Collaboration Models in Distributed Software Development: a Systematic Review

Rodrigo G. C. Rocha
Federal Rural University of Pernambuco (UAG/UFRPE)
Garanhuns – PE – Brazil
rodrigo@uag.ufrpe.br
and
Catarina Costa
Federal University of Acre (UFAC)
Rio Branco – AC – Brazil
csc@cin.ufpe.br
and
Cleyton Rodrigues, Ryan Ribeiro de Azevedo, Ivaldir H. Junior, Silvio Meira
Federal University of Pernambuco (Cin/UFPE)
Recife – PE – Brazil
{cmor, rra2, ihfj, srlm}@cin.ufpe.br
and
Rafael Prikladnicki
Pontifical Catholic University of Rio Grande do Sul
Porto Alegre – RS – Brazil
rafaelp@pucrs.br

Abstract

Several years ago software development has become critical to the global market. In the past decade, as a reflection of globalization, software companies began to distribute their development processes in different places, creating distributed software development (DSD). With the growing of Distributed Software Development, organizations have attempt to sketch the best possible development structure, in order to improve productivity and quality. The aim of this paper is to present a Systematic Review of Literature to identify which ways of collaboration are commonly used by software organizations where teams are temporal and geographically dispersed, with also different native languages and culture. Further, the research was based on the basic life cycle of the traditional development, and where phases of the project are performed: onsite, distributed/offshore and multi-site. The systematic review have examined 868 papers published between 2000 and 2010.

Keywords: Collaborative Models, DSD, Systematic Literature Review.

1 Introduction

Given the demand for better software products, faster services and the growing number of companies developing these solutions, in the past decade, as a result of globalization, software companies began to distribute their development processes in different places, creating the Distributed Software Development [6], also known as Global Software Development (for projects involving other countries). Several authors argue that systems development is a complex activity [15], since software development involves various risks and uncertainties. By adding to the traditional difficulties of developing software, the physical and temporal distance between the participants of the

process, the challenges of development tend to be accented and other difficulties can emerge.
Thus, many companies try to adopt techniques, methodologies and tools to support them when dealing with common variables of a distributed context. At various times, companies that use this model of development need to define what methods they will use in a given project and their practices, because they do not provide information that will be able to determine which method is most appropriate for a particular type of project or which practices are most suitable.
According to Binder (2007) [2], although many organizations have been executing projects with distributed teams, just a few of them have effective established practices on supporting software process and developers working on this novel environment.
In addition, Pichler (2007) [10] reports that many distributed project teams are still created as if all their members were working in the same place, ignoring some problems of the distribution. Moreover, DSD projects usually face the same problems of co-localized projects, such as challenging schedules, quality loss, and cost overrun. Therefore, those congenital problems get even more difficult to be addressed in distributed projects.
With the decentralization of business and production taking place in distributed environments, software development becomes more complex, requiring that organizations need to seek models of collaboration that can meet their characteristics and needs. Smite and Borzovs (2009) [14] say that the project leaders must be aware of the way the organization of teams and the project can be done. Hence there must be a consensus regarding how the project leaders can arrange the models of collaboration, i. e., how a project can be split into minor activities in each phase, and further, how each activity can be performed.
The objective of this study is to identify, through a Systematic Review of Literature, what forms of collaboration are used by industry and/or academia to develop software in distributed environment, based on the basic life cycle of traditional software development (requirements, analysis, implementation and testing), but also whether the variations and phases of the project are conducted onsite (client), offshore (distributed) and multi-site (in both). Furthermore, the methodology adopted to identify models of collaboration will be described.
This paper is organized as follows. Section 2 presents the concepts of Distributed Software Development and Systematic Literature Review, Section 3 describes the steps taken during the systematic review, the way the search was conducted, the search strategy (key research, search sources, among others) and their results; in Section 4, the development models are outlined, and finally, Section 5 covers the related works, and the Section 6 presents the final considerations.

2 Theoretical Foundation

The concepts for the realization and understanding of the work are presented below.

