The Distributed Software Development: Towards a Taxonomy

Fabio Levy Siqueira and Paulo Sérgio Muniz Silva
Departamento de Engenharia de Computação e Sistemas Digitais
Escola Politécnica da Universidade de São Paulo
São Paulo, Brazil
levy.siqueira@usp.br, paulo.muniz@poli.usp.br

Abstract

The distributed software development has many forms. Both a project executed by groups located in different buildings in a same city and a project executed by people scattered across different countries are distributed software developments. But each of these situations has different motivations, benefits, and also problems. Seeking to classify the distributed software development projects according to its problems and advantages, this paper presents the basis and the challenges to create a DSD taxonomy.

1. Introduction

Traditionally the software development team works in a common environment that allows a greater flow of information between the people involved [6]. This common environment is so important to the software development that De Marco and Lister [8] discuss the ideal work place, suggesting a set of patterns for the environment's project. Showing a greater concern and importance for this matter, the software development process can even suggest a facilities strategy as, for instance, in the Extreme Programming [3].

Although a common environment is an important issue, it is not always possible to confine all the people involved in the software development in only one place. The complexity of a project may require various companies spread over one city; the search for an expert – either a person or even an organization – can lead to other country states; the competitive need for high qualified but low cost labor may involve other countries. In these situations, where the personnel are separated through time and space, it is necessary to execute another kind of software development: the distributed software development (DSD).

At a first sight, a distributed software development project may seem easy to execute. The various technological advances have shortened the distance between the workers. The new forms of communication allow high speeds, with low delays, and almost ubiquitous access to corporative information. The innovations in collaborative tools ease the group work, making a virtual interaction possible. Beyond the technical advances, there are also some strategies, and technical and managerial solutions trying to lessen the problems and conflicts in this kind of projects. Even so, the DSD is still a challenge: as much as there are projects that succeed there are several that fail.

The practical experience shows that the space and time separations deeply affect the software development, making its execution more difficult due to reasons that vary from the absence of a common idiom and work time, to the lack of confidence and sense of teamwork among the people involved. But not all DSD projects will suffer from the same problems or even will benefit from the same advantages. The term distributed software development comprises different situations, including cases where people are distributed in groups located in different buildings in a same city and cases where the people involved are completely dispersed throughout the world. Observing these two distribution possibilities, it is perceptible that the DSD has different forms, each one demanding different software processes, involving different risks, and requiring different project management strategies.

To organize and understand the difference involved in DSD projects this paper discusses the basis and the difficulties for a DSD taxonomy. The goal of this taxonomy is to classify distributed software development projects based on its problems, allowing the comparison between the different difficulties that exists in DSD projects. With this kind of taxonomy it will be possible to foresee possible problems in a project and take actions to prevent or manage them.

To create this taxonomy it is proposed a method based on the definition of a set of characteristics of the DSD that represents the source of the related problems and advantages involved in a DSD. For each of the characteristics it is also proposed a set of variables, seeking to create a set of metrics to evaluate the elements of the taxonomy.

The paper is structured as follows. Section 2 presents the theory involved in the DSD and its related terms. In section 3 it is presented the proposed method to create the taxonomy. To understand the research context, section 4 presents the related works. In section 5 it is presented the foundation of the criterion to classify the projects, based on a set of characteristics of the DSD and its variables. Afterwards, it is presented in section 6 the comparison of three case studies using the proposed characteristics and variables, discussing some difficulties. In section 7 it is presented the main challenges in creating a taxonomy. Finally, in section 8 it is presented the conclusion with some perspectives for future works.

2. The distributed software development

There are many motivations for a distributed software development. Probably the main motivation is the access of low cost labor. In developing countries it is possible to find high-qualified workers with salaries lower than those paid in countries like the United States and Japan [5]. But it is not always the search for a low cost labor that motivates a DSD. Sometimes it is the opposite: the difficulty to find an expensive, qualified, and talented resource. A problem or a technical constraint may require a specialist (organization or person) located in another city or even a different country. Moreover, there are many other motivations for having a DSD, such as to shorten the schedule by taking advantage of the time-zones to have a 24-hour software development [11]; to have a part of the team closer to the software client or user ; and to break the team in different places because of the size of the project. Many of these motives for having a DSD can be present in a project. However, the mere existence of these motivations may not be enough for having a DSD. Each organization has to analyze the real need and the cost-effectiveness of a DSD in a project.

