Factors which affect the Distributed Software Development: an Experience Report

Rodrigo Rocha
*Informatics Center
Federal University of Pernambuco
Recife – PE – Brazil
rgcr@cin.ufpe.br*

Catarina de S. Costa
*Informatics Center
Federal University of Pernambuco
Recife – PE – Brazil
csc@cin.ufpe.br*

Alan Kelon O. Moraes
*Informatics Center
Federal University of Pernambuco
Recife – PE – Brazil
akom@cin.ufpe.br*

Alexandre Vasconcelos
*Informatics Center
Federal University of Pernambuco
Recife – PE – Brazil
amlv@cin.ufpe.br*

Silvio Meira
*Informatics Center
Federal University of Pernambuco
Recife – PE – Brazil
srlm@cin.ufpe.br*

Abstract

For many years the software development has become essential to the global market. In the past decade, as a reflection of globalization and other causes, software companies began to distribute their development processes in different places, creating the Distributed Software Development. For several reasons, this approach of Software Engineering inherited the existing problems in the traditional way of software development. This paper aims at describing the main factors found in the literature and in distributed software projects that positively or negatively influence the Distributed Software Development. Through a real experience of software factory, this article presents which factors influence the Distributed Software Development and how it happens, by citing factors needed greater attention, describing the solutions used to negative influences.

1. Introduction

Rocha [1] mention that it is notable that software evolution has happened in an accelerated way, so that several areas of knowledge have recognized its importance as strategic position before the market. In the last decades, globalization of business also caused impact on Information Technology industry. Economic forces have transformed national markets into global markets. These transformations have not just altered the marketing of distribution, but also, the form how products are conceived, built, tested and delivered to the final customers. [2].

Herblesb [3] justifies that, in the end of the last decade, many companies began to *experiment* the Distributed Software Development (DSD) with the *purpose* of decreasing costs and in search of more qualified resources.

Rocha [1] highlight that with the ascension of the use of this software development concept in the industry, the subject started to be discussed in International Congresses as IEEE International Conference on Global Software Engineering (ICGSE 2006-2008) and IEEE/ACM International Conference on Software Engineering (ICSE). This transformation in the manner how companies will produce software tends to be growing. According to Kiel [4], there is all reason to believe that companies will be pressed to adopt some DSD approach.

Even with several factors contributing to the growth of DSD, as well as, in the co-located development, to build software systems is not a simple task, and the complexity tends to increase in the scenery of distributed development .

According to Komi-Sirvo and Tihinen [5], DSD project inherit the same problems that co-located projects face. Therefore, they have problems with quality, time and cost that many times become more difficult to work in function of the distribution.

In this way, this work is based on a revision in the literature about the factors that affect DSD negatively and positively, and it has as objective to present how a team worked with these factors to minimize the effects of the negative factors through a practical experience.

In addition to the introductory section, this article is organized in the following way: in the Section 2, the concepts that DSD embraces, some characteristics and problems are presented in a brief way; in the Section 3, some factors that influence projects in distributed environments are mentioned; in the Section 4, the project that was used as case study is presented, describing its characteristics shortly, besides the factors that influenced the project and which adaptations were accomplished by the team; in the Section 5, the final considerations are presented.

2. Distributed Software Development

In some of his works, Herblesb [3] [6] mentions that it is no longer more uncommon software projects to possess development teams distributed in more than a place, sometimes even in more than a continent. The growing search for higher competitiveness has taken companies to adopt DSD, where different parts of the software are developed at different places. Trying to accomplish development at a low cost, companies have crossed borders forming a global market. This paradigm change has caused impact in the market, in the distribution and in the form of conception, of production, of project, of test and of delivery of the software to the customers.

In this manner, it has been characterized by the collaboration and cooperation among sections of organizations and for the creation of developers' teams, who work together, located at different cities or countries [2].

According to Audy and Prikladnicki [7], the main characteristics that differentiate the co-located development of the distributed development are: geographical dispersion (physical distance), temporary dispersion (time zone difference) and social and cultural differences (language, traditions, habits, norms and behavior).

