Journal of Universal Computer Science, vol. 18, no. 19 (2012), 2686-2705
submitted: 30/3/12, accepted: 6/9/12, appeared: 1/11/12 © *J.UCS*

An Integrated Approach of Software Development and Test Processes to Distributed Teams

Gislaine Camila Lapasini Leal
(Production Engineering Department - State University of Maringa, Maringa Brazil
gclleal@uem.br)

Ana Paula Chaves
(Federal Technological University of Parana, Campo Mourao, Brazil
anachaves@utfpr.edu.br)

Elisa Hatsue Moriya Huzita
(Computer Science Department - State University of Maringa, Maringa, Brazil
emhuzita@uem.br)

Marcio Eduardo Delamaro
(University of Sao Paulo, Sao Carlos, Brazil
delamaro@icmc.usp.br)

Abstract: The Distributed Software Development (DSD) is a development strategy that meets the globalization needs concerned with the increase productivity and cost reduction. However, the temporal distance, geographical dispersion and the socio-cultural differences, increased some challenges and, especially, added new requirements related with the communication, coordination and control of projects. Among these new demands there is the necessity of a software process that provides adequate support to the distributed software development. This paper presents an integrated approach of software development and test that considers distributed teams peculiarities. The approach purpose is to offer support to DSD, providing a better project visibility, improving the communication between the development and test teams, minimizing the ambiguity and difficulty to understand the artifacts and activities. This integrated approach was conceived based on four pillars: (i) to identify the DSD peculiarities concerned with development and test processes, (ii) to define the necessary elements to compose the integrated approach of development and test to support the distributed teams, (iii) to describe and specify the workflows, artifacts, and roles of the approach, and (iv) to represent appropriately the approach to enable the effective communication and understanding of it.
Key Words: global software development, software process, test
Category: H.2, H.3.7, H.5.4

1 Introduction

The software process is defined as an orderly set of activities to management, development and software maintenance, and should be aligned with organizational conditions [Fuggetta 2000].

The Distributed Software Development (DSD) is an approach for software development that comes to meet of globalization need, those are: increase of productivity; quality improvement, better resources allocation and costs reduction [Herbsleb et al. 2000]. This new configuration added to software development challenges related to cultural differences, geographic dispersion, coordination and control, communication and team spirit, which intensified some problems found during the project lifecycle.

The DSD features added influences on the way with the software is projected, developed, tested and delivered. According to [Damian and Lanubile 2004] to minimize these effects and improve productivity, are necessary new technologies, processes and methods appropriate for distributed development approach.

In [Jimenez et al. 2009] is that there was an increase in the interest of researchers for DSD. However much effort is devoted to dealing separately on human resources, organizational management, infrastructure and project management. This shows that although there are advances with respect to adopt DSD practices, there is still a gap regarding to technical and design aspects. Therefore, demand for adequate techniques, tools and process.

The purpose of this paper is to present an integrated approach to software development and test that offers support to development global projects. This support is offered by contemplate its peculiarities and, with this, improving the communication among teams, minimizing the artifacts ambiguity and offer to stakeholders, better visibility of artifacts and activities.

2 Methodology

The research carried ut is featured as quali-quantitative and exploratory type. The work followed the methodology proposed by [Mafra et al. 2006], which has the objective to enable the development and technologies maturation since its conception in academy until its deployment for the industry.

The Mafra's methodology has two phases: Definition and Refinement. During the Definition phase is conducted systematic review to identify evidences available in literature and thereby minimize difficult and uncertainties in definition process. The systematic review followed the protocol of [Biolchini et al. 2005] and aimed to identify papers that addressed the software development process when it considered distributed teams. The our systematic review enabled to identify the software processes that have been used with dispersed teams. The main selected studies were related to problems founded when DSD is adopted [Leal et al. 2010] or with evaluation carried out considering practices or specific aspects of DSD [Patil et al. 2011] [Boden et al 2011].

Based on the evidences found changes and improvements are proposed. So, after that feasibility study, the observational study and case studies are performed. These are carried out both in the context of the whole life cycle of the proposed technology as well as in the industrial context. In the Refinement phase, a feasibility study on the proposed technology is realized. With this, the goal is to create a body of knowledge and do not necessarily get a definitive answer as to its applicability. The researcher can evaluate the feasibility of the application of technology, if it meets the objectives initially set out to justify (or not) to go ahead with the research [Shull et al. 2001]. Furthermore, the body of knowledge built can shed light on the refinement of the technology and also in generating a new hypotheses on its implementation. In our case only viability study was performed. The other steps of refinement were not applied in this experiment because they are beyond its scope.

3 Integrated Approach

The approach presented here consists of the integration of a development process with a testing process. The testing process is considered as a set of tasks that can be instantiated in parallel and also performed throughout the development process. The development process uses Unified Modeling Language (UML). Its use is attractive since it has interesting information for test. Furthermore, there are no costs associated with training, since both the academic community as some industries, have already consolidated its use [Hartman et al. 2004].

The test process uses UML 2.0 Profile (U2TP). It is a test modeling language that can be used with component technologies, object oriented language and applied in several application domains [OMG 2005]. Moreover, it assists in documentation, understanding and in test artifacts traceability. The test artifacts were specifications an the recommendations of IEEE 829 standard [IEEE 1998].

