Audy, 2001). The lack of a formal planning phase can be one of the main problems in the software development process. According to Martin (Martin, 1991), the lack of a formal planning phase causes a great number of problems in the next phases.

We found that the initial planning was a formal phase to decide if a project should be distributed and to plan its development. Thus, planning involves the definition of the strategies that will lead the development of the whole process.

The SEPG representative mentioned that "a planning phase is very important for distributed projects, because we are able to select the proper projects to be developed by distributed teams, and also to define the better way to engage the teams in the project". Additionally, a project manager said that "when you are working in a distributed project, if you don't have a formal planning phase before the project starts, there is a greater possibility to discover risks not considered when the decision was made to distribute the project development. Theses risks can range from security constraints and physical environment availability, requirements clarity and complexity, experience in distributed projects, to other technology risks."

Lesson 6: The investment in recruiting and training global teams can minimize the difficulties related to the non-technical dimension

In global development, project managers have to organize and manage projects with a team composed by individuals from different cultures, with different customs. According to Kiel (Kiel, 2003), the technical barriers are diminishing rapidly. On the other hand, the human factors are less studied. Therefore, when distribution ultimately fails, it can be a web of social, cultural, linguistic and political factors, rather than use or misuse of specific tools or techniques (Kiel, 2003). There are other factors that can be added to this list (communication, context, interpersonal relationship), but this study brought up a very important conclusion. The lack of investment in the recruiting and training of project teams to become global teams can lead to unexpected problems in project development.

The policies of Organization Two included investing in team training, focusing on communication, cultural differences, trust, and context sharing. As a result of this initiative, the interactions between distributed teams (including customers, users and project team) became more effective. Problems identified before the training started to occur less frequently, showing that the management of distributed teams is a key to the project success.

The importance of training can be reinforced by a project manager, who said that "we will always have problems when working with people from different sites. We will have problems even with people from the same site. But in a global scenario this is even more critical due to the geographical and time dispersion. For this reason, we are investing in minimizing the impact of these differences, investing in training to be effective working with global teams. And this training is conducted with all people as soon as they are onboard".

Lesson 7: Tools can act as a facility in the distributed interaction

Theory suggests that some problems can be solved using specific tools to support collaboration, cooperation, knowledge management, and requirements engineering, for example Sarma (Sarma, 2002), and Lanubile (Lanubile, 2003). The use of tools to support the global development depends on the characteristics of each team, since people are different from each other, and these differences can show in tool use.

We found that both organizations have strategies to work with global tools, aiming at global knowledge management and global integration. Moreover, tools to support communication, like e-mail, video conferencing, teleconferencing, and chat are frequently used.

A project manager interviewed said that "there is a tool aiming at the global knowledge management in the organization, and the idea to develop this tool appeared in a meeting with all employees. Thus, the tool was developed as an internal project and now we expected this tool to help in information sharing".

Lesson 8: Distributed Software Development is a maturity process

According to Paulk (Paulk, 1993), maturity means the state or condition of full development, state or quality of mature. When the CMM model was proposed, it was structured and developed in a way that could be represented in different levels of maturity. Moreover, each level has a set of practices and standards as a guide to the process improvement. After the CMM framework, other maturity models have emerged in different areas for different activities.

The data showed a clear difference between the maturities of both organizations related to the distributed software development. Organization 1 was working with distributed projects for at least four years, while organization 2 was doing the same for only one year. Thus, the organization 1 was experiencing different and more complex problems compared with the organization 2. Based on our data, the development of distributed projects is something the needs time to mature.

In GSD, the maturity concept is applied to global teams and to the organization on a strategic level (maturity levels for offshore development, for example). For this reason, the maturity of distributed projects must consider not only the maturity of distributed teams, but also other factors, complementing the GSD theory.

6. FINAL REMARKS

It is becoming harder to justify completing a software development project inside company walls. As the software community appreciates the economy of merging diverse development skills and domain expertise and as communication media become more sophisticated, the cost and technology pressures push more organizations toward global software development. It is becoming less and less cost-effective or competitive to develop a software product in the same building, company, or even country.