According to Freitas [8], there are several reasons that motivate the adoption of DSD, the main ones are:

- Necessity of qualified professionals in specialized areas;
- Fiscal incentives for the investment in computer science researches;
- Availability of specialized labor and of reduced costs at countries in development;
- Accomplishment of stages of the software development close to the customers;
- Reduction of delivery time provided by the round-the-clock development;
- Formation of organizations and of virtual teams to take advantage of market opportunities;
- Necessity to integrate resulting resources of acquisitions and organizational fusions.

Karolak [9] defines an instance of DSD calling it of global development of software (Global Software Development - GSD) when the physical distance among the developers in an environment of DSD involves more than one country. GSD is instanced through global teams of software development (Global Software Teams).

Audy and Prikladnicki [7] explain that the traditional software development have always presented itself in a complex way and that the Distributed Software Development have increased other challenges to the process when adding factors as physical dispersion, temporary distance and cultural differences.

3. Factors whitch affect the Distributed Software Development

As mentioned previously, with the appearance of DSD, new challenges were increased to software development. Carmel [2] affirms the existence of five factors that can take to the failure a distributed team: (1) inefficient communication; (2) lack of coordination; (3) geographical dispersion; (4) loss of the team spirit; and (5) cultural differences, called of centrifugal forces. To add to that, the author also mentions the existence of six factors that can make success: (1) communication infrastructure; (2) product architecture; (3) team construction, (4) development methodology; (5) collaboration technology; and, (6) management techniques, called of centripetal forces.

Based on their researches, Komi-Sirvo and Tihinen [5] also present some problematic areas and associated factors when the development is distributed: (1) environments and development tools; (2) communication; (3) requirement engineering ; (4) management of projects; (5) cultural differences and, (6) time and budget above the calculated, among others.

The factors mentioned by Carmel, Komi-Sirvo and Tihinen [2] [5], among other factors that can influence directly and indirectly the development of a distributed project is described in the following sub-sections.

3.1 Project Management

According to Cleland and Ireland [10], the main functions of management of projects are: planning, organization, motivation, direction and control.

Some researches indicate that management of projects problems in distributed environments do not differ significantly of the problems in the traditional development (co-located). However, the importance of the formal techniques of management of projects is reinforced in the distributed context, for example, the ones presented in PMBOK [7].

In their work, Audy and Prikladinicki [7] assure that the main focus of the metrics in DSD is the coordination improvement and they mention examples of common metric in the distributed environment: (1) time using collaboration technologies, as audio and videoconferences; (2) amount of sessions / messages using collaboration technologies; (3) index of team construction; (4) days of trip; (5) resolution of problems rate.

In relation to coordination and control, the project management becomes extremely difficult in DSD. Integration between the several parts and modules of the project should happen in an efficient way so that the fact of they are far physically does not interfere [11].

Prikladnicki [12] affirms that the DSD management of projects demands an adaptation of some techniques used in co-located projects, as a way to support and to reduce the difficulties imposed by the dispersion of the team.

3.2 Software Processes

Evaristo [13] affirms that a development process is fundamental, since the synchronization of activities is one of the main objectives and that when the teams distribute the development process in several places, the lack of synchronization can become critical.

In their research, Angioni, Sanna and Soro [14] explain the proximity between Agile Methodologies and DSD mentioning some common characteristics for both, as continuous feedback, frequent releases, valuation of communication and code patterns. Besides, Angionini presents and describes MAAD (Methodology for Agile Distributed Development), that it is a process based on agile methodologies, created initially with focus on the open source community. That methodology was idealized from the evaluation of other processes of software development related to the distributed environment, as it is the case of DXP (Distributed eXtreme Programming) [15].

Some organizations use conventional methodologies to develop distributed projects, while others use experimental adaptations and others already use solid adaptations, in other words, they created their own process to work with the distributed context.

Several authors [2], [7], [1], among others mention the importance of using a reference as development model in the distributed environment, but they also highlight the scarcity of adapted methodologies that assist to that context and its needs.

3.3 Social-cultural Distance

Culture is the acquisition of a set of knowledge, experience, faiths, values, attitudes, senses, hierarchies, religion, time notions, functions, space relationships, concepts of universe for a group of people through the generations [16].

Souza [11] relates the social-cultural factor mainly in relation to the trust among the involved teams. So that a project has an efficient development, it is necessary that the members located in different atmospheres have trust among the participants and also in the project.