As much as there are many motivations for having a DSD, there are also many problems related to this kind of software development. According to Carmel [5], the main problems in the DSD are: geographical dispersion, loss of communication richness, coordination breakdown, loss of "teamness", and cultural differences. But besides these problems, it is possible to observe innumerous other problems in experience reports found in the literature as: the difficulty of obtain visa [19], the lack of common working times [17], differences in tool technical support and global sales [2], the lack of trust among the people [4], the delay on solving problems [17], and many others.

Although there are innumerous motivations and problems in a DSD, many of these advantages and difficulties are directly associated with some forms of this kind of software development. The term distributed software development can be used to express a great variety of situations. Probably because of that there are many terms related to the DSD. Some of these terms are presented below.

- **Global software development**. The distributed software development involving groups of people located in different countries is known as global software development [5] [12].
- **Disperse software development**. The term disperse software development is used when workers are distributed into small groups (of one or two) in each of the workplaces [1] [30]. This term is usually employed by agile methods enthusiasts (it is also used the term *distributed agile* to represent the use of agile methods in the DSD), although a characteristic of the DSD seems to contradict a main principle of agile methods: the importance of face-to-face communication. More than that, a common recommendation to work in DSD is to increase the process ceremony [20], which is the opposite to what the agile methods propose. Even so, there are some solutions to this kind of DSD as in [18] [28].
- **Open-source development**. In an open-source development the user is a potential programmer as the source code is open (can be read and written) and available (acquired free of charge or with a nominal fee) [13]. As the software is normally published on the Internet, its code is available to everyone, everywhere. That way, the users and, consequently, the developers can be geographically dispersed - making the open-source development a kind of DSD. But its differences in philosophy, economic and team structure models makes it a very particular DSD [6].
- **Outsourcing**. According to Pressman [24] outsourcing is the contract of a third party to do a software activity with lower cost and higher quality. Although outsourcing may be seen as a common reason for having a DSD, it is important to notice that outsourcing is not always a DSD. In some situations the contractor may work in the same place as the client, avoiding the problems of the geographical distance in a project. Similarly, DSD is not always outsourcing. There are some other relations between organizations that cannot be classified as outsourcing, such as joint ventures, strategic partnership, and projects inside global companies. Also there are other possibilities as telework, company teams distributed over different places, and open-source development, which can be seen as DSD, but not as outsourcing.
- **Virtual teams**. Teams geographically and (or) organizationally linked through telecommunications and information technologies are known as virtual teams [29]. In the software development context, virtual teams are not always DSD. According to Lipnack and Stamps [22], virtual teams can vary between time-space and organization dimensions (only the team in the same space and within the same organization is not a virtual team). That way, even teams sharing the same workplace but having workers from different organizations are virtual teams.

- **Telework**. The popularization of the computer and the advance of the communication and collaboration technologies allow working everywhere, either at home, at a client's, in a hotel or even in the car. This modality of work is known as telework [26] [10].

Figure 1 depicts the relation between these terms, noting that some of them represent a subset of the DSD (global software development, disperse software development, and open-source development) and others are related to the team involved in a DSD (outsourcing, virtual teams, and telework).

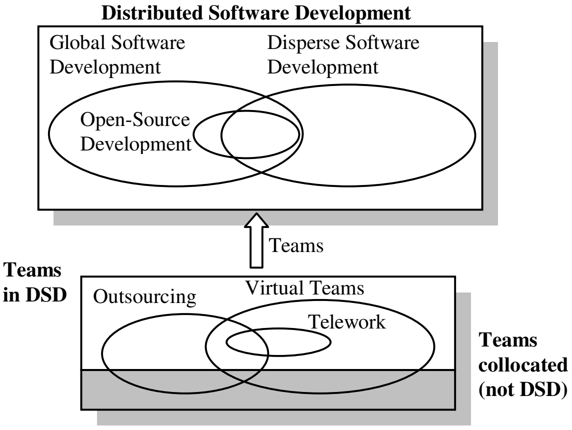
Distributed Software Development
Global Software Development
Disperse Software Development
Open-Source Development
Teams
Teams in DSD
Outsourcing
Virtual Teams
Telework
Teams collocated (not DSD)

Figure 1: The terms related to the DSD.

