39 A REFERENCE MODEL FOR GLOBAL SOFTWARE DEVELOPMENT[i]

Rafael Prikladnicki
School of Computer Science. PUCRS, BRAZIL, rprik@inf.pucrs.br
Jorge Luis Nicolas Audy
School of Computer Science, PUCRS, BRAZIL, audy@inf.pucrs.br
Roberto Evaristo
College of Business Administration, University of Illinois at Chicago, USA, evaristo@uic.edu

The objective of this paper is to propose a reference model for global software development, based on the results found in a case study conducted in two software development units from multinational organizations located In Brazil. Since the number of organizations distributing their software development processes worldwide keeps increasing, this change is having a profound impact on the way products are conceived, designed, constructed, tested, and delivered to customers. The focus of this study is to understand the factors that enable multinationals and virtual corporations to operate successfully across geographic and cultural boundaries. Lessons learned and reference model are presented. Moreover, part of this model is being implemented in one of the organizations studied, and a preliminary evaluation is made.

1. INTRODUCTION

Software has become a vital component of almost every business (Pressman, 2001). 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. 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. Software development is increasingly 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 (Herbsleb, 2001).

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. Many organizations have faced difficulties and misunderstandings in their experience with global software development (GSD).

This paper has as objective to understand which problems organizations have faced when going global in software development and how these problems have been addressed, proposing a reference model, as a result of a two years long study in two multinational organizations in Brazil. Two case studies were conducted identifying difficulties, solutions, and critical success factors of distributed software development. The results are analyzed and the existing challenges identified. Some of the solutions being implemented are presented. Our contributions are the lessons learned from the case studies and the reference model proposed. Although the main title suggests global development, the topics in this paper apply to most distributed software development environments, even those across town.

2. THEORETICAL BASE

2.1 Global Software Development

As said before, many organizations began to experiment with remotely located software development facilities (distributed software development). Several factors have contributed to build this scenario, such as the business market proximity advantages, the pressure to improve time-to-market by using time zone differences in "round-the-clock" development, and the need to have a global resource pool to cost-competitively have resources, wherever located (Herbsleb, 2001).

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 (Karolak, 1998). Many of these tools are focused in supporting procedures of formal communication such as automated document elaboration, processes and other non-interactive communication channels.

Organizations search for competitive advantages in terms of cost, quality and flexibility in software development, looking for productivity increases as well as risk dilution (Prikladnicki, 2002). Many times the search for these competitive advantages forces organizations to search for global solutions (offshore software development). This epitomizes the traditional problems and the existing challenges.

2.2 Related Work

Global software development requires a structure involving different technologies and characteristics from the one used in collocated environments. A few studies have proposed reference models for global software development. In the following sections we present two of these studies.

2.2.1 The approach of Carmel, 1999

The author 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. The five centrifugal forces pull the global software team apart and inhibit its performance. The first force is geographic dispersion, something we know intuitively, that it is harder to manage from distance. Then there are three forces that 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.

A centrifugal force must be balanced by centripetal force, a counter force that is directed into the center, pull the global software teams together, and make it more effective. As centripetal force, there is telecommunication infrastructure, which the author sees as the foundation for all the other strategies. Collaborative technology is a force that holds it all together. Then we have development methodology and product architecture. Team building is the human resources effort and finally, managerial techniques are focused on global managers.

2.2.2 The approach of Evaristo, 2003

The author 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. The dimensions proposed are trust, levels of dispersion, type of stakeholders, type of projects, synchronicity, complexity, systems methodology, perceived distance, policies and standards, and culture. The main objective is to understand what "distributed" means when discussing the management of distributed projects and to suggest better ways to manage them by finding out what the critical problems in "distributed" projects are.

2.2.3 Critical Analysis

The approaches described previously consider global projects in two different perspectives. While Carmel (Carmel, 1999) consider the global teams, their characteristics and the main challenges to have success in global software projects, Evaristo (Evaristo, 2003) also talks about global teams, but related not only to software development, but also to a more general type of project.

Many other authors have been studied these characteristics, expanding the concepts and developing specific studies (Karolak, 1998), (Herbsleb, 2001), and (Morstead, 2003). Different models search for strategies to manage distributed projects and we can see many improvements in tools and methods over the last decades, allowing the GSD to happen. Despite that, since GSD is increasing, organizations are experiencing many difficulties. This has motivated studies based on industrial data, trying to understand these problems, and the solutions adopted.

3. RESEARCH METHOD

This research is exploratory in nature based on case studies (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 the recognition as a SW-CMM[ii] level 2 organization.