There are cultural differences in a global software development context, because there are people spread about of the world that have different habits, faiths, attitudes in relation to hierarchy, sense of time and communication styles. Souza [11] says that cultural differences can influence in several decisions inside of the project due to certain traditions and habits of each country and that there is necessity that the cultural differences are minimized to the maximum so that they do not interfere in the end result of the product.

Audy and Prikladnicki [7] assure that many companies work with the liaison concept, that is, a person who plays the role of a bridge between two or more cultures, exactly for understanding or already have lived.

3.4 Motivation

According to Olson and Olson [17], the individuals' motivation also differs from country to country depending on its cultures. In countries where the valuation of the individualism exists, the people look for gain material and personal recognition.

But, in others, where the emphasis is focused on the collective, personal relationships and family are sought. The reward systems should take into account these values, rewarding each group with monetary values or rest days in agreement with their values.

Teams are formed by people; therefore, they are fragile social units that for several reasons (weak communication, temporary distance and geographical distance, used infrastructure, lack of informal communication, cultural difference and group size, among others) become vulnerable. Evaristo and Scudder [18] assure that trust is fundamental for the team to maintain the team spirit, that it is difficult, even so essential to maintain trust in the development of a distributed project.

3.5 Geographical and Temporal Distance

According to Herblesb.[19], when the distance among the involved ones in distributed projects surpasses 30 meters, the communication frequency decreases to the identical level to the involved ones that are distributed to hundreds / thousands of meters.

Evaristo and Scudder [18] affirm that the high dispersion level provides more difficulty to monitor the behavior among different groups in relation to other groups.

In their work, Audy and Prikladnicki [7] mention that it is necessary to understand the level of existent distance, because with this, it is possible to help in the identification of possible sources of difficulties or just to characterize the physical distribution of the teams involved in distributed projects in a better way. And this manner, four levels of physical dispersion are presented:

- Same physical location: situation in which the company possesses all the actors in a same place and face to face meetings can happen without difficulty;
- National distance: people involved in this situation are at the same country, can be in different areas, states and/or cities. Difficulties already appear to face to face meetings;
- Continental distance: actors participating at countries of the same continent;
- Global distance: actors located in different countries and continents. In this situation, as well as, in continental distance, face to face meetings become much more difficult of happening, generally they happen in the beginning of the project;

Besides physical dispersion, temporary dispersion produces an effect on DSD. Participants of a team are dispersed at the time when there is difference in the work schedules, time zones and/or work rhythms that decrease the available time for synchronous interaction [7].

Souza [11] mentions that geographical and temporary distance can be a problem if they are not administered in the best possible way. Due to distance and difference of time zones that can exist, communication, interaction and information change among the development teams should happen through tools (for example, based on web, framework and control of versions), and together with a support, so that the project reaches the foreseen objectives.

3.6 Project Complexity and Size

Herbsleb [3] affirms that project complexity level can also affect performance of distributed projects and that project dimension can also be a complexity factor.

As stated by Souza [11], projects with distributed software development are usually complex and of great load, due to the fact of the initial investment in this type of project is high, and it can involve several countries. The use of traditional and more complete methodologies is necessary in this aspect.

3.7 Communication

Based on several researchers, Carmel [2], Audy and Prikladnicki [7], Rocha [2], Souza [11], among others, communication is fundamental factor for the development of distributed projects. Teixeira [19] mentions that communication is one of the factors more harmed in DSD, for this reason, a well-structured communication channel among the teams through the environment is important, maintaining all the participants informed on the project process.

Souza [11] assures that communication, in all its senses, is one of the great problems of the distributed development. It is necessary that there are efficient means that surpass the barriers imposed by the development in different places. The definition of formal communication interfaces can be obtained by means of much defined process models, with marks (milestones) and metric very established. The channels of informal communication, for its time, can embrace videoconference, spaces of shared work, programs of message change (instant messengers), among other [20].

Rocha [1] mention in their experience that some tools were fundamental for communication among the members, such as instantaneous communicators or messengers and a website, but, that common tools of communication (e-mail, messengers, etc) would need to develop to support the distributed work better.