Frequently the meaning of these terms is confused as the criteria used to differentiate some of them are sometimes vague [7]. Also, it seems that there is not a consensus in the literature as to what some of these terms mean – even about the term DSD. In this paper, the term DSD is employed in a broad sense: *the distributed software development is the software development that is executed by people not collocated.* People in different countries working together in a software development are clearly in a DSD. But that definition can be vague as the physical distance decreases. In a borderline, the people can even work exactly in the same place and still be a DSD: not collocated can also mean a time separation as, for instance, when people work in different shifts. The important issue when considering the physical and temporal distance seems to be the difficulty to work together brought about by these separations occurring in the software development. The management has to decide, according to the project characteristics, whether the software development executed by people separated by 10 meters is better seen as a DSD.

3. The method to create the taxonomy

The unique common aspect in any distributed software development is the fact that people are not collocated. However this separation can be manifested in several ways, which makes the distributed software developments different. Depending on how the separation is, the problems that occur in a certain project may not exist (or even be immaterial) to another project. Similarly, the solutions successfully applied in a project may not be as useful in another project.

To transfer the possible problems and solutions from a project to another it is necessary to have some form of classification of the distributed software development. For this purpose it is proposed to create a taxonomy for the DSD. The main goal of this taxonomy is to classify the DSD projects based on its problems and advantages, making possible the organization of the knowledge acquired in similar distributed software developments in order to make a project decision.

One possible way to create this taxonomy could be the organization of the different terms used to denote the DSD and group them according to its problems and advantages. However, as discussed before, there is not a clear separation between all these terms frequently employed. Therefore, it is proposed that analyzing the problems and advantages present in DSD projects should create the taxonomy and, from them, define the types of DSD. This way, each element of the taxonomy would represent types of projects that share similar problems and advantages.

The main point to create a taxonomy with this goal is the definition of a classification criterion. This criterion would allow the definition of the constituent elements of the taxonomy and, more than that the classification of a specific DSD project in the taxonomy. But the definition of this criterion requires a better understanding of what the DSD means. Based on this knowledge, it is possible to create an effective approach to evaluate projects.

Seeking to obtain a taxonomy for the DSD, in this work it is proposed the basis for a classification criterion. This criterion is based on the definition of a set of variables that represents the source of the problems and advantages in the DSD, called the DSD characteristics. Each of these characteristics is refined in a set of variables, with which it is intended to obtain the metrics to evaluate the elements of the taxonomy. Therefore, in evaluating several projects using these metrics and correlating the actual problems with the values obtained, it will be possible to find several types of distributed software developments that could be organized in a taxonomy of the DSD.

4. Related Works

Some authors have already attempted to classify the DSD based on some characteristics of the DSD. Cockburn [6] considers the size of the team, the roles found in each place, and also the type of the project (open-source or commercial) to propose a simple classification of the DSD in 4 groups: multisite, offshore, distributed, and software open-source. To create a classification of distribution levels, Prikladnicki et al. [25] consider the external and internal physical separation of some groups of project stakeholders. Another classification of the DSD is proposed by Kobitzsch et al. [19], when identifying four major cooperation models: separate teams in basically independent companies, separate teams in legally related companies, one team distributed across multiple sites of legally related companies, and one team distributed across multiple sites of several basically independent companies. To create this classification, Kobitzsch et al. consider only the legal relation and the team configuration existent in the DSD.

The main problem in all these classifications is that they are only concerned with a small set of dimensions of the DSD. Some other authors try to define, with different level of detail and goals, what the dimensions of the DSD are. For example, Carmel [5] points out what differentiates the global software development from a traditional one: the physical distance, the time-zones, and the national culture. However a characterization in three dimensions seems to be restricting when observing some other works that considers a broader variety of characteristics.

A multidimensional view that tries to obtain a broader division into the characteristics of the DSD is proposed by Evaristo and Scudder [9]. This division has the focus on the project management and seeks the variables that influence the project's performance. The proposed dimensions are: type of project, structure, perceived distance, synchronicity, complexity, culture, information systems, methodology, existence of policies and standards, level of dispersion, and stakeholders. As the main concern of this division is the complexity of the project management, it considers some characteristics that are not necessarily related only to the manner on which the people are distributed. The division also considers the nature of the project (type of the project) and the difficulties caused by the number of people involved (complexity and stakeholders).