The data collection was constituted of individual interviews and, was also used secondary sources as complement, such as document reviews, and software development process description. We interviewed project team members, development managers, quality assurance team members, and software process improvement responsible, all defined according to the unit of analysis (projects) and the study purpose. Our convenience sample was not probabilistic although we

looked for a good representation of all groups involved. For data analysis a content analysis was developed, with stability test [10].

4. CASE STUDY

The case study was conducted in two software development units from multinational organizations located in Brazil. In the next sections we present specific information of each organization and the consolidated results.

4.1 Organization 1

The 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 known methodologies like RUP (Rational Unified Process) and PMI (Project Management Institute). The unit studied is certified as ISO 9001[iii] since 1996 and recognized as a level 2 organization in the SW-CMM model since 2002. It was interviewed people from two projects:

Project 1: the objective was to develop an application to 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. The project team was located in two different offices in Brazil, while customers were located both in Brazil and in the U.S. Some company employees worldwide represented the users.

Project 2: the objective 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.

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. It has 180 collaborators working in software development and all clients are internal to the organization. Considering the software development process, it is based on the MSF (Microsoft Solutions Framework), and also, on known methodologies, like RUP, and PMI. The unit studied is recognized as a level 2 organization in the SW-CMM model since 2003. It was interviewed people from two projects:

Project 1: the objective was to develop an application to manage talent to be used by the global human resources department. The project team was located in Brazil and the U.S., while customers (human resources department) were located in the U.S., in the same physical localization. The users were also located in the U.S., in the same physical localization, but dispersed from the customers.

Project 2: the objective was to develop an application for the organization manufacturing area. The project team was dispersed, but located in Brazil.

Customers and users were located in the U.S., each one in the same physical localization, 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. According to all interviews conducted in both organizations, the main GSD difficulties found were related to requirements engineering, software development process, software configuration, knowledge management, communication and language, culture, context sharing and trust.

For each difficulty identified, all respondents were also invited to describe solutions implemented to solve or at least to minimize the difficulties found. These solutions were related to planning and better engagement definition, training, standardization, risk management, software development process definition, trust acquisition, and requirements elicitation improvement.

Finally, some critical success factors were identified and were directly related to the organizational "modus operandi". All people interviewed were invited to share the main critical success factor when acting in global software development. The main points were the software development process, training, planning and engagement, infrastructure, team integration, 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: The project management, and in particular risk management need additional effort and steps.

In the study, all activities involving project management and risk management have a huge importance for distributed projects and almost all project managers interviewed said that in distributed projects these activities take longer than in traditional projects (collocated), requiring a larger effort and some additional steps in the traditional models.

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

The study showed that in both organizations, all projects without a well-defined process had many difficulties, some of them related to the process (requirements, configuration management, testing, etc.), and others inherited, 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.

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

The interviews conducted indicated that a great differential of global development is related to the investment in knowledge management (tools or activities that stimulates the information sharing), minimizing many difficulties. This was concluded based on the fact that project team members were not stimulated to share information based on the situations they were living every day

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

Project managers and technical leaders interviewed pointed out difficulties related to requirements engineering activities. One project had the requirements instability as the main problem, mainly because the distance between teams, compromising the understanding and agreement. In all projects it were identified the requirements as a challenge, involving the requirements documentation as soon as defined, meetings, traceability, requirements control and management.

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

The initial planning was identified as a formal and basic phase to decide if a project can be distributed and how to plan for its development. Thus, the planning basically involves the definition of the strategies leading to the development of the whole process. Based on the case studies, it is possible to consider the planning phase as a former cycle of many projects cycles derived from the planning process.

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

Organization 2's policy included investing in team training (focusing communication, cultural differences, trust, and context sharing. As a result of this initiative, the interactions between distributed teams were easier. Problems identified before the training started to occur less frequently, showing that the management of distributed teams is a key to the project success.

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

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

Lesson 8: Distributed Software Development is a maturity process.

The data collected in the study showed a clear difference between the maturities of both organizations related to the distributed software development (organization 1 was working with distributed projects at least for four years, while the organization 2 was working in this scenario for one year). Thus, the organization 1 was living different and more complex problems comparing with the organization 2. Based on the data collected with the interviews the development of distributed projects is something that needs time to mature. Although there are maturity models in software engineering, the GSD area doesn't have a model that identifies the organization maturity, based on all factors related to GSD activities.