According ISO/IEC 12207 the approach presented here can be classified as a fundamental process when considering development. In addition, it includes some elements of the organizational process and support. With regarding to organizational process, it includes managerial aspects. In what concerns to to support, it encompasses items of verification an validation, which provides subsidies to increase the quality of the product to be developed.

The integrated approach to development and test is structured in terms of disciplines. An overview of the information flow is shown in Figure 1.

For each Discipline are defined its objectives, the artifacts that are generated, the activities and roles involved in the execution of each activity. The generated artifacts are commonly refined in subsequent Disciplines. A discipline, class, use case or method can be considered units to be distributed at different levels of granularity. The integration of these can occur either within a discipline as well as going through different disciplines.

The integration of development and test processes can bring several benefits, such as: reduction of development cost; achieve a higher level of automation in

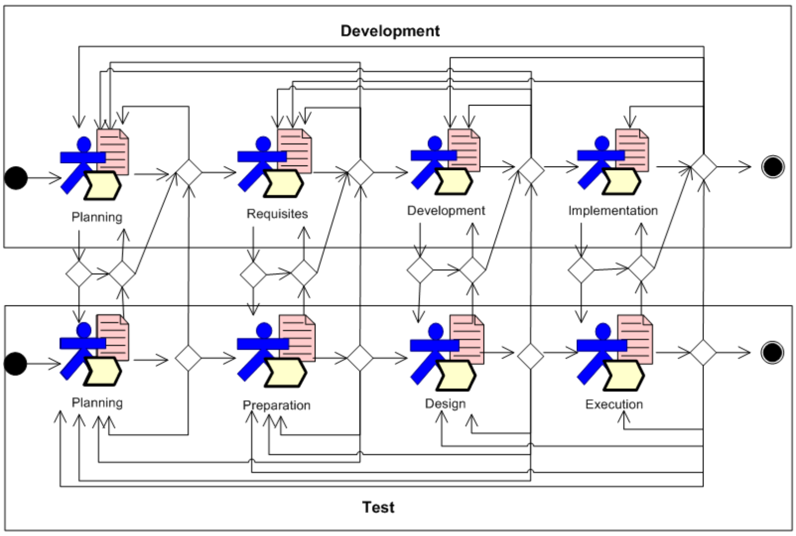

Figure 1: Activities Diagram of Integrated Approach of Software Development and Test.

development and generation of test cases; facility to perform changes in requirements, due the traceability offered; and, less maintenance costs.

The approach is modeled using SPEM. Its notation is based on UML and, for this reason, offer to process modeling the same diagrams that are used to model software. The next sections describe, with more details, the development and test processes [OMG 2005].

3.1 Development Process

The Development Process is composed by four Disciplines: Planning, Requirements, Development and Implementation. They are described in next sections.

3.1.1 Planning

This Discipline is responsible by development process configuration and definition of aspects related to project management. It is composed by the following activities:

– define the development configuration (onshore insourcing, outsourcing, offshoring or internal offshoring) to be adopted in next Disciplines. In this activity is important to explore, strategically, the alliances among organizations to obtain competitive advantage from global production;

- identify teams involved in project, as well as their global e local manager to stimulate the trust and commitment among members. Whenever possible, it is interesting that the teams involved have same native idiom and so minimize communication problems. Should be considered beyond the idiom, affinity and ability questions of members. The central team, that will be responsible by synchronization activities among the several teams, should also established;

- specify the granularity and strategy of distribution (discipline, component, use case, class or method). To establish the distribution strategy the Global and Local Project Managers can consider some points, such as: (i) a modular software architecture (weak coupling and strong cohesion) can be used for effort distribution among teams. This reduces the complexity and lets a parallel and simplified development; (ii) the team proximity of client can be considered as criteria to distribute the discipline, in this case Requirements; (iii) the abilities and competences of involved teams (modeling, test, implementation and other).

  The use of multi-criteria model for task allocation could be suggested. The model considers the task and site features and the relationship between. Them, such as: proximity, feasibility for task, productivity, time cost, quality cost and development quality [Lamersdorf and Munch 2010]. It should be noted, that the distribution strategy adopted has direct impact about the intensity, frequency, and the coordination necessary among team members [Sangwan et al. 2006].

- define an infrastructure to communication and collaboration, both internally (teams involved) and externally (client). For that this communication occurs in adequate way, is necessary that the infrastructure is correctly set. Generally, are necessary several formats, such as: travel, phone, e-mail, videoconferencing, project management tools, instant messenger and wiki [Al-Asmari and Yu 2006].

- define an idiom to formalization of process and interaction among teams;

- define an infrastructure to control version of artifacts and documentation;

- define a common repository to all teams involved. Both Global and Local Project Managers should define the access levels of each participant profile;

- distribute the activities among involved teams and their members. Weissleder and Schlingloff [Weissleder and Schlingloff 2008] present two strategies that can be used to distribute the activities: minimize the necessary collaboration among the teams, because it reduces the negatives impacts of communication problems and collaboration; and, minimizes the teams differences, reducing the time displacement (timezone) that affects the synchronous communication or the cultural differences;

- communicate what is need to be performed for each team, as well as the responsability of each member. This lets that all have consciousness about what is happening, dependences, who is performing determined activity and where it is executed. The consciousness is important, mainly, from the documentation and code perspectives. According to [Carmel and Tija 2005], to increase the awareness some techniques, such as: use of common repositories; systems for project controls; integrated development environment; status meeting; schedule of teams and process description can be used.
- train the stakeholders in relation to approach.