3.8 Environments and Development Tools

Komi-Sirvo and Tihinen [5] affirm that although the available technical infrastructure now seems to provide the support adapted for the distributed development, researches done with companies that possess distributed projects indicated that tools and development environments do not still support the distributed development in an efficient way.

According to Komi-Sirvo e Tihinen's research [5], the incompatibility of tools and versions used by sites of different development are very common, and the establishment of a uniform environment is a challenging task. Additionally, development tools are based on the presupposition that nets of communications are extremely fast, which not always it happens.

4. TechnoSapiens and CitIX Mobile Project

The presented practical experience is part of a discipline of Software Engineering [21] focused on software creation factories that make use of distributed development for the accomplishment of real projects. With the definition of customers and projects, students have four months to execute a software factory environment and to deliver the defined products in common agreement with the customer.

Called TechnoSapiens, the factory was composed by nine students, all of them working in a distributed way, with part of the team working at the same city and some members at other cities, totalizing 3 cities, and all the members were working together for the first time.

The project assumed by the factory was the development of a version for mobile devices of citIX service [22] with reduced functionalities. The citIX uses a system of geographical information that allows that the service users build a net of knowledge on characteristics (public safety, entertainment, infrastructure, public services, etc.) of a certain area.

The team had had meetings twice a week, a meeting in the beginning of the week and another in the end of the week. In the first meeting, the activities were defined, and in the second, the same ones were evaluated, for this, discussion group and a tool planning web and accompanying were used, besides instantaneous communicators or messengers and a factory website.

4.1 Adaptations to the factors that affected TechnoSapiens distributed development

In the next sub-sections, factors that affected the distributed development of TechnoSapiens software factory are mentioned, as well as, the ones that were treated to minimize its effects.

4.1.1 Project Management

With the dispersed team, distributing and accompanying the activities is not an easy task. In the beginning of citIX project, it was allocated a manager and a vice-manager to plan and to orientate the project activities. Even with an allocated manager, there was difficulty in the distribution of tasks, mainly for lack of knowledge about the difficulty degree in the accomplishment of the same ones. Besides, the accompanying became much more difficult for e-mail, since, the manager only knew what the team members informed; the manager did not have precise information of the project process.

In the beginning of the project, the uncertainty degree was high and the team roles were still being defined. The manager's and a vice-manager's allocation helped the team to define the roles and related activities in a better way in the beginning of the works. At the end of the time foreseen for the certain activity accomplishment, the member should report to the manager the accomplishment or not of the same, the difficulties and impediments. The accompanying was done through e-mails and face to face meetings in the beginning and in the end of the week, they could not always count with the presence of all the members.

During the first weeks, the fragility of this approach was noticed because as the manager was the responsible person for the estimates, the members were not really committed with the dates that were established and with elapsing of the works; there were an increase and more dependence among the activities, what began to delay the project. Besides, the management did not get to accompany the project process precisely.

The solution found by the team was to use of a management web tool in which the management or the own responsible person made the cadastre of activities and time esteemed for their accomplishment. It was noticed that when the own member is responsible for the *estimation* of his/her activities, the compromising degree increases.

All the team members had access and they could attribute, accompany, share and re-define activities. With this, the management accompanied the project process in a more realistic way, and the availability of some data given by the application facilitated the use of metric, for example: dedication in hours of each member, time used in the execution.

4.1.2 Software Processes

The process definition was one of the first team's concerns, considering that without a defined process, with phases and patterns to be followed, risks would be very higher. Part of the team was responsible for process definition, that it should take into account the participants' physical distribution, the little experience of the team with JavaME development technology and the need of a light process that contemplated a smaller amount of engines.

Due to little experience of the team members in other development methodologies, a process instance based on Rational Unified Process (RUP) [23] was defined. This instance assisted the foreseen demands of the project, but there was the preoccupation in resulting in a heavy process, with a lot of documentation, since the team was with a reduced number of members. Thus, the possibility of a better process adaptation was verified by the addition of some practices of agile methodologies, as: frequent communication between the group and the customer and among the group members; programming in pair, that even at distance, served to that members with less experience could be aided by others with more experience; and, re-factoring, accomplished by the member more expert of the group in the technology with the purpose of organizing the structure, to improve the legibility and to standardize the system code.