2.1. Distributed Software Development

Developing software on the same physical space, in the same organization or even in the same country, has become increasingly expensive and less competitive [12]. The economy advancement, the sophistication of the media and the pressure of costs have encouraged massive investment in DSD.
According to Brooks (1978) [3], the software has many features that make it suitable for this approach: it can be replicated, transmitted, corrected and used over unlimited distances with virtual zero cost. Yet it is also subject to change, intangible and can become potentially complex.
In some of his words [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 part 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.
Thus, the distributed development has appeared in recent years as an alternative to software development. It is a phenomenon that has been growing since last decade and had been characterized by collaboration and cooperation among departments and organizations, creating groups of developers working together, located in different cities or countries [8].
The principles aforementioned are not, exclusively, the only factors that DSD contributes to, as cited by Prikladnicki, Audy and Evaristo (2004) [11]:

- Increase resources when the availability of local labor is scarce;
- Proximity of the local market, including knowledge of customers and local conditions;
- Speed up the formation of corporations and virtual teams to explore market opportunities;
- Pressures on time-to-market, using different time zones allowing the development including 24x7 (twenty-four hours, seven days a week).

In this context, a software organization that works in a distributed development environment is subject to several technical, human and organizational problems as communication, management, relationships, among others. Prikladnicki e Audy (2009) explains that the software development has always presented itself as complex process, and DSD has aided other challenges like physical and temporal dispersion, and cultural differences. When the project development involvs other countries it is known like Global Software Development.

2.2. Systematic Literature Reviews

Systematic Literature Review (SLR) is part of the paradigm of evidence-based practices. Widely used in medicine and health sciences, systematic reviews are becoming popular in other areas, but are not yet well established in software engineering software [9].

The essence of evidence-based paradigm is systematically collect and analyzes all available data about a particular phenomenon to obtain a fuller and wider perspective than one can pick through an individual study. One of the main evidence-based methods for software engineering is systematic reviews, which are classified as secondary education, since they depend on their primary studies used to discover evidence and construct knowledge [9],[16]. The Systematic Literature Reviews (SLRs) are studies that act as a guide for projects development, in order to target a specific research, summarizing all the relevant studies that address the particular issue.

According to Biolchini et al. (2005) [16], a SLR is a way to perform an unbiased literature review in a comprehensive manner, so that its results have a scientific nature. There are some aspects that precede the onset of a systematic review, which should be taken into account, such as defining the purpose, recognizing the literature and evaluate the possible studies to be included. These three items are essential to the researcher, because they help to complete the core question of the review, making it well-formulated and clear [5].

The onset of the SLR is given first by determining the protocol that lists the issues to be researched and the methods used to guide the review. Thus, Biolchini et al. (2005) [16] point out three key stages to build the SLR, they are: Planning, Implementation and Analysis of Results, which are detailed in Figure 1.

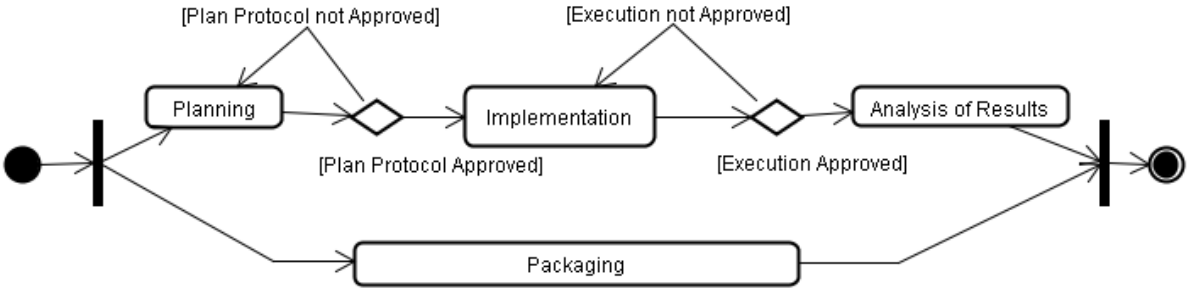

Figure 1: The Generic Process for Systematic Review. Biolchini et al. (2005)

In **Planning** is included the main objectives of the research, besides the issues and criteria that influenced the inclusion or exclusion of findings, research strategies, the prior choice of the articles obtained, their final selection and direction of the review. This phase defines the protocol for review, where it is suggested a review by specialists or even through a test run.

It is at the stage of **Implementation** where the research is carried out in the pre-established sources, for the study of selected works, followed by their classification according to the criteria of inclusion and exclusion defined in the protocol. If there are restrictions in the course of searches, adjustments can be made to meet any limitations.