The Global and Local Project Managers are responsable to configuring the development process by performing activities above described. The artifacts generated on software development plan (global and local), that offers a baseline of necessary resources and schedule. In these artifacts are specified project scope, restrictions, organizational structure, roles and responsabilities, hardware and software resources and deadlines.

3.1.2 Requirements

This Discipline describes how to define a system vision and translate it in models. The main goals are: establish and keep contact with stakeholders informing what system should do, facilitate the developers understanding about the requirements by generated models and define the system scope. The activities of this discipline are:

- describe textually the system to be developed;
- represent the business model to understand its structure and dynamic, the current problems and so identify potential improvements. They will considered in the initial project architecture;
- prepare business object model by identifying important items, that constitute the system vocabulary;
- model use cases. An Use Case represents a system functionality showing the interaction among it and the external actors. So, are easy to understand and have a simple structure. Wherefore, they are a good source to identify requirements to test.
- detail the use case specification restrictions using the Object Constraint Language (OCL), that is a language of precise textual expressions used to

describe restriction in object oriented models. It is used as a supplement of the graphic models, to describe restrictions that cannot be diagrammatically represented. OCL is a part of UML standard. In DSD context the use of OCL lets reduce the ambiguity generated in artifacts due to culture factors.

The roles involved are Business Engineer and Specifier. The Textual Description of System, Business Model (Initial Architecture), Business Object Model, Use Case Diagram and Extended Description of Use Case, are the artifacts generated.

3.1.3 Development

The goal of this Discipline is to translate the artifacts generated in Requirements Discipline in a specification that describe how to implement the system.

In this Discipline the following activities are performed:

- modeling the static vision of system by describing a set of objects that share the same attributes, operations, relations and semantic;
- modeling the sequence of messages among objects;
- modeling the communication among objects;
- refine the architecture description presented by business model.

The involved roles are Architect and Designer. The artifacts generated are: Sequence Diagrams, Communication Diagram, Class Diagram and Architecture Description.

3.1.4 Implementation

The goals of this Discipline are to organize the code generation, implementation of class and objects. The integration of results obtained by several teams in runnable system is also performed, if this Discipline, have been distributed.

The activities that compose this discipline are: translation of generated models in code and perform unit tests. Developer is the role involved to perform these activities. The artifact produced is code.

3.2 Test Process

Test Process is composed by four Disciplines: Planning, Preparation, Design and Execution. For each Discipline are presented its elements in the following sections.

3.2.1 Planning

Is the Discipline responsible by test process configuration, defining management aspects. The activities are:

- define the test configuration (onshore insourcing, offshoring, outsourcing or internal offshorring). Eventually, this configuration can be different that defined in development process. The integration test and also system and acceptance test can be outsourced for organizations which are experts in this area. The outsourcing of test activities can be motived by cost reduction, increase of speed with which the tests are performed, acquisition and installation of test environments. The main benefits are: return of investment; greater reliability in the software, a greater range of testing; hiring of a qualified and efficient test team in a short space of time and, major efficiency. In cases of outsourcing, the artifacts defined in the approach should be used to ensure that the necessary information will be delivered as a result of the test. Thus, one could minimize any negative impacts caused by outsourcing and/or also by differences in organizational culture;
- establish the test team and training necessities;
- define environment requirements (tools, people and hardware).

The roles involved are Global Project Manager and Test Designer. The artifact generated is an initial version of Tests Plan.

3.2.2 Preparation

The goal of this Discipline is describe the planning of all activities involved in a software test. The following activities are performed:

- define the test context. It consists in describe the functionalities and features to be tested, identifying the goals and test scope;
- characterize the test items, describing their briefly;
- identify the functionalities and features (for example, number of concurrent access and volume of data in stressfull situations) to be tested. The techniques of risk based test can be used to prioritise, based on ocurrence and impact, and also more coverage test in certain system functionalities. It is due to cost of test activity, the input and output domains are diverse and there are many path possibilities to be tested. So, the efforts as well as the resources can be prioritized and allocated for functionalities that need to be tested more carefully;

– establish approach and criteria. This activity encompasses the determination of test approach, activities definition, techniques and criteria to test, selection of additional criteria of conclusion based on coverage, determination of criteria for pass/fail of each test item and also criteria for suspension and or resumption of test. From use case diagrams, can be applied six test criteria: (i) **criterio all communications, all inclusions and all extensions**, that require the exercise of all relationships of each type; (ii) **criterio all communication-inclusions-extensions** that require the exercise of all relationships of a diagram; (iii) **criterio all extended-combinations and all-extensor-combinations** that require the exercise and not exercise of extension relationships.

  The detailed use cases specification make possible derive test cases from expected flow description, alternative flow and required items;

– establish responsibilities, determining the responsible by test activities;

– make schedule. It consists in to define the test milestones, estimate the time and define the deadlines to perform each activity and use of resources.