The main phases considered for the process were: Pre-sale, Requirements, Analysis and Project, Execution, Tests and Implantation. Support phases to the previous phases were also defined: Planning and Project Management, Configuration Management and Quality Warranty. With the beginning of the activities, a new support phase was necessary, the Revision phase and Approval were added to the process with the aim of accomplishing the prevention and correction of technical mistakes, guaranteeing in this way, its consistency and correctness. In the project elapsing, other adaptations were necessary, with the team's reduced size, the number of engines was reduced.

4.1.3 Geographical and Temporal Distance

The team was distributed in a national distance, in 3 cities of a same country. Although there was not difference in the time zone among these cities, face to face meetings are more difficult to be set up. With the project process, the face to face meetings that occurred twice a week defined in the beginning of the project, became more difficult for the members who needed to move from other cities, and communication with part of the team became virtual, with few face-to-face meetings.

Mainly in the execution phase, geographical distance became a great challenge, since, the team members who possessed a larger domain of the development technology were the ones who were more distant from the other members.

The solution found by the team was the use of programming in pair, practice used in agile methodologies. Thus, the members with less experience in development technology were aided virtually by others with more experience. This way of collaboration was very effective, and it was supported by tools that contributed to this, for example, e-mails and messengers with or without audio.

4.1.4 Project Complexity and Size

The CitIX Mobile project had 4 months of duration, and in the beginning, its goal was very detailed. As there was already a citIX web application, the team should develop the mobile application based on the same functionalities of the web application. The team noticed the complexity of the project soon, since the development technology was not very known by most of the team, and there was a great number of functionalities requested by the customer.

With the project elapsing, the complexity increased, some services that were in the customer's responsibility, that had already developed the citIX web, were delivered out of the period, delaying the factory activities and the team needed to negotiate with the customer the reduction and even the exclusion of some functionalities to conclude the project inside of the period requested by the discipline. Even with these actions, the team needed to dedicate plenty of time in the execution, taking more time than the calculated, what delayed the phase of tests, reducing the number of accomplished tests.

4.1.5 Communication

The communication was always treated as essential for the team's works. Besides the two weekly meetings, several communication mechanisms were used with the purpose of softening the difficulties imposed by the dispersion, as: e-mails, messengers and a factory website.

Although such mechanisms were good to aid the course of the project, it was observed the inefficiency of some tools by the team; for example, the software for audio conference didn't support more than 5 people speaking at the same time. When it was necessary to arrange an online meeting through conference, a moderator was defined to coordinate the chat, deciding who would speak in which moment through the speech solicitation for someone of the team, in this way, overload in the conversation was avoided.

The e-mail is also a means of communication that is not synchronous in several moments and it had not assisted the team's needs completely because the doubts and information were only answered when other members were available. The best manner to deal with this problem was to use synchronous means of communication, as telephones and chats.

4.1.6 Environments and Development Tools

In the beginning of the project, there were many problems in relation to the development structure and the used environment. Although a team member was expert on development technology, which facilitated the preparation of a structured environment for the development, the installation and adapted configuration of the development tools was not trivial for most of the team. To resolve this problem, there was training for the team about the installation, configuration and adaptation of the environment.

The configuration manager assisted individually all members who even after the training had problems with the configuration and adaptation of the development environment.

Although the team delayed some activities for difficulties in the configuration of the development environment, the team succeeded in solving the problems and the works continued, mainly with the efforts of the most experienced members.

5. Final Considerations

In the distributed scenery, software projects assume different perspectives and consequently new risks. If there is not a good knowledge of the factors that can influence the project, the same will have more chances of not obtaining success.

In this work, it was possible to present factors that influence DSD found in the literature and to mention solutions used in a practical experience to deal with negative factors. Based on this, some learned lessons were observed in our experience, such as: identification of the factors that in fact influenced our project and the search of solutions in similar situations of practical experiences.

Futurely, some works can be developed starting from this report, as: studies and researches more deepened about negative factors that influence DSD projects in order to identify better ways to work with these factors; classification of all the existent factors that influence DSD in human, technical and organizational factors with objective of establishing a higher consolidation of these factors; to analyze how the factors affects the project and how negative factors can be managed.