Another characterization in a set of dimensions is proposed in [30]. In this division it is presented some aspects that are out of the developers' control: distance separating developers, time-zone, culture, broadband availability, scale of project, size of sub-teams, development for product or for project, external or internal customer, reason for dispersed working, corporate culture and political constraints. But this definition is very brief, as the main concern is the definition of the scenarios of dispersion. But, as in Evaristo and Scudder's division, some of these variables consider some other aspects of the software development that are not exactly related to the DSD (scale of project, development for product or project, and external or internal customer).

5. The proposed classification criterion

Seeking to obtain a more precise classification for the DSD, this paper presents a set of dimensions that try to represent the source of the advantages and problems of the DSD, called the DSD characteristics. From these characteristics it is intended to define metrics to evaluate DSD projects in face of its problems and advantages and properly group them into elements of the taxonomy.

The creation of the DSD characteristics considered the various proposals to define the dimensions of the DSD that exist in the literature. Moreover, it was made a bibliographical research observing the DSD project's problems and advantages in several experience reports and discussions about this subject. From these sources it was observed the main problems and advantages of software developments of this kind, and the several possible options of the DSD, arriving to the proposed characterization.

The DSD characteristics are organized into three main categories: the form of group separation (*grouping*, *physical distance*, and *temporal separation*), the involved regions (*regional culture*, *idioms*, and *local differences*), and also the participant organizations (*organization culture*, *infra structure*, and *business relation*). The critical point that defines the distributed software development is the form of group separation. However, it is very common that the people are located in different regions (different cities, states, and countries), and also that the groups geographically separated are from different organizations. Consequently, the other two categories (involved regions and participant organizations) are indispensable to understand the DSD, and were considered in the proposition of the characteristics.

Each of the characteristics seeks to represent a source of problems and advantages of the DSD. Nevertheless it seems to be difficult to directly obtain values from the DSD characteristics, as each of them seems to be composed of variables. In the grouping characteristic, for instance, the statement that there are two groups involved in the software development only gives a notion of how the stakeholders are separated but it is insufficient to understand the existent problems. Adding to this characterization that there are 10 people in each place helps to understand the distribution, but it does not represent what each people or group does. For example, a DSD in which the customer is in one place and the engineering group is in another place is completely different from a DSD in which the customer and the

Table 1. DSD characteristics and its variables.

Characteristics	Grouping	Physical distance	Temporal distance	Regional cultures	Idiom	Local differences	Organizational Culture	Infra structure	Business relation
Variables	• Roles • Number of people • Number of groups	• Distance • Difficulty to communicate face-to-face	• Time-zones • Work hour	• Power distance • Individuality • Masculinity • Uncertainty avoidance • Long-term orientation	• Proficiency • Semantic differences	• Law • Calendar • Local structure	• Artifacts • Exposed values • Basic assumptions	• Hardware • Software • Tool • Technique • Standard • Facilities	• Legal relation

analysis team are in one place and the rest of the engineering team is in another place. Therefore, the grouping seems to be composed not only by the number of groups involved, but also by the number of people and the software process roles existing in each group.

Hence, each of the characteristics is developed in some elements that represent the variables involved in this characteristic. When considering these variables it seems to be easier to obtain a set of metrics to evaluate the characteristics. Table 1 shows a summary of the DSD characteristics and its respective variables.

Next, it is presented the proposed DSD characteristics, observing its definitions and the variables that seem to compose it.

5.1. Grouping

The grouping represents the form into which the stakeholders in a DSD can be separated.

Some authors have already discussed the grouping, normally relating it to others characteristics of the DSD. Cockburn [6], when classifying the DSD, considers the stakeholders communication needs and represent the group distribution mainly by the group size in each place and by the division of the stakeholder's roles. In another work, Prikladnicki et al. [25] proposes criteria to represent the level of the organization distribution, seeking to highlight the importance of the user and client in the DSD. Thereby, Prikladnicki et al. propose a model that observes the physical separation between the involved actors (inter-group) and, inside the group of actors (intra-group), between the project team, the client, and the user.

Considering the ideas to represent the grouping presented by Prikladnicki et al. and Cockburn, the grouping seems to be composed of three elements: *the roles played by the people*, *the number of people in a group*, and *the number of groups*.

5.2. Physical distance

The physical distance represents the physical separation between the groups involved in the software development.