The role involved is the Test Designer. The artifact generated is an update of Tests Plan, containing the planning to test execution, coverage, resources and schedule of test activities.

3.2.3 Design

This Discipline has as objective to detail the technical aspects to will be adopted to perform test activity. The activities this Discipline are:

– detail approaches and test criteria;

– specify test cases. The OCL constraints defined in class methods can be used as oracle [Packevicius et al. 2007]. The Sequence Diagram can be converted in a test tree and each path corresponds a test case;

– establish the requirements of test environment based on the necessary infrastructure;

– define the steps of test procedures.

The involved role is Test Designer and the artifacts generated are: (i) Specification of Test Project: contains a refinement of approach presented in Test Plan and identify functionalities and features to be tested, identifying the cases and test procedures; (ii) Specification of Test Cases: define the test cases, including input data, waited results, actions and general conditions to test realization; (iii) Specification of Test Procedures: specifies the steps to execute a set of test cases.

3.2.4 Execution

This Discipline has the objective to perform and register of test activities designed in previous disciplines. The following activities are performed:

- prepare the test procedure that consist in sequence of steps and actions necessary to realization of a set test cases related [Crespo and Jino 2005];
- execute the test procedure;
- prepare the test registry with result of test realization;
- suspend the test;
- close the test;
- register the activities and events;
- describe the test incident, that is any event that occurs during the test realization and requires investigation, such as: software defect or anomaly in operational environment;
- prepare summary description of test items;
- describe the specification deviations, ie, the discrepancies of test items on specifications;
- prepare summary description of test result;
- evaluate test items.

Tester is the role involved and artifacts generated are: (i) item Test Diary: present the chronological register of relevant details of tests realization; (ii) Incidents Report: document any event that occured during the test activities and needs a further analisys; (iii) Test Summary: presents the summary of test activities and provides evaluations based in these results.

4 Viability Study

The main feature of a viability study is that the data are collected according with some experimental project, but it doesn't have the control about all involved variables. The objective is not find a definitive answer, but instead construct a knowledge body that deals the plausibility of study continuity, generate new hypothesis about the approach and its utility [Shull et al. 2001].

This study was conduced according process described in [Wohlin et al. 2000], that presents the main activities that should be executed. They are: Study Definition, Planning, Execution, Analysis and Packing. The next sections describe them in more details.

4.1 Study Definition

In **Study Definition** activity, the global goal, the measuring goal and the study goals (using the Goal-Question-Metric approach) and also the questions to be answered were defined. So, the global goal is ”characterize the viability of integrated approach of software development and test to distributed teams”. So, investigate and characterize the approach viability in relation to DSD context is the measuring goal.

The study goal is analyze the integrated approach of software development and test to distributed teams. With it the purpose is to characterize the viability of the activities, artifacts and roles defined, from researcher viewpoint, in the context of people involved with DSD. The questions defined were:

- **(Q1)** The activities present in approach are enough to support the DSD? **Metric:** The list of activities offered by approach.
- **(Q2)** The artifacts present in approach are enough to support the DSD, supporting communication among development and test teams? **Metric:** The list of artifacts specified by approach.

4.2 Planning

The **Planning** prepares the study taking into account: hypothesis formulation, context selection, variables selection, participant selection and project under study. The hypotheses formulated were: **(H0)** The integrated approach doesn't contemplate all peculiarities of distributed development; and, **(H1)** The integrated approach contemplates all peculiarities of distributed development.

The study was conduced using questionnaires those were sent to participants with knowledge covering the study domain. The participants were selected based on their DSD knowledge and availability. The study was performed in academic community, because according to [Shull et al. 2001] it is appropriate for viability studies, once it makes possible that new concepts are tested before their use in industry.

In the instrumentation two questionnaires were used. The first one was related with the participant profile (independent variables), for example: experience with software development, knowledge in distributed software development, project management and software testing. The second questionnaire aimed at to evaluate the approach (dependent variables), for example: degree with that the set of disciplines, activities and artifacts meet the DSD.

4.3 Execution

In **Execution**, the specification and questionnaires were sent to 16 people. A specification presenting the goal study, the features of approach and Package Diagrams of each discipline, were also elaborated.

4.3.1 Study Results

The Table 1 presents the results for the independent variables (A - C) and dependent (D-L), where: **(A)**: Knowledge in Distributed Software Development; **(B)**: Experience in Project Management; **(C)**: Knowledge of software testing; **(D)**: Set of disciplines is satisfactory for the DSD; **(E)**: Set of activities addresses the needs of the DSD; **(F)**: Set of artifacts meets the needs of DSD; **(G)**: The use of UML notation can reduce the communications problems, since it provides information relevant to both teams (development and testing); **(H)**: Identification of staff and responsibilities of its members through the Global Development Plans, Local Development Plan and Test Plan; **(I)**: Common nomenclature to all involved in the project enables to understand the dependence of the activities and artifacts; **(J)**: Use of OCL (*Object Constraint Language*) to specify restrictions; **(K)**: Impact of standardization of artifacts and activities on the quality of the products developed; **(L)**: Test activities through all disciplines.

In Table 1 the scale of measurement adopted is: 0 - None, 1 - Low 2 - Intermediate 3 - Satisfactory.

