A Requirements Engineering Process Adapted to Global Software Development

Juan P. Mighetti
Universidad Nacional de La Matanza, DIIT,
San Justo, Buenos Aires, Argentina, B1754JEC
jmighetti@unlam.edu.ar

and

Graciela D. S. Hadad
Universidad Nacional del Oeste, Escuela de Informática,
Merlo, Buenos Aires, Argentina, 1722
ghadad@uno.edu.ar

Abstract

Global software development is spreading increasingly within companies. Although it provides some competitive advantages, such as speeding up the software delivery, reducing cost, and taking advantage of more economic resources, this working model is a very complex one. Threats, such as geographic distance and cultural differences, may impact negatively on activities and artifacts of the software process. Untreated threats usually affect the quality of the requirements, propagating defects to subsequent development phases. Global software development threats were studied in a real major project, where no special treatment was put into practice to mitigate them. Analyzing the serious consequences on that project, a proposal was developed using a Lexicon model and Scenarios in order to mitigate threats to requirements in this distributed working mode. The proposal was applied in a new real project of similar characteristics, and the comparison of results from both projects gives promising perspectives in terms of requirements quality and process time improvements.

Keywords: Global Software Development, Communication, Requirements Engineering, Natural Language, Scenarios.

1 Introduction

The requirement specifications are the entrance door for the subsequent phases in the software development process. In this sense, the quality related to specifications is crucial in order to obtain a proper software product and to fulfill with the agreed deadlines and costs. The requirements production is not a trivial or monolithic activity [1] and, thus, it should be encouraged following a defined process, which takes into account the circumstances of the project and the application context [2]. A particular case is the global software development (GSD), which is often affected by various threats, such as geographical and temporal distance, different cultures, different languages and communication difficulties, among other factors [3], [4], [5]. These latent threats usually impact negatively on the requirements [4]. To understand the dimension of the requirements within the software process, it should be mentioned that the problems associated with the requirements engineering process are the main cause of failure of software projects [6].

It is important to know the level of exposure to threats that the GSD project will deal with; this level will also depend on social aspects, domain knowledge of the involved team, trust between the different stakeholders, and common or individual interests competing. Belatedly recognize these threats and