In the last phase, the **Analysis of Results** is performed to collect data extracted from the articles identified and selected according to criteria established in the protocol. From this, these data are analyzed and synthesized according to the method chosen. At the end of a SLR execution, we may obtain a mapping of the real situation of the subject, providing a better planning.

3 Process to Identify the Collaboration Models

This systematic review aims to answer a research question, which, particularly was: "What are the models of development for the DSD". This question aims to assimilate what are the models used by industry and/or academia, to develop software in distributed environment; based on the basic life cycle of traditional software development and its variations, and also, where each phase of the cycle is performed.

In accordance to the recommendations of Kitchenham and Charters (2007). [7], a systematic literature review should be conducted through the following steps:

1) Planning the Review
- Identification of the need for a review
- Specifying the research question(s)
- Developing a review protocol
- Evaluating the review protocol (carried out by an specialist in systematic review and DSD)

2) Conducting the Review
- Identification of Primary Studies
- Selection of Primary Studies
- Study quality assessment
- Data extraction
- Data synthesis

3) Reporting the Review
- Specifying dissemination mechanisms
- Formatting the main report

3.1. Search Strategy

When starting the search of primary studies, Kitchenham and Charters (2007) [7] suggests that a strategy should be used, by defining the keywords, digital libraries, journals and conferences.

3.1.1 Search String

From combinations made with relevant keywords, it was built the directed standardized search string. The terms and synonyms identified are presented below:

- Distributed Software Development: Distributed Software Development, Distributed Software Engineering, Distributed Software Teams, Global Software Development, Global Software Engineering, Global Software Teams, Collaborative Software Development, Software Collaborative Engineering, Collaborative software teams, Globally Distributed Work, Distributed Development, Distributed Teams , Global Software Teams, Globally Distributed Development, Geographically Distributed Software Development, Offshore Software Development, Offshore, Offshore Outsourcing, Dispersed Teams;
- Models of Development: Collaboration Models, Collaboration Model, Models of Collaboration, Model of Collaboration, Development Model*, Models of Development*, Collaboration form*, Form of Collaboration*, Development process*, Process* of Development, Work form*, Form* of Work, Life Cycle Models *.

The terms marked with '*' indicates that these were searched in both singular and plural forms.

3.1.2 Definition of Source Search

To select the sources of research, some criteria were defined: the papers must be available on the web, search engines by keywords with guaranteed single results to the same set of keywords. The articles can also be obtained by people with experience in the subject. The language of the sources must be in English as well as the language of the articles, since it is the most common language between the main conference s investigated.

The sources used were: (i) IEEEXplore Digital Library, (ii) ACM Digital Library, (iii) Elsevier ScienceDirect, (iv) EI Compendex, (v) SCOPUS. All the first four sources were used to search the terms up to 2009. Nevertheless, the SCOPUS was added to search new results in 2010**.** Besides the sources cited above, it was also included in the study the book's chapter Infonomics for Distributed Business and Decision-Making Environments, Creating Ecology Information System [14], since it deals directly with the DSD models of collaboration.

3.2. Criteria for Studies Inclusion

The analysis for inclusion of articles was done by title, keyword, abstract and conclusion of work. The following inclusion criteria were defined:

- Available on the Internet;
- In English;
- It has been published between 2000 (when DSD started to really be consolidated) and 2010;
- Studies that have (primary or secondary) development models for distributed software projects related to the basic cycle of development (requirements, design, coding, testing);
- Having commercial / industrial / academic projects.

3.3. Types of Study

The types of studies are classified as:

1. Experimental and Empirical Studies (studies where data were collected and analyzed);
2. Theoretical (conceptual studies based on an understanding of an area, referencing other related works);
3. Systematic Reviews (articles that use a well defined methodology for identifying, analyzing and interpreting evidence related to a specific research question);
4. Industrial Experience Report.

3.4. Quality Evaluation

Although there is no universal definition of what constitutes quality of study, most checklists include questions which aim to assess the extent in which bias is minimized while the internal and external validation are maximized [8, 9].