This characteristic seems to be formed by *the distance between the people* and *the difficulty to communicate face-to-face*. In some cases, travel to meet other people can be excessively expensive, lengthy, or fatiguing, which can significantly hamper the holding of face-to-face meetings. Considering the difficulty to have a collocated communication, one proposed degree for the physical distance is described in [30], and similarly in [25]:

- *Different compartment scenario*: the team members are in physical proximity, but separated.
- *Cross town scenario*: there is a long distance for the team members to meet face-to-face.
- *No time-shift scenario*: a separation of 3 to 6 hours in a same or close time-zone.
- *Continental*: intra-continental separation (maximum of 3 to 4 hours of time-zone).
- *Transglobal*: all distances that need 24 hour or more for the people to work collocated.

5.3. Temporal distance

The temporal distance represents the existence of different work hours between the groups.

The groups of people that are involved in a DSD may be spread all around the world, separated by long physical distance. Depending on the distance between the places, this physical separation may also bring about a temporal separation through the differences in *time-zones*. It is important to highlight that the differences in time-zones are not exclusive to global software development. Countries with a great longitudinal extension have inside its border many time-zones as, for example, the United States of America and Russia.

Besides the time-zone, there can be also a difference on *work hours* between the people. The existence of small differences in work time, something like 1 or 2 hours, is very common in software development organizations and causes few problems. But when this difference gets big enough, something like the occurrence of different work shifts, the difference in work hours become a limit case of the DSD where people work in the same place but in different hours.

5.4. Regional cultures

The regional culture represents the difference of values and practices that happens between people from different regions.

A possible definition of the regional culture elements can be observed in a set of dimensions that represents the actual cultural differences. Hofstede [14] proposes five dimensions to aid the identification of cultural differences between countries:

- *Power distance*: the degree in which the members with less power accept an unequal division of power in an organization or institution within a country.
- *Individualism and collectivism*: represents the individual and group link. In societies more individualistic people tend to think mainly in themselves, while people in collectivistic societies favor the group.
- *Masculinity and femininity*: the differences between the emotional sex roles in a society.
- *Uncertainty avoidance*: represents the ambiguity acceptance in a culture, indicating the degree in with the people is comfortable with ambiguous or unknown situations.
- *Long- and short-term orientation*: these are the values associated with greater concern about the present – short term – and the opposite, the vision of the future – long-term.

A culture conflict will not necessarily arise or be important when a difference in Hofstede's dimensions between the countries involved exists. On the other hand, even a relationship between people of one same country can suffer from cultural problems. The Hofstede's abstraction in a national extent is a very interesting tool to analyze the problem, but maybe a less general view, inasmuch as treating culture as a regional aspect may be more appropriate.

5.5. Idiom

The idiom represents the difficulty of expression and understanding of a language.

The simple expression in an idiom may not be enough for a team member in a project. It is necessary that he/she has a reasonable degree of proficiency to allow a dialog without problems to understand and to be understood, and also to avoid semantic and syntactic problems when communicating. Even so, it is possible that problems in interpretation may occur, even between people who speak the same idiom, due to differences in education and culture [23]. Thus, the idiom seems to be composed of two variables: *the proficiency* and *the semantic differences*.

5.6. Local differences

The local differences represent the difficulties caused by the group location in a certain region.

The local differences demand to understand the region where the group is located, something beyond the culture and the idiom. These differences represent several types of problems. The three main variables seem to be: *the law* (for instance, import and export rules [2], official business year [19], and rules to obtain a visa [19]), *the calendar* (as in different off-days and holidays [19]), and *the local structure* (availability of reliable means of communication, and energy, for instance).

5.7. Organizational culture

The organizational culture represents the differences in strategies, purposes, philosophies, beliefs, perceptions, thoughts, and feelings that are originated and shared by the organization where the people work.

In an organization, this form of culture is expressed in several ways. According to Schein [27], there are three distinct levels of organizational culture: *the artifacts*, *the exposed values*, and *the basic assumptions*. The artifacts are the more visible elements of the culture, as the work environment, the rituals, the myths, the products, the technology, the stories, and the common language of the company (some of these elements are viewed as the infrastructure of the organization). Although evident, the artifacts are hard to understand, as it is not possible to observe their real meaning. This understanding of the artifacts can be obtained when observing the exposed values, a higher abstraction level of the organizational culture on which the group members base their behavior. The exposed values are the strategies, the objectives, and the philosophies followed by the workers from somebody's rationalization in order to solve a problem or a doubt. When this solution works repeatedly, some times it becomes a basic assumption that is the main point, the essence of the organizational culture. The basic assumptions represent the set of unconscious ideas shared by the people and that are hardly discussed between them, but assumed as true and employed without hesitation.