6. THE REFERENCE MODEL

The centrifugal and centripetal forces proposed by Carmel (Carmel, 1999) are concentrated in the main factors related with distributed software teams. Additionally, Evaristo (Evaristo, 2003) presents ten dimensions related not only to software development projects, but also to more general distributed projects. Other studies analyse these factors from another point of view and also add other factors in GSD projects as a whole (Karolak, 1998), (Herbsleb, 2001), and (Morstead, 2003).

Considering these studies and the case study developed in this research, we propose here a reference model for GSD. This model has as purpose to support the global software development, acting as a guide for the software development done by teams geographically dispersed and heterogeneous. The model is composed by a set of critical variables and their relationship, identified during the study.

6.1 MuNDDoS Reference Model

The reference model (Figure 1) has two dimensions: organizational and the project dimension. As we expand the focus of the software development process, and try to adopt a more strategic position in relation to the process, we identify the planning stage as the first one to take place (organizational dimension). Additionally, it is possible to consider the planning phase as a former cycle of many project cycles (project dimension) derived from the planning process.

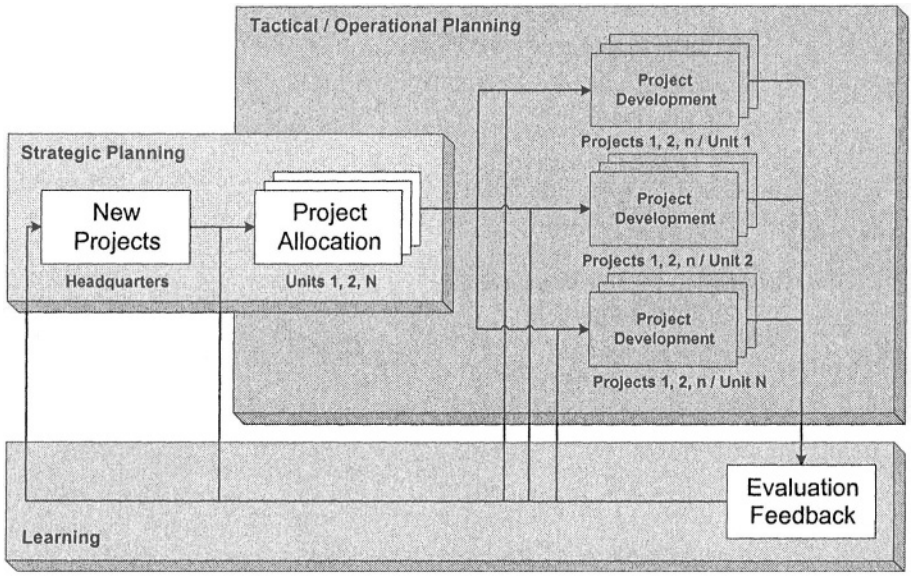

Figure 1 – The reference model (MuNDDoS)

Two cycles of planning can be identified for the management of GSD projects. The first one lies on the strategic planning, conducted by the organization headquarters and has as purpose to identify and prioritize new projects to be developed. The decision involves both projects from the organization areas and demanded by external clients. Moreover, the participants of this planning level are responsible for the strategic alignment between the perspectives and goals of each distributed software development unit and the headquarters.

The second cycle involves the tactical-operational planning in the scope of each distributed software development unit. The overlapping of the two planning cycles occur exactly when the projects are allocated, involving the planning and selection of projects that will be developed in each unit and the resource allocation. The project allocation has as purpose to select the projects that will be better developed in each unit, according an allocation policy defined by the organization. The

allocation must have as entry criteria the list of projects to be developed and the output of this step is the unit or units selected for each project. The tactical planning stage is of final responsibility (approval) of coordinators in each software development unit, while the operational planning involves the project management (project dimension), by the responsibility of the project manager.

The project dimension (Figure 2) involves specifically the software development project administration centered in the general coordination of the work between the collaborators, interfaces among teams, communication, and contacts with clients and conflict solving. We consider some factors that are the base to identify and minimize problems and weaknesses found in each project. The factors are presented using a conceptual map. Since each organization has its own project management process, we understand that the conceptual map presented can facilitate the identification of factors and can lead to implement strategies to minimize or anticipate problems.