To answer the questions of the quality criteria, the researchers used the following levels of agreement or disagreement for each primary study: totally agree, partially agree, undecided, partially disagree and strongly disagree. Table 1 presents the issues of quality assessment studies. It must be considered the following observations:

- Totally agree: equivalent to a maximum of five points and should be granted in cases where the work fully meet the criteria of the issue;
- Partially agree: equivalent to a maximum of 4 points and should be granted in cases where the work meets partially the criteria of the issue;
- Undecided: equivalent to 3 points and should be granted in cases where the work does not make clear whether or not answers the question;
- Partially disagree: equivalent to 2 points and should be granted in cases where the criteria in question are not hold by the work evaluated;
- Disagree: equivalent to 1 point and should be granted in cases where there is nothing in the work that meets the criteria in question.

Table 1: Questions for Study Quality Evaluation

Planning			
1. Does the study goals or research questions are clearly defined (including the reasons for the study realization)?			
2. Does the study type is clearly defined?			
Execution			
3. Does the study provide a clear description about the research context?			
4. Does the study cite related works and is based on models and theories from the literature?			
Results			
5. Does the study present the results clearly and unambiguously?			
6. Do the study goals were achieved?			
Empirical Studies	**Theoretical Studies**	**Industrial Experience Report**	**Literature Reviews**
7. Does any method or set of methods was used in the study?	7. Is there an unbiased process for study selection?	7. Is there a description about the organization, team, project and distribution researched?	7. Is there a rigorous process description and it is followed?
Criteria for Investigation			
8. Does the study present (primary or secondary) development models for distributed software projects related to the basic cycle of development (requirements, design, coding, testing)?			

From the results returned by the evaluation questions, each study received a final score. Thus, the studies reviewed fell into five quality levels from the values of the final evaluation of each study: Great (33-40), Good (25-32), Regular (17-24), Poor (9 - 16) and Terrible (0-8). The quality evaluation of this work is presented in Table 2.

Table 2: Quality Evaluation of Primary Studies

	Terrible	Poor	Regular	Good	Great
Amount of studies	0	0	1	3	5

3.5. Data Extraction and Synthesis

In order to register the systematic review, facilitating the extraction and synthesis of data for later analysis, the tool Jabref (http://jabref.sourceforge.net/) was used. This tool is an open-source reference manager, which allows customization and facilities when importing and exporting data.

In addition to this tool, for every article approved by the selection process, the researchers also made use of forms A and B. The former lists the articles included, with only the information that identify themselves and graphs displaying the results of the review. The second was used to extract general information and implementation of quality assurance.

3.6. Results

Table 3 describes the results obtained, ranging from the issue of research to the primary studies. After the search into the four established sources, the papers found were selected primarily based on the title and keyword. From 95 works found in IEEEXplore Digital Library, 58 were entered in search criteria. In ACM Digital Library, from 443 results, only 139 met the purpose of this research. Further, we found 235 articles in Elsevier ScienceDirect, of which only 48 fit on the specifications, and 42 works (of the 67 found in EI Compendex) were accepted for the second round of selection. Further, through SCOPUS the search has found 127 researches at first round, of which 38 (thirty-eight) works were accepted after second selection round**.**

Table 3: Search Method and Choice of Primary Studies

Source	**Search Results (a)**	**Potentially Relevant (b)**	**Not Relevant**	**Repeated**	**Incomplete**	**Relevant Studies (c)**
IEEEXplore	95	58	52	0	1	5
ACM	443	139	134	0	4	1
ScienceDirect	235	48	47	0	0	1
EI Compendex	67	42	31	2	9	0
Infonomics for Distributed Business and Decision-Making Environments: Creating Information System Ecology	1	1	0	0	0	1
Scopus	127	38	15	3	19	1
Total	**968**	**326**	**279**	**5**	**33**	**9**

After the first selection, the 968 works were reduced down to 326 potential studies that were analyzed again, this time through the abstract and conclusion. From this, the studies that met the inclusion criteria were identified as primary and the others not accepted were subdivided into Irrelevant, Repeated/Duplicate and incomplete. The works selected as primary studies were assessed according to their year of publication.

The chart in Figure 2 shows the temporal distribution of the primary studies. All selected studies were published between 2001 and 2010, and 8 have been published after 2006, which coincides with the startup of new conferences in the theme, including ICGSE.