Although software development engineering is still far from a mature discipline, improvements in tools and methods over the last several decades are allowing groups from different locations and backgrounds to come together as a global software development team. Moreover, GSD is leading the researchers to acquire new knowledge and to be more interdisciplinary.

This paper advances the knowledge in the GSD area by identifying important characteristics of this recent and growing field. Lessons were learned based on case studies in two software development units from multinational organizations located in Brazil. This study enabled a better understanding of GSD and the relationship between the project team, customers and users. These results give us indication that the search for greater formalism and the selective utilization of international patterns will provide full conditions to overcome the problems originating from the dispersion.

There is no need to tell experienced managers that it is better to manage groups of people who are co-located rather than dispersed. Co-location allows managers to manage by observation but has some disadvantages like informal oral communication. Despite that, an advantage of dispersion is the innovation of smaller, more independent groups and more formalism in the software process. The main consequence of such software development is the reliance on asynchronous communication techniques, primarily e-mail. Problems that should have been simple to resolve many times can take days. Furthermore, the conversation through e-mail can introduce greater opportunities for misunderstanding, particularly when the content of the communication is argumentative. The challenge in GSD is to create strategies, techniques and practical lessons from experience to alleviate such problems.

Planned follow up studies in this topic will delve in the study of specific factors like requirements engineering, risk management and project allocation.

7. ACKNOWLEDGMENTS

We would like to thank the organizations that accepted to be part of this study. The study was funded in terms of the Brazilian Federal Law for Information Technology (Law No. 8.248/91).

8. REFERENCES

Audy, J. L. N. 2001. Strategic Planning Model of Information Systems: contributions of decision process and organizational learning (in Portuguese)". *Ph. D. Tesis*, PPGA – UFRGS, Brazil.

Carmel, E. 1999. *Global Software Teams – Collaborating Across Borders and Time-Zones*. Prentice Hall, USA.

Damian, D. 2002. The study of requirements engineering in global software development: as challenging as important. *Proceedings of International Workshop on Global Software Development at ICSE*, Florida, USA.

Damian, D., Zowgui, D. 2002. The Impact of stakeholders' geographical distribution on managing requirements in a multi-site organization, *Proceedings of International Conference on Requirements Engineering*, California, USA.

Desouza, K., Evaristo, R. 2003. Global Knowledge Management Strategies. *European Management Journal*, v. 21, n.1, 62-67.

Evaristo, J. R., Scudder, R., Desouza, K. and Sato, O. 2003. A Dimensional Analysis of Geographically Distributed Project Teams: A Case Study. *Journal of Engineering Technology and Management*: v. 11, n. 4, 58-70.

Herbsleb, J. D., Moitra, D. 2001. Global Software Development, *IEEE Software*, v.16, n.2, 16-20.

Karolak, D. W. 1998. *Global Software Development – Managing Virtual Teams and Environments*. Los Alamitos, IEEE Computer Society, USA.

Kiel, L. 2003. Experiences in Distributed Development: A Case Study. *Proceedings of International Workshop on Global Software Development at ICSE*, Oregon, USA.

Krippendorff, K. 1980. *Content analysis: an introduction to its methodology*. Sage. California.

Lanubile, F. 2003. A P2P Toolset for Distributed Requirements Elicitation. *Proceedings of International Workshop on Global Software Development at ICSE*, Oregon, USA.

Lindvall, M., Rus, I. 2002. Knowledge Management in Software Engineering. *IEEE Software*, v.19, n.3, 26-38.

Marquardt, M. J., Horvath, L. 2001. *Global Teams: how top multinationals span boundaries and cultures with high-speed teamwork*. Davies-Black Publishing. Palo Alto, USA.

Martin, J. 1991. *Information Engineering (in Portuguese)*. Rio de Janeiro, Campus.

McConnel, S. 1996. *Rapid Development*. Microsoft Press, Canada.

Oberg, R., Probasco, L., Ericsson, M. 2000. Applying Requirements Management with Use Cases. *Rational Software White Paper*, Cupertino, CA, USA.