5.8. Infrastructure

The infrastructure represents the differences in the available infrastructure in the involved organizations.

Every software development requires an adequate hardware to perform the necessary work, such as computer stations to be used by developers or by the physical part of the system being built. Similarly, a set of tools and software are necessary: operational systems, development environment, compilers, text editors, web browsers, etc. However the infrastructure is not only *hardware*, *software* and *tools*. Using the standard ISO/IEC 12207 [15] as a reference, the infrastructure also covers *techniques*, *standards*, and *facilities* involved in the development, operation, and maintenance of software products.

In a DSD the existence of an adequate infrastructure for all stakeholders may be hard to be obtained. Inside a company problems may happen if its employees work at home, hotels, or telework centers. When more than one organization is involved, this difficulty can be greater. The organizations can work with different versions of tools, data standard [12], notation standard, techniques to specify requirements [21], operational systems, code standard, development environment, and many other differences that can make the coexistence of different infrastructures a big challenge.

5.9. Business relation

The business relation represents the relation between the organizations and the people involved in the project.

In DSD projects it is very common that several organizations work together, sharing the development activities. The companies can be linked through a contract (as in an outsourcing), or be a part of one same company (as a main office and a branch), or even some other form of relationship – as strategic partnership (alliance of companies to develop and promote products [16]), or joint ventures (relationship between companies that creates a new entity [16]). Depending on the *legal relation* in a project, there will be some differences and problems specifics to the software development.

6. Case studies

To observe how the DSD characteristics and its respective variables can exist in real projects, three distributed software development projects are discussed. Project A was an academic endeavor to build a collaborative software, involving people who met frequently, but worked separated. Project B involved researchers and scholarship students to build a software in the area of astrophysics. Project C was a big project to develop a framework to build financial applications for a large financial institution.

The information about these projects was obtained either through the analysis of the project by using the experience obtained in its development (projects A and C), or either by applying a questionnaire and instantiating the characteristics (project B). Table 2 shows how the DSD characteristics are instantiated in the three projects. It is important to emphasize that the values defined in this table only serve to illustrate the differences between the projects (they are not definitive metrics).

It is possible to observe through the table that although the projects are distributed software developments, they are very different. In project A the software development was almost not affected by the DSD, only requiring some minor changes in the process employed and in the infrastructure used, while the projects B and C have been greatly influenced by the DSD. In project B the DSD caused several difficulties as some conflicts of infrastructure, different philosophies and objectives in the methods (from the differences in organizational culture), and the difficulty to work through distance. In project C almost every characteristics caused some kind of problem. Some of the problems were similar to those verified in project B, as the organizational culture differences. But the project C presented other particularities that led to other problems as, for instance, the regional culture and idiom that caused conflicts and misunderstandings, and the grouping that had primarily influenced the discussion between the different groups of stakeholders.

In the three case studies it is possible to observe the difficulties to precisely define a range of values for the considered characteristics. Each project is completely different from the other but, apparently, there are similarities in some characteristics as the regional culture and local differences in projects A and B. In some other characteristics it seems to be difficult to affirm if the characteristics are sufficiently similar, as in the temporal separation of the projects B and C. In other cases as, for instance, the grouping for the three projects, it seems to be evident the existing differences.

Besides the difficulty to compare projects, there are some doubts about the adequacy of some variables selected for some characteristics. For instance, it is difficult to say that the similarity of all variables for the organizational culture means that the projects B and C share the same problems. This kind of problem is not evident for the regional culture characteristic, since only one project has suffered from this difference. However it also seems difficult to state that this choice of variables is enough to organize the problems, moreover if a project involves more than two regional cultures.

From the analysis of these case studies with the DSD characteristics and its variables, it appears to be evident that there are still several problems to define values for some variables. Even so, the initial application of these characteristics and variables allowed a deeper comparison between the projects. This fact motivates a refinement of these variables aiming a more precise definition of a criterion to create a taxonomy for the DSD.

Table 2: The characteristics in three case studies.