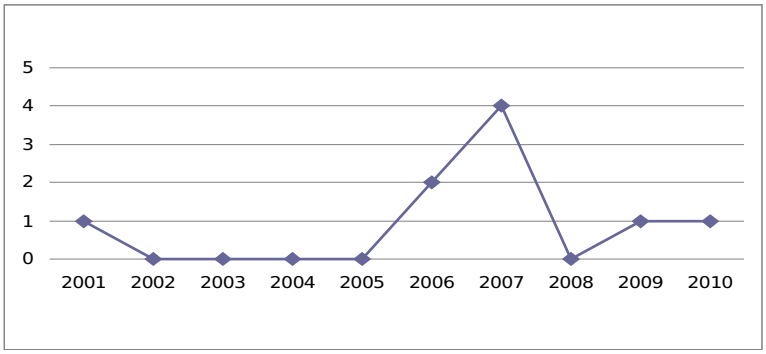

Figure 2: Temporal View

The types of study are classified as: Experimental, Theoretical, Systematic Reviews or Reports of industrial experience. Table 4 shows that among the 9 studies, 5 are Empirical Studies, 2 are Theoretical Studies and 1 are Industrial reports. No systematic literature review was identified.

Table 4: Type of Study Final Distribution

Type of Study	#
Empirical Studies	6
Theorethical Studies	2
Industrial Experience Report	1
Others	0

In Figure 3, the distribution of the primary studies based on the country of origin of the research team is presented. Four studies (4/9) were developed by research groups in one single country and four involved collaboration among two or more countries.
All together, the studies are originated from 13 countries. However, there is a concentration of studies from United States (USA), India and Pakistan based teams. Is also to highlight that some of these studies were researched in unspecified countries (security reasons) from Scandinavia and Western Europe.

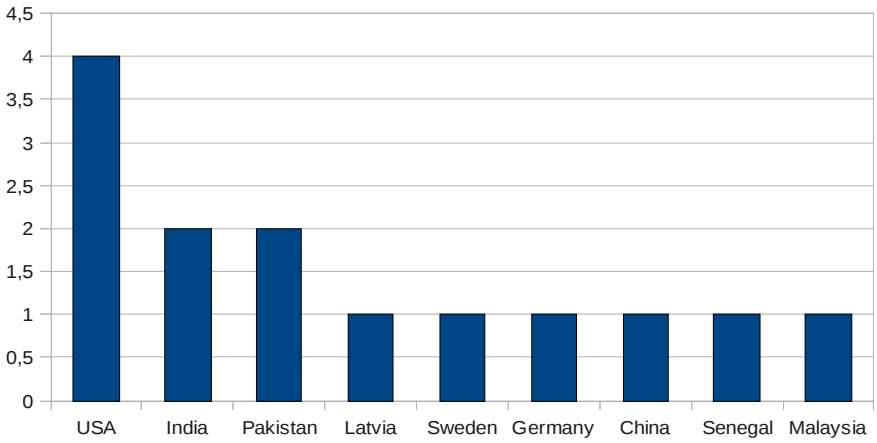

Figure 3: Countries in the Studies

4 Collaboration Models Identified

Table 7 presents the models of development identified in the systematic review. In the left column are exposed all studies and the other columns represents the phases that were found in the Systematic Review of Literature. Beside, the research and models of development are presented, e.g., study 1 (W1 - meaning Work 1) possesses five phases of development, and the requirements phase performed multisite, the design phase and encoding performed offshore or distributed, the tests were multisite, and deployment, onsite. Similar to W1, other studies (W2, W3, W4, W5, W6, W7, W8 and W9) were described following the same reasoning, with the same set of activities being realized either in the client or on a distributed environment.
The Table 6 is used to understand the phases found in primary studies. These phases were found through the extraction of information in data analysis step of the SLR. The primary studies characterized the stages used in those projects.

Table 6: Subtitle of phases

Phase	**Name**
P1	Requirements
P2	Schedule and Release Planning
P3	Team Selection and Contract
P4	Analyze
P5	Design
P6	Encoding
P7	Test
P8	Quality Assurance
P9	Training Team Offshore
P10	Usability Evaluation
P11	Deployment

Table 7: Models of distributed software development identified

Phases / Studies	P1		P2		P3		P4		P5		P6		P7		P8		P9		P10		P11	
	on	dis	on	dis	on	dis	on	dis	on	dis	on	dis	on	dis	on	dis	on	dis	on	dis	on	dis
W1	■	■								■		■	■	■								
W2	■						■		■			■			■	■						
W3									■	■		■							■	■		
W4	■	■					■	■	■	■	■	■	■	■							■	■
W5	■	■							■	■	■	■	■	■	■	■		■				
W6	■		■	■	■	■	■	■		■		■	■	■				■				
W7	■	■					■	■	■	■	■	■	■	■								
W8	■	■								■		■		■	■	■		■			■	