Table 1: Results of Viability Study

N	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**
1	2	0	1	3	3	3	2	2	3	2	3	3
2	2	2	2	3	3	3	2	2	3	3	3	3
3	2	1	2	3	3	3	2	3	2	2	2	3
4	2	1	1	2	2	2	3	2	3	3	2	3
5	2	0	1	3	3	2	2	3	3	2	2	3
6	3	1	1	3	2	3	2	3	2	2	3	3
7	2	2	2	2	2	3	3	2	3	3	3	3
8	2	1	1	2	2	3	3	2	2	3	2	2
9	3	2	2	3	3	3	3	3	2	2	1	3
10	1	1	1	2	3	3	3	3	2	2	3	3
11	3	1	1	3	3	3	3	2	3	2	2	3
12	2	0	3	3	3	3	3	3	3	3	3	2

4.4 Analysis and Interpretation

Analysis and Interpretation are divided into four stages: Validation of Data, Descriptive Statistics and Analysis, Application of Statistical Testing and Verification of Hypotheses. These steps are described as follow.

4.4.1 Data Validation

The study used 12 participants, all completed the questionnaire and profile Interviewed Feasibility Study. We can not find outliers in the responses to the questionnaires.

4.4.2 Descriptive Statistics and Analysis

Tabulation and presentation of data are fundamental to the proper trial because they allow statistical focus on relevant features to be used in solving problems. Thus, average, median and mode make possible organize the events highlighting the sample since the values are in ordinal scale.

The Table 2 presents the descriptive statistics (median and mode) of collected data. After tabulating the data were generated graphics using the gnuplot 4.0 graphical tool, which are shown in Figures 2, 3, 4, 5, 6, 7, 8, 9 and 10. These graphics illustrate the relationship between the knowledge of participants in Distributed Software Development, Project Management Experience / Knowledge in software testing. The third coordinate, represents the characteristic under analysis of the proposed approach.

Table 2: Descriptive Statistics

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**
Median	2	1	1	3	3	3	3	2,5	3	2	2,5	3
Mode	2	1	1	3	3	3	3	2	3	2	3	3

On the Figures 2, 3 and 4 can be observed that the participants with intermediate/advanced, knowledge level in DSD and with basic to intermediate knowledge level in Project Management assessed that the Disciplines, Artifacts and Activities in the approach proposed include the necessary specifications for DSD. Also these specifications meet among intermediate to satisfactory level. The valley represents the participants who do not have experience in project management.

The Figure 5 shows that the participants with intermediate/advanced knowledge level in DSD, regardless of experience in Project Management assessed that the use of information provide by UML notation can alleviate the communication problems between development and testing teams in intermediate/satisfactory grade.

On Figure 6, graph shows two peaks, for identification of teams and their members as a mechanism to raise awareness so intermediate/satisfactory.

On Figure 7, are showed a peak and a valley, which are given by the variation of participants experience in project management. However, it is observed that the approach meets with intermediate/satisfactory degree the needs of the DSD for providing a common nomenclature for all involved in the project and so facilitating the understanding of dependencies between activities and artifacts.

The graph in Figure 8 presents two peaks, which are formed due to variation of experience in project management. But, in general the use of OCL was

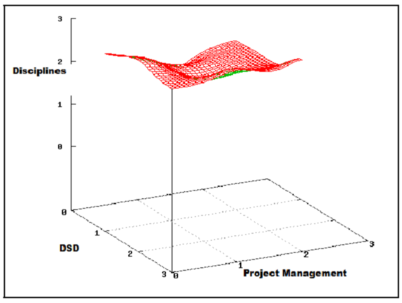

Figure 2: Graphic of relation Knowledge in DSD x Experience in Project Management x Disciplines.

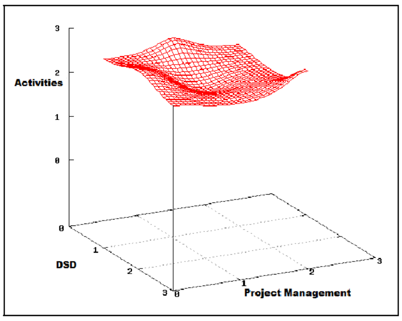

Figure 3: Graphic of relation Knowledge in DSD x Experience in Project Management x Activities.

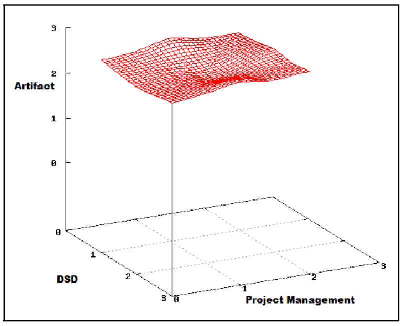

Figure 4: Graphic of relation Knowledge in DSD x Experience in Project Management x Artifacts.

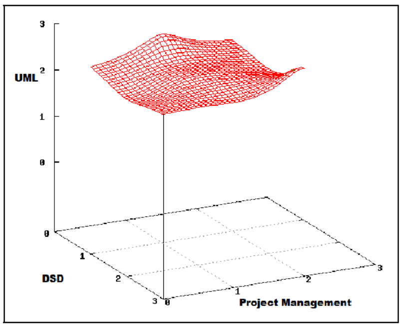

Figure 5: Graphic of relation Knowledge in DSD x Experience in Project Management x UML.

evaluated as an intermediate/adequate for the problems of imprecision of the artifacts.