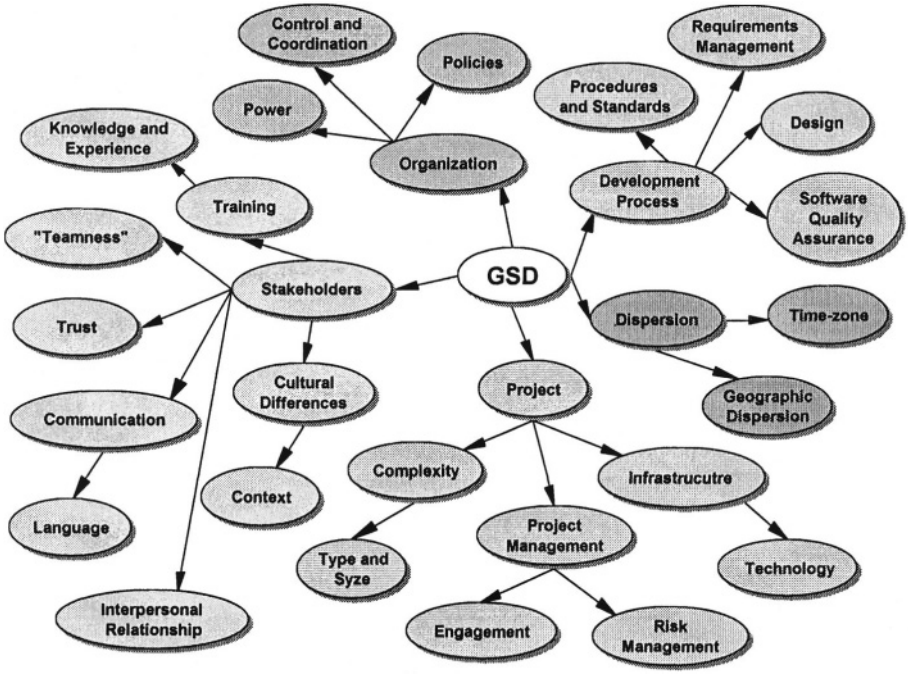

Figure 2 – The conceptual map for project development

Finally, the last cycle proposed is the learning cycle, related to the activities evaluation and strategies adopted. The model suggests the existence of a process to support the collection of data, involving the work evaluation, lessons learned from the projects, and other relevant information. In order to reflect all information in the previous cycles, all data need to be updated in a common repository.

6.2 The reference model in practice

Part of this model (new projects and project allocation) is being implemented in industry, in one of the organizations studied. Since we propose here a conceptual model, the organization is creating its own process and implementing a tool, having

the reference model to support the process. Besides that, a training program was developed to deal with concepts in the project development (project dimension), considering initially trust, culture, and communication.

Furthermore, it is being highlighted the role of dispersion in the software development process. Moreover, some "distributed processes" are being created, like requirements engineering and risk management. In the near future we will qualitatively analyze the results found for this implementation, search for improvements in the process and will try to implement in another organizations.

7. 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 are pushes more companies toward global software development. 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. As result, some lessons were learned and a reference model was proposed, based on case studies in two software development units from multinational organizations located in Brazil.

Although two organizations were formally interviewed, we lived other experiences interviewing people from other organizations, informally. Planned follow up studies in this topic will continue to analyze the organizations difficulties and solutions and will going deep in the study of specific factors, found in this work, like requirements engineering, risk management and project allocation, for example.

8. REFERENCES

1. Carmel, E. "Global Software Teams – Collaborating Across Borders and Time-Zones". Prentice Hall, USA, 1999, 269p.
2. Herbsleb, J. D., and Moitra, D. "Global Software Development", IEEE Software, March/April, USA, 2001, p. 16-20.
3. Pressman, R. S. "Software Engineering: A Practitioner's Approach". Fifth Edit, USA, 2001.
4. Prikladnicki, R., Peres, F., Audy, J., Móra, M. C, and Perdigoto, A. "Requirements specification model in a software development process inside a physically distributed environment", Proceedings of ICEIS, Ciudad Real, Spain, 2002.
5. Karolak, D. W. "Global Software Development – Managing Virtual Teams and Environments". Los Alamitos, IEEE Computer Society, USA, 1998, 159p.
6. Evaristo, J, R., Scudder, R., Desouza, K. and Sato, O. "A Dimensional Analysis of Geographically Distributed Project Teams: A Case Study," forthcoming in the Journal of Engineering Technology and Management, 2003.
7. Yin, R. K "Case study research: design and methods", Sage, USA, 1994.
8. Krippendorff, K. "Content analysis: an introduction to its methodology", California: Sage, 1980.
9. Morstead, S., Blount, G. "Offshore Ready: strategies to plan & profit from offshore IT enabled services", ISANI Press, US, 2003.

[i] Research developed at CDPe research center at PUCRS University, and funded in terms of the Brazilian Federal Law for Information Technology (Law No. 8.248/91).

[ii] The SW-CMM describes the principles and practices underlying software process maturity and is intended to help software organizations improve the maturity of their software processes (http://www.sei.cmu.edu/cmm/cmms/cmms.html).

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