Characteristics	Variables	Project A	Project B	Project C
Grouping	Roles	all disperse	almost a division by roles	division by roles
	Number of people	1in each place (4 total)	2 or 3 in each place (9 total)	25-30 in each place (80 total)
	Number of groups	4 groups	3 groups	3 groups
Physical distance	Distance	same city	in different Brazilian states	two sites in different Brazilian states and the other site in another country
	Difficulty to communicate face-to-face	saw each other frequently	meetings every 2 months	budget to travel frequently
Temporal separation	Time-zones	inexistent	inexistent	great time-zones differences
	Work-hour	same	various, depending of the availability of the person	basically the same in each work place
Regional Culture[1]	Power distance	same	same	almost the same
	Individuality	same	same	almost the same
	Masculinity	same	same	different
	Uncertainty avoidance	same	same	different
	Long term orientation	same	same	almost the same
Idiom	Proficiency	same idiom	same idiom	not everyone was proficient
	Semantic differences	none	few	many
Local differences	Law	same	same	different
	Calendar	same	same	different
	Local structure	same	same	different
Organizational culture	Artifacts	same	different	different
	Exposed values	same	different	different
	Basic assumptions	almost the same	different	different
Infrastructure	Hardware	same	different	same
	Software	same	same	same
	Tools	same	same	same
	Technique	almost the same	different	different
	Standard	same	different	same
	Facilities	same	different	Same
Legal relation	Legal relation	academic	personal agreement with some people and a formal (scholarship) with others	partnership between organizations

7. Challenges to create the taxonomy

There are several challenges to create a DSD taxonomy. It is difficult to form generalizations and create a classification criterion, since there is a great variety of possibilities for the DSD. A deeper analysis of the proposed elements is still required, observing a larger number of projects and also of experience reports in the literature. For some characteristics the range of possible values seems to be almost mature, as for the physical distance when considering the scenarios of dispersion proposed in [30] and similarly by Prikladnicki et al [25]. However for others DSD characteristics, the definition of the range of values appears to be difficult to be obtained, e.g. for the organizational culture and for the local differences.

But maybe the bigger challenge to create a DSD taxonomy is the multidisciplinarity of the subject. Although the DSD is a type of software development, the technical challenges are only a small source of problems. Others areas of knowledge directly influence the project as: the social sciences (grouping, idiom, and regional culture), administration (organization culture), law (local differences and business relation), and others. Although this is the biggest challenge of the DSD, maybe this is also the most interesting aspect of this kind of software development. The research of the characteristics of DSD will make possible to advance the understanding of what the software development really means.

[1] The values for this characteristic were assumed based on the values presented in [14]: “same” when the values were equal, “almost the same” when the countries have different values but the same orientation, and “different” when the countries have different orientation.

8. Conclusions

The DSD is a vast subject with still little research considering its importance and complexity. The tendency is that the organizations will employ more frequently this kind of software development, wishing to take advantage of the available motivations and benefits. Progresses in the communication technologies, support tools, and solutions in theory and in practice, will allow a better management of the actual problems, making this kind of software development even more attractive and important

But to work in a more adequate way in the DSD it is required a more precise understanding of what this kind of software development means. It is necessary to understand the problems to create reasonable solutions.

This paper contributes to the advance in the understanding of what the DSD means by proposing the basis for a taxonomy of this kind of software development. It is proposed the use of a set of DSD characteristics to create a classification criterion of this kind of projects.

In future works it is intended to detail each one of the characteristics and its respective variables, observing a larger number of DSD projects and experiences reports.

References

[1] Ambler, S. Bridging the Distance. Software Development, September 2002. Avaiable at: <http://www.sdmagazine.com>.

[2] Battin, R. D.; Crocker, R.; Kreidler, J.; Subramanian, K. Leveraging Resources in Global Software Development. *IEEE Software*, v.18, n.2, p.70-77, March/April 2001.

[3] Beck, K. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999.

[4] Boland, D. Fitzgerald, B. Transitioning from a Co-located to a Globally Distributed Software Development Team: A Case Study at Analog Devices Inc.. In: *International Workshop on Global Software Development*, 3., Edinburgh, 2004.

[5] Carmel, E. *Global Software Teams: Collaborating Across Borders and Time Zones*. Prentice Hall, 1999.

[6] Cockburn, A. *Agile Software Development*. Addison Wesley, 2002.