On Figure 9, the graph shows a valley and a depression. The valley represents participants with basic knowledge in DSD and with intermediate/basic degree experience in project management. They assessed as being intermediate the degree of the impact that standardization of artifacts and activities can cause in quality of their products The depression shows that participants with intermediate knowledge and experience in DSD rated that standardization is not decisive in the quality of products.

The graphic on Figure 10 presents a depression, which represents participants with advanced knowledge in software testing and intermediate knowledge

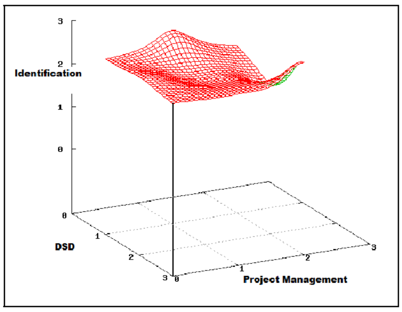

Figure 6: Graphic of relation Knowledge in DSD x Experience in Project Management x Identification of Teams.

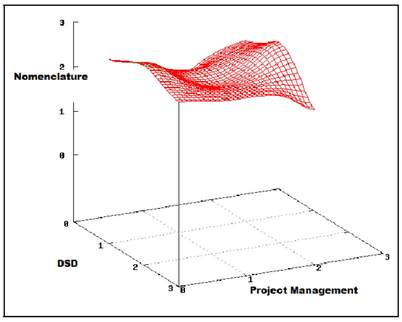

Figure 7: Graphic of relation Knowledge in DSD x Experience in Project Management x Nomenclature.

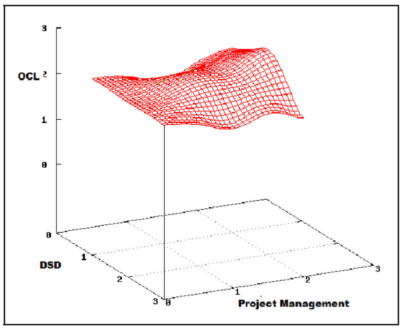

Figure 8: Graphic of relation Knowledge in DSD x Experience in Project Management x OCL.

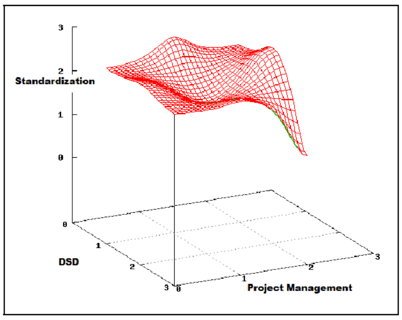

Figure 9: Graphic of relation Knowledge in DSD x Experience in Project Management x Standardization.

in DSD. They evaluated that in the DSD test activities through development disciplines may reduce integration problems as being of intermediary degree. The participants with basic/intermediate knowledge in test, told us that the problems with integration could be reduced in satisfactory degree.

4.4.3 Application of Statistical Testing

Participants were classified into only one level of knowledge from the input variables. The dependent variables were related to the independent variable that measures the knowledge of the participant through the chi-square statistical test.

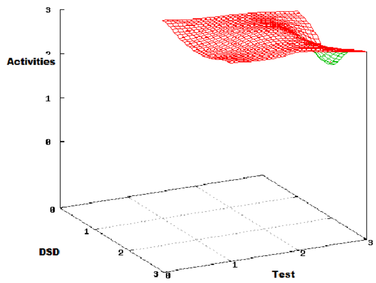

Figure 10: Graphic of relation Knowledge in DSD x Knowledge Test x Test Activities along the Disciplines.

As the chi-square test considers the comparison of distributions, we compared the distribution of participants' responses to the optimal distribution.

The probability density function of chi-square is represented by x_v^2, where v is the number of degrees of freedom of x^2. If v observations of a variable are independent, then the number of degrees of freedom is equal v. However, a degree of freedom is lost for every restriction on the v comments.

The distribution function is given by $P(X^2 >= X_v^2) = \alpha$, where α is the probability of sums of squares equal to or higher than the corresponding tabulated value. Significance α is, generally, set around the value 0,05. The higher α, the greater the risk of rejecting good chance; conversely, the risk of accepting false hypotheses increases, the extent that the value of α decreases. If the calculated value of the random variable x^2 is greater than the tabulated value x_v^2, rejects the hypothesis that the variables are random, however, if it is less than or equal, the hypothesis is accepted.

The level expected in the approach addresses the needs of the DSD according to the knowledge of participants.

The α adopted is 0,05 and v is 9. Thereby, the X_v^2 calculated for "disciplines set meets the needs of the DSD" was 6,125. The X_v^2 obtained for "activities that meets the needs of DSD" was 6,125. The X_v^2 obtained for "artifact set meets the needs of the DSD" was 0,04847. For "UML notation can ease communication problems" and "identification of teams can increase awareness" the X_v^2 calculated was 4,5. The X_v^2 obtained for "approach serves the needs of the DSD" was 3,125. The X_v^2 obtained for "OCL that can alleviate the ambiguity" was 4,5. The X_v^2 calculated for "standardization of activities and artifacts can impact on quality" was 6,125. For "testing activities throughout the process can reduce the problems of integration" the X_v^2 was calculated 0,04847.