Through the results, it was found eight models of distributed development, and the phases of development and where they are held vary from work to work. The nine study (W9) is to Smite and Borzovs (2009) [14], which can be considered as a related work. In this study, we have identified nineteen models of collaboration through a field survey with thirty-eight distributed projects, however, unlike this paper, the study of Smite and Borzovs focuses on just four development phases (Analysis, Design, Coding and Testing). This may limit the study, since each project is unique, and its context and characteristics vary from project to project.

The collaboration models for distributed environment determine how companies/organizations have developed their projects, relating what phases have been completed and used in a distributed way, along with the client or both. For example, if the requirements phase was performed in a distributed way, with a team in another region, geographically distant, if it was done at the customer site or was carried out both forms, on-site (together) and so distributed.

The Requirements phase was the most utilized onsite form, except for work W3, which was a project with requirements already set. The use of the onsite model in most jobs is because of the complexity in getting the customer to extract information about what problem he has and how he wants to solve.

Team Selection, Contract, Schedule and Release Planning phases were less used (except for P10). This can be explained by the two projects because they are specific stages in the development process used for the work, and

both have been done onsite and distributed. Analyze and Design phases are developed by most projects.
These steps are not performed by some models due the context of the situation, some with performed analysis and well-defined projects. Four projects used the stage of design so well distributed, and this is due to the fact that with an analysis done right, one can design.
The phase that was more developed in accordance to the distributed paradigm was Encoding. It is because it is only necessary to refer to the documentation that must be implemented and the teams who understand the pattern of such documentation implements the functionality without major problems.
The test phase was carried out by most of the projects and almost all made such a phase in a distributed and onsite manner. This is explained by the fact that some tests can be distributed performed and others need to be made together with the customer.
Quality Assurance was a phase that was conducted in only three projects. This is explained by the fact that the projects owned this phase as a specific activity to their development process, including work W3 which had focused on Quality Assurance.
Only three projects (W6, W7 and W8) have used the phase of Training Team Offshore, it should be because the projects involve other countries and the involved have experience in project management and distributed projects, and use specific and general tools and techniques. Thus the leaders realize the importance of training Offshore team.
Work W3 was the one who used the stage of Usability Evaluation, since it was focused on software quality and was only the adaptation of an existing tool. Thus, it only had three phases: design, which was performed in a distributed way and with the client, the test phase, wihch was conducted in a distributed fashion also and quality assessment which, in turn, was made in both ways.
Only three projects had the Deployment phase. These were performed with the client, but only one of them had also used distributed facets. This can be explained by the degree of difficulty to deploy a system in a distributed manner in order that one will deploy to the client.
Most studies found in systematic review concerns the description of the projects and their results. In other words, they are not papers that seek to identify models of development that exist and ways of collaboration. Thus, it is clear that the lack of work in this area is an incentive for researchers and organizations to invest in this segment, since it is still an open field without much exploration.

5 Related Works

Albeit with different focus and levels of detail, some works of literature present SLR contributing the evolution of DSD, to improve, assesses and provides the basis for professionals and companies in Software Engineering. Among those studies, some are listed in the next paragraphs.
The SLR proposed by [4], where it was selected 25 (twenty five) primary studies from 2001 to 2009, had identified 11 (eleven) models and 24 (twenty four) effective tools for managing DSD. Despite the satisfactory results, this review aim was the identification of models and tools used to support the DSD project management, where each identified model are different from those proposed herein. In this case this paper presents only results based on project management, without having direct relation to the developmental stages and their distributions
The SLR proposed by [13] had as its main objective to identify and summarize studies that describe the process models for distributed software development in the context of offshoring. There were found 27 (twenty- seven) primary studies that describe process models related to DSD stage. Only 5 (five) of these studies examined the outsourcing of a subsidiary company. This SLR does not intend any internal displacement; in addition, the proposed models should be analyzed and properly validated by an empirical method. Hence, more studies addressing the technical aspects and process models in DSD (due to part of the proposed models are related to a business perspective) are required.
The work proposed by [1], although with very similar data on primary studies proposed here does not deal specifically with DSD collaboration models between corporations, specifications of the phases and their distribution. It is just the specification of three basic phases and how they can be deployed (distributed) among the developer, forgetting the customer stakeholder. The focus of the work is in process models for requirements engineering for DSD with focus on: structural impacts, corporate management and technical requirements engineering process. Our work, rather, presents the collaboration models based in 4 phases of the development cycle and whether these phases are distributed or onsite.
The text below explains the work proposed by [14] had identified 19 (nineteen), models of collaboration through a survey with 38 (thirty eight) distributed projects. In this research the authors focus only on four development phases (Analysis, Design, Coding and Testing). Thus limiting the study, since each project is unique, and its context and characteristics vary from project to project. Given this context, it was realized that adopting this new form of software development, often motivated by budgetary pressures, seeking to adopt a strategy of reducing costs, is doomed by lack of awareness of the threats and risks to the project. In this investigation we conclude that the most