Paulk, M. C., Curtis, B., Mary, B. C., Weber, C. V. 1993. *Capability Maturity Model for Software*, Version 1.1. Software Engineering Institute. Technicak Report, 1993.

Pressman, R. S. 2001. *Software Engineering: A Practitioner's Approach*. Fifth Edit, McGraw-Hill, USA.

Prikladnicki, R., Peres, F., Audy, J., Móra, M. C., Perdigoto, A. 2002. Requirements specification model in a software development process inside a physically distributed environment. *Proceedings of ICEIS*, Ciudad Real, Spain.

Prikladnicki, R., Audy, J., Evaristo, R. 2003. Distributed Software Development: Toward an understanding of the relationship between project team, users and customers. *Proceedings of ICEIS*, Angers, France.

Project Management Institute. 2000. *A guide to the project management body of knowledge (PMBOK guide)*. USA.

Sarma, A., Hoek, A. 2002. Palantír: Increasing Awareness in Distributed Software Development. *Proceedings of International Workshop on Global Software Development at ICSE*, Florida, USA.

Yin, R. K. 1994. *Case study research: design and methods*. Sage. USA.

Appendix 1

Questionnaires

Questionnaire 1: Organizational Dimension

Demographic Data

Personal Identification

Name: _____ **Age:** _____ **years old**

Scholarship **(only the high grade)**

☐ **School** ☐ **High School** ☐ **College Incomplete** ☐ **College Complete**

☐ **Specialization** ☐ **Master Course / MBA Incomplete** ☐ **Master Course / MBA Complete**

Please fill the information below based on the scholarship selected above:

Course: _____

University: _____

Conclusion date: _____

Experience in computer science**:** _____ **years**

Organization: ☐ ***Organization 1*** ☐ ***Organization 2***

How long have you been in the organization?

☐ **Less than 6 months** ☐ **From 7 to 12 months** ☐ **From 1 to 2 years** ☐ **From 2 to 5 years**

How are you related to the organization?

☐ **as an Employee** ☐ **as an Outsourcing Service** ☐ **as an Intern**

Role**:** _____

(Following the role used in the organization)

Strategic References

1. What is the organization main business?
2. What is the mission of the organization?
3. What are the policies of the organization?
4. What are the organization main clients?
5. Why the organization adopts strategies of offshore outsourcing of software development process?

Business Process

6. What are the main business processes in the organization?

Organizational Structure

7. What is the general organizational structure?
8. What is the organizational structure specifically for software development?

Software Development Process

9. Is there a formal software development process being used by the organization? (Methodology, life cycle, activities)
10. What are the types of projects supported by the software development process?
11. What is the team structure in a software project (roles, responsibilities)?

Questionnaire 2: Project Dimension

Demographic Data

Personal Identification

Name: _____ **Age:** _____ **years old**

Scholarship **(only the high grade)**

☐ **School** ☐ **High School** ☐ **College Incomplete** ☐ **College Complete**

☐ **Specialization** ☐ **Master Course / MBA Incomplete** ☐ **Master Course / MBA Complete**

Please fill the information below based on the scholarship selected above:

Course: _____

University: _____

Conclusion date: _____

Experience in computer science**:** _____ **years**

Organization: ☐ ***Organization 1*** ☐ ***Organization 2***

How long have you been in the organization?

☐ **Less than 6 months** ☐ **From 7 to 12 months** ☐ **From 1 to 2 years** ☐ **From 2 to 5 years**

How are you related to the organization?

☐ **as an Employee** ☐ **as an Outsourcing Service** ☐ **as an Intern**

Role**:** _____

(Following the role used in the organization)

Difficulties found

12. In your project, what were the main difficulties related to global software development?
13. In your experience in similar projects, what were the main difficulties related to global software development?
14. In what activities in the software development process each difficulty was found?

Solutions

15. For each difficulty, what was the solution?
16. How this solution was identified?
17. What was the impact of this solution in the project?

Critical Success Factors

18. Considering this project and your experience in similar projects in your organization, in your opinion, what are the main critical success factors for global software development?

General Comments

19. Considering your experience in software development, how do you compare the distributed versus the collocated development?
20. Do you have some additional comment to add to this interview?