4.4.4 Hypotheses Verification

For the table of the chi-square $X_v^2 = 7{,}815$. By applying the chi-square tables on all related dependent variables the response were $X^2 >= X_v^2$. Thus, H1 was rejected in favor of H0. With this, we can say that "The integrated approach of software development and testing does not include all the peculiarities of distributed development".

Since the study sample was small it has the following problems related to validity: (1) The power of statistical Chi-square is low, which affects the validity of conclusion, (2) As the level of knowledge of participants is near there is no diversity of profiles, which compromises the external validity (generalization of results).

Moreover, we observed that the study may have been influenced by the classification of knowledge in only one level because most participants possessed intermediate or advanced knowledge in DSD, but had no or little experience in project management and basic knowledge in software testing. Thus, it is possible to stratify the analysis using each type of knowledge of the participant and apply Pearson correlation to visualize the linear relation among variables.

5 Final Considerations

The growing search for more competitiveness has taken the organizations to adopt the DSD. Trying to reduce costs with software development, organizations have crossed boundary, forming a global market. The global production can be motivated, among other facts also, by quality questions, production management, knowledge of adopted technologies, reduction of needs immediate hiring or knowledge transfer to a subsidiary

This change of paradigm has caused impact in marketing, in distribution and conception way, as well as the production, design, test and software delivery to clients. According [Damian and Lanubile 2004] to minimize this effects and improve productivity are necessary new technologies, processes and adequate methods with the global development approach.

In this scenario this work presented an integrated approach to offer support to development with distribute teams. This approach was elaborated based on gaps identified in the systematic review and in development processes and test analysed. The approach covers a set of activities that should occur throughout project life cycle and essentials features of each one. Also, considers activities since the business model (global companies, strategic alliances and others) be adopted in DSD until the implementation.

The approach was modeled using SPEM notation aiming at to facilitate communication, the process understanding, the reuse, support its evolution, facilitate to management and thinking in the continuous improvement of process.

The main contributions of this work are:

- the approach is structured in terms of roles, artifacts and activities well defined. This assists directly in synchronization, by offering to all stakeholders a common nomenclature. This allows a better understanding of the terms of the business domain and project milestones, despite of cultural differences and organizational structure, which can occur in the scenario DSD. Moreover, it has a well defined process allowing it to be analysed and as a result of this analysis, to envolve and therefore subject to continuous improvement;
- offers support to development process, planning and software design. The activities and guidelines for strategic level (manager aspects) as well as operational (verification and validation), that offer subsidies to quality of software to be developed are presented. Moreover, offers guidelines to help the project manager (Local and Global) in activities distribution;
- presents activities and guidelines to idiom definition to be adopted in process formalization and communication among teams;
- formalizes the software test importance in a project, treating like as an approach that should be instantiated in parallel to development and with activities since the software conception;
- offers mechanisms to standardization and imprecision reduction of artifacts. With artifacts standardization is possible to reduce the communication problems among teams. Templates specification for each artifact generated is a primordial factor to facilitate the effective communication among teams members;
- uses the UML notation, that is easy understanding in academic way as well as at industrial and has semantic to transmit information to developers. Also, uses formal test specification through U2TP profile to minimize the ambiguity problems and reduce the communication needs;
- assists the awareness and knowledge of activities performed, such as those being conducted at each location as well as of each member responsabilities through two categories of awareness: (i) activity awareness: find answer for questions, like as: "Who is working in which activity?", "Who is the responsible to determined task and which the result?", "Where the activity is being executed?"; (ii) process awareness: relates to questions as: "How task of that person fits in mine?", "What should do now?"

To continue the activities of this work, we suggest as future works: (i) replicate the viability study to enlarge the knowledge acquisition about process application, credibility and the confiability index; (ii) refine the approach through of conduction of followings studies: observation study, case studies (life cycle and industry); (iii) define mechanisms to evaluate the social networks formed during the project; (iv) define metrics to evaluate the productivity, communication and quality of generated artifacts; (v) integrate this approach to an environment that offer support to DSD, like DiSEN (Distributed Software Engineering Environment) [Huzita et al. 2007].

References

[Al-Asmari and Yu 2006] Al-Asmari, K.; Yu, L. Experiences in Distributed Software Development with Wiki. In: Software Engineering Research and Practice, (2006).

[Biolchini et al. 2005] Biolchini, J.; Mian, P.; Natali, A.; Travassos, G. Systematic review in software engineering: Relevance and utility, COPPE/UFRJ, Rio de Janeiro, RJ, (2005).

[Boden et al 2011] Boden, A., Mller, C., Nett, B.Conducting a Business Ethnography in Global Software Development projects of small German enterprises. Information and Software Technology, 53(9), 1012-1021, (2011)

[Carmel and Tija 2005] Carmel, E.; Tija, P. Offshoring Information Technology: Sourcing and Outsourcing to a Global Workforce. Cambridge University Press, New York, (2005).