common type of collaboration found was outsourcing without the joint efforts of the teams. However the successful adoption of DSD software projects should be reasoned in analysis of many variables depending on the context of the organization, where the project is included. Based on empirical research and interviews, the study lists some recommendations for DSD projects. The Figure 4 presents the nineteen models found through this survey.

Nr.	Involvement of remote sites in software development activities				No of Projects
	System Analysis	Design	Implementation	Testing	
M1	jointly	jointly	jointly	jointly	3
M2	customer	customer	jointly	jointly	1
M3	customer	jointly	jointly	customer	2
M4	customer	customer	jointly	customer	1
M5	[no information]	[no information]	supplier	[no information]	2
M6	[no information]	[no information]	supplier	customer	1
M7	customer	customer	supplier	customer	5
M8	customer	customer	customer	supplier	1
M9	customer	customer	supplier	supplier	3
M10	customer	supplier	supplier	supplier	1
M11	customer	supplier	supplier	customer	5
M12	customer	customer	supplier	jointly	1
M13	customer	supplier	supplier	jointly	2
M14	jointly	jointly	supplier	supplier	3
M15	customer	jointly	supplier	jointly	2
M16	customer	jointly	supplier	customer	1
M17	customer	jointly	supplier	supplier	1
M18	jointly	supplier	supplier	supplier	1
M19	supplier	supplier	supplier	jointly	1

Figure 4: Different forms of work. Smite, D., Borzovs, Juris. (2009)

This work presents a survey that identified 19 different models out of 38 investigated projects. In addition, it has also gathered qualitative measurements of success and failure of each model; however, the conclusions cannot be generalized due to statistical insignificance. Nonetheless, investigation of different collaboration models supports conclusions on different forms of work and their variations based on the level of remote site involvement.

6 Final Remarks

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

This research presents a systematic review of literature to identify the models of collaboration for distributed environments that both the software industry and academia use. It was conducted from September 2009 until January 2010, involving 841 works published between 2000 and 2009. Its extension was conducted between December 2010 and February 2011, involving over 127 jobs.

Although this study followed a rigorous methodology, it has some limitations, such as the amount of studies of the Systematic Review of Literature, which took as its basis, only 9 papers. The small number of studies may be explained by immaturity in DSD field, since it emerged in the last two decades but the academy has only addressed works on the issue only in the last ten or twelve years. These models serve to further studies and for practical applications, since they can be used as reference. In this sense, practical and experimental approaches remain as a crucial study field for companies that are evaluating the best ways to work with DSD.

Some future work can be developed from this study, such as extending the research through manual searches in

Conferences and Journals or references from primary studies; further results analysis to identify and treat multisite models with a large discrepancy between the offshore and onsite developments; investigate additional models, and so on.

Finally, models of collaboration, processes and tools for DSD still need further research with respect to integration of features and larger coverage of the issues that are novel in DSD with respect to co-located development. 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.

References

[1] Berenbach, B. “Impact Of Organization Structure on Distributed Requirements Engineering Processes: Lessons Learned”, Workshop on Global Software Development for the Practioners at ICSE, Shangai, 2006.

[2] Binder, J, C. Global Project Management: Communication, Collaboration and Management Across Borders. Grower Publishing.

[3] Brooks, F. P. The Mythical Man-Month: Essays on Software. 1st Addison-Wesley Longman Publishing Co., Inc. 1978.

[4] Costa, C. et al. "Models and Tools for Managing Distributed Software Development: A Systematic Literature Review", In 14th International Conference on Evaluation and Assessment in Software Engineering. 2010.

[5] Domholdt E. Rehabilitation research: principles and applications. Missouri: Elsevier Saunders. 2005.

[6] Herbsleb, J. D. Global Software Engineering: The Future of Socio-technical Coordination. *IEEE Computer Science*. P188-198. 2007.

[7] Kitchenham, B.**;** Charters, S. Guidelines for performing Systematic Literature Reviews in Software Engineering. Vol 2.3 *EBSE Technical Report*, EBSE-2007-01, 2007.

[8] Meyer, B. The unspoken revolution in software engineering. *IEEE Computer.* 2006.

[9] Oates, J. B.; Capper G. Using systematic reviews and evidence-based software engineering with masters students. International Conference on Evaluation & Assessment in Software Engineering, EASE, 2009.

[10] Pichler, H. Be successful, take a hostage or outsourcing the outsourcing Manager. Proc. Second IEEE International Conference on Global Software Engineering, ICGSE, 156-161. 2007.

[11] Prikladnicki, R., Audy, J., Evaristo, R. "A Reference Model for Global Software Development," Proc. 5TH IFIP Working Conference on Virtual Enterprises, 2004, Toulosse. 2004.

[12] Prikladnicki, R., Audy, J. "Comparing Offshore Outsourcing and the Internal Offshoring of Software Development: A Qualitative Study," Proceedings of the Americas Conference on Information Systems (AMCIS), San Francisco, USA. 2009.

[13] Prikladnicki R., Audy, J., Nicolas L. Process models in the practice of distributed software development: A systematic review of the literature. Information & Software Technology 52(8): 779-791. 2010.

[14] Smite, Darja., Borzovs, Juris. New Forms of Work in the Light of Globalization in Software Development. Infoconomic for Distributed Business and Decision-Making Enviroments: Creating Information System Ecology. Business Science Reference – Blekinge Intitute of Technology, p346. 2009.

[15] The Standish Group. CHAOS 2001: A Recipe for Success. 2001.

[16] Biolchini, J., Mian, P.G., Natali, A.C.C., Travassos, G.H. “Systematic Review in Software Engineering”, Relatório Técnico, PESC, COPPE-UFRJ, ES-679/05, 2005.

Appendix: Primary Studies

W1 Gaurav Caprihan. (2006) Managing Software Performance in the Globally Distributed Software Development Paradigm, Proceedings of the IEEE international conference on Global Software Engineering, p.83-91, October 16-19.

W2 James Cusick, Alpana Prasad. (2006). A Practical Management and Engineering Approach to Offshore Collaboration, IEEE Software, v.23 n.5, p.20-29.

W3 Alvin W. Yeo. (2001). Global-software development lifecycle: an exploratory study, Proceedings of the SIGCHI conference on Human factors in computing systems, p.104-111.

W4 Rizwanjameelqureshi, M., Hussain, S. (2008). An adaptive software development process model. Advances in Engineering Software. Volume: 39, Issue: 8, Pages: 654-658.

W5 Setamanit, S., Wakeland, W., Raffo, D. (2007). Improving Global Software Development Project Performance Using Simulation. In: Portland Internacional Conference on Management of engineering and Technology Portland, OR, USA.

W6 Faiz, M. F., Qadri, U. Ayyubi, S.R. (2007). Offshore Software Development Models. Information and

Emerging Technologies (IIET 2007). pp. 1-6.

W7 Leszak, Marek., Meier, Mandref. (2007). Successful Global Development of a Large-scale Embedded Telecommunications Product. International Conference on Global Software Engineering (ICGSE 2007). pp. 23-32.

W8 Smite, D., Borzovs, Juris. (2009). New Forms of Work in the Light of Globalization in Software Development. Infoconomic for Distributed Business and Decision-Making Enviroments: Creating Information System Ecology. Business Science Reference – Blekinge Intitute of Technology, p346.

W9 Scharff, Christelle., Gotel, Olly., Kulkarni, Vidya. (2010) Transitioning to Distributed Development in Students' Global Software Development Projects: The Role of Agile Methodologies and End-to-End Tooling. Fifth International Conference on Software Engineering Advances. Nice, France.