[7] Dafoulas, G.; Macaulay, L. Investigating Cultural Differences in Virtual Software Teams. *Electronic Journal on Information Systems in Developing Countries*, v.7(4), p.1-14, 2002.

[8] De Marco, T.; Lister, T. *Peopleware: Productive Projects and Teams*. 2nd ed. Dorset House Publishing, 1999.

[9] Evaristo, J. R.; Scudder, R. Geographically Distributed Project Teams: A Dimensional Analysis. In: *Hawaii International Conference on System Sciences,* 33., Hawaii, 2000.

[10] Grozdanovic, M.; Pavlovic-Veselinovic, S. Framework for Teleworking. In: *International Conference on Telecommunications in Modern Satellite, Cabke and Broadcasting Services*, 5., Yugoslavia, 2001.

[11] Haywood, M. Working in Virtual Teams: A Tale of Two Projects and Many Cities. *IT Professional*, v.2, n.2, p.58-60, March/April 2000.

[12] Herbsleb, J.; Moitra, D. Global Software Development. *IEEE Software*, v.18, n.2, p.16-20, March/April 2001.

[13] Hissam, S.; Weinstock, C. B.; Plakosh, D.; Asundi, J. Perspectives on Open Source Software. *Technical Report CMU/SEI-2001-TR-019*. Carnegie Mellon Software Engineering Institute, 2001.

[14] Hofstede, G.; Hofstede, G. J. *Cultures and Organizations: Software of the Mind*. McGraw-Hill, 2005.

[15] ISO – International Organization for Standardization, ISO/IEC 12207 – Information Technology – Software Lifecycle Processes. ISO, 1995.

[16] Karolak, D. W. *Global Software Development: Managing Virtual Teams and Environment*. IEEE Computer Society Press, 1998.

[17] Kiel, L. Experiences in Distributed Development: A Case Study. In: *International Workshop on Global Software Development,* Portland, 2003.

[18] Kircher, M.; Jain, P.; Corsaro, A.; Levine, D. Distributed Extreme Programming. In: Marchesi, M.; Succi, G.; Wells, D; Williams, L. (eds.) *Extreme Programming Perspectives*. Addison Wesley, 2002, p.553-568.

[19] Kobitzsch, W.; Rombach, D.; Feldmann, R. L. Outsourcing in India. *IEEE Software*, v.18, n.2, p.78-86, March/April 2001.

[20] Kroll, P.; Kruchten, P. *The Rational Unified Process Made Easy: a Practitioner's Guide to the RUP*. Addison-Wesley, 2003.

[21] Layzell, P.; Brereton, O. P.; French, A. Supporting Collaboration in Distributed Software Engineering Teams. In: *Asia-Pacific Software Engineering Conference*, 7., Singapore, 2000.

[22] Lipnack, J.; Stamps, J. *Virtual Teams: Reaching Across Space, Time and Organizations with Technology*. John Wiley & Sons, 1997.

[23] Mockus, A.; Herbsleb, J. D. Challenges of Global Software Development. In: *International Software Metrics Symposium,* 7., London, 2001.

[24] Pressman, R. S. *Software Engineering: A Practitioners Approach*. 5th ed. McGraw-Hill, 2001.

[25] Prikladnicki, R.; Audy, J.; Evaristo, R. Distributed Software Development: Toward an Understanding of the Relationship Between Project Team, Users and Customers. In: *International Conference on Enterprise Information Systems,* 5., Angers, 2003. Proceedings, v.3. p.417-423.

[26] Ryan, J. The Business Strategy Behind Teleworking. *Technology Guide*. 2000. Avaiable at: <http://www.techguide.com>.

[27] Schein, E. H. *Organizational Culture and Leadership*. 2nd ed. Jossey-Bass Publishers, 1992.

[28] Schümmer, T.; Schümmer, J. Support for Distributed Teams in eXtreme Programming. In: Succi, G.; Marchesi, M. (eds.). *Extreme Programming Examined*. Addison Wesley, 2001, p.355-378.

[29] Townsend, A. M.; DeMarie, S. M.; Hendrickson, A. R. Virtual teams: technology and the workplace of the future. *Academy of Management Executive*, v.12, n.3, p.17-30, 1998.

[30] Wills, A. C. Dispersed Agile Software Development and Dispersed eXtreme Programming. Wiki Site. Avaible at: <http://www.fastnloose.com/cgi-bin/wiki.pl/dad>.