[Crespo et al. 2004] Crespo, A. N.; Silva, O. J.; Borges, C. A.; Salviano, C. F.; Argollo, M. T.; Jino, M. A Methodology for Software Testing in the Context of Process Improvement. In: Brazilian Symposium on Software Quality, Braslia-DF, (2004) (Portuguese).

[Crespo and Jino 2005] Crespo, A. N.; Jino, M. Software Testing Process. *Centro de Tecnologia da Informao Renato Archer* (CenPRA), (2005) (Portuguese).

[Damian and Lanubile 2004] Damian, D.; Lanubile, F. The 3rd International Workshop on Global Software Development. In: ICSE '04: Proceedings of the 26th International Conference on Software Engineering, IEEE Computer Society, Washington, DC, USA, pp. 756–757, (2004).

[Fuggetta 2000] Fuggetta A. Software Process: A Roadmap. In: International Conference on Software Engineering, Proceedings of the Conference on The Future of Software Engineering, (2000).

[Hartman et al. 2004] Hartmannm, J.; Vieira, M.; Ruder, A. UML-based Test Generation and Execution. In: Proceedings of the 21st Workshop on Software Test, Analyses and Verification (GI-FG TAV), Berlin, (2004).

[Herbsleb et al. 2000] Herbsleb, J. D.; Mockus, A.; Finholt, T. A.; Grinter, R. E. Distance, dependencies, and delay in a global collaboration. In '2000 ACM conference on Computer supported cooperative work (CSCW '00)', ACM, USA, p. pp.209, (2000).

[Humphrey and Kellner 1989] Humphrey, W. S.; Kellner, M. I. Software Process Modeling: principles of entity process models. In: ICSE '89: Proceedings of the 11th international conference on Software engineering, New York, NY, USA, pp. 331–342, (1989).

[Huzita et al. 2007] Huzita, E. H. M.; Tait, T. F. C.; Colanzi, T. E.; Quinaia, M. A Distributed Software Development Environment - DiSEN. In: I Workshop of Distributed Software Development, pp. 31–38, Joao Pessoa, PB, (2007) (Portuguese).

[IEEE 1998] IEEE. IEEE Standard for Software Test Documentation. ANSI/IEEE Std 829-1998, New York, (1998).

[Jimenez et al. 2009] Jimnez, M.; Piattini, M.; Vizcano, A. Challenges and Improvements in Distributed Software Development: A Systematic Review. Advances in Software Engineering, vol. (2009).

[Juristo et al. 2001] Juristo, N., Moreno, A.M. Basics of Software Engineering Experimentation, Hardcover, ISBN: 0-7923- 7990-X, 2001.

[Lamersdorf and Munch 2010] Lamersdorf, A.; Mnch, J. A multi-criteria distribution model for global software development projects. Journal of the Brazilian Computer Society, Volume 16, Number 2, Springer, pp.1–19, (2010).

[Leal et al. 2010] Leal, G. C. L. and Silva, C. A.; Huzita, H. M. M. Towards a Distributed Software Process. In: IADIS International Conference IADIS Information Systems, Porto, (2010)

[Mafra et al. 2006] Mafra, S. N.; Barcelos, R. F.; Travassos, G. H. Applying an evidence-based methodology in the Definition of New Technology Software. In: Brazilian Symposium on Software Engineering (SBES), pp. 239–254, Florianoplis, SC, (2006) (Portuguese).

[McGregor and Sykes 2001] McGregor, J. D. and Sykes, D. A. A practical guide to testing object-oriented software, Addison-Wesley, Boston, MA, (2001).

[OMG 2005] Object Management Group. UML 2.0 Testing Profile. [S.l.], (2005). (Document-formal/05-07-07).

[Packevicius et al. 2007] Packevicius, S.; Usaniov, A.; Bareisa, E. Software testing using imprecise OCL constraints as oracles. In: Proceedings of the 2007 international Conference on Computer Systems and Technologies (CompSystech'07), ACM, New York, NY, USA, (2007).

[Patil et al. 2011] Patil, S., Kobsa, A., John, A., Seligmann, D. Methodological reflections on a field study of a globally distributed software project. Information and Software Technology, 53(9), 969-980, (2011).

[Sangwan et al. 2006] Sangwan, R. and Bass, M. and Mullick, N. and Paulish, D. J.; Kazmeier, J. Global Software Development Handbook (Auerbach Series on Applied Software Engineering Series), Auerbach Publication, Boston, MA, USA, (2006).

[Shull et al. 2001] Shull, F.; Carver, J.; Travassos, G. H. An Empirical Methodology for Introducing Software Processes. In: Proceedings of the 8 th European Software Engineering Conference, pp. 10–14, (2001).

[Weissleder and Schlingloff 2008] Weissleder, S. and Schlingloff, B. Quality of Automatically Generated Test Cases based on OCL Expressions. In: ICST '08: Proceedings of the 2008 International Conference on Software Testing, Verification, and Validation, IEEE Computer Society, pp. 517–520, Washington, DC, USA, (2008).

[Wohlin et al. 2000] Wohlin, C.; Runeson, P.;Höst, M.; Ohlsson, M. C.; Regnell, B.; Wesslén, A.. Experimentation in software engineering: an introduction. Kluwer Academic Publishers, Norwell, MA, USA, (2000).