In this sense, practical and experimental approaches work as an interesting field for companies that are evaluating the best manners of working in this development model, because through reports of experiences, it is possible to verify problems and how the same ones were solved.

6. References

[1] Rocha, R. Arcoverde D. Brito, R. Aroxa, B. Costa, C. Silva, F. Q. B. Albuquerque, J. Meira, S. R. L. “An experiment in the adaptation of RUP for small teams of distributed development”. II Workshop de Desenvolvimento Distribuído de Software (SBES-SBBD). pp. 87-96. 2008.

[2] E. Carmel. *Global Software Teams: Collaboration Across Borders and Time Zones*. Prentice-Hall, EUA. 1999.

[3] Herbsleb, J. D.; Moitra, D. *Guest editors’ introduction: global software development*. IEEE Software. 2001.

[4] Kiel, L. Experiences in Distributed Development: A Case Study. Workshop on Global Software Development at ICSE 2003, Oregon, EUA. 2003.

[5] Komi-Sirvo, S; Tihinen M. “Lessons Learned by Participants of Distributed Software Development”. Journal Knowledge and Process Management, vol. 12 nº 2 pp. 108–122. 2005.

[6] Herblesb. J. D. “Global Software Engineering: The Future of Socio-technical Coordination”. IEEE Computer Science. P188-198. 2007.

[7] J. Audy, R. Prikladnicki . *Distributed Software Development: Software Development with distributed teams*. Rio de Janeiro: Elsevier. 2007.

[8] Freitas, A. V. “APSEE-Global: a Model of Processes Management of Distributed Software Processes”. Faculdade de Informática – UFRS – RS- Brazil. 2005.

[9] Karolak, D. W. “Global software development – managing virtual teams and environments”, IEEE Computer Society, EUA, 1998.

[10] D. Cleland, L. Ireland. “Project Management”. 1° Edition. Rio de Janeiro: Reichmann & Affonso Editores, page 324. 2002.

[11] Souza, M. “Analysis of the software development process for use in Distributed Software Development”. Pelotas, Brazil. 2007

[12] Prikladnicki, R. “Munddos: A Reference Model for Distributed Software Development”. Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, RS, Brazil. 2003.

[13] Evaristo, R., Audy, J. L. N., Prikladnicki, R., Pilatti, L., Lopes L. “Innovation in Information Systems Education-V: The Management of Outsourcing: Development of a Module with Implications for the IT Curriculum”. Communications of the Association for Information Systems, Volume 15, Article 21. 2005.

[14] M. Angioni, R. Sanna, A. Soro. “Defining a Distributed Agile Methodology for a Open Source Scenario”. In Proceedings of the First International Conference on Open Source Systems, pp 209- 214. 2005.

[15] Kircher, M. “eXtreme Programming in Open-Source and Distributed Enviroments”, JAOO (Java And Object-Orientation) conference, Aarhus, Dinamarca. 2001.

[16] L. A. Samovar and R. E. Porter. “Communication Between Cultures”, 5a edition. Belmont (California), Wadsworth/ Thomson Learning. 2004.

[17] Olson, J. S., Olson, G. M. Culture Surprises in Remote Software Development Teams. ACM Queue, New York, v. 1, n. 9, pp. 52-59. 2003.

[18] R. Evaristo, R. Scudder. “Geographically distributed project teams: a dimensional analysis”. In: HICSS, pages 1-15., Havaí. Proceedings..., EUA, 2000.

[19] Herbsleb, J. D., Mockus, A., Inholt T. A., Grinter, R. E. “An empirical study of speed and communication in globally-distributed software development”, IEEE Transactions on Software Engineering, 29:6, pp. 481-494. 2001.

[20] Teixeira J. “Modeling an environment for managing distributed software development based workflow”. Pelotas, Brazil. 2007.

[21] IN953. Software Engineering: Building Open Source Software Factories. http://www.cin.ufpe.br/~in953/. 2008. (Accessed Nov 20, 2008).

[22] CitIX. city Information eXchange network. http://www.citix.net/index/. 2008 (Accessed Jul 20, 2008).

[23] RUP. Rational Unified Process. http://www-01.ibm.com/software/awdtools/rup/. 2008. (Accessed Ago 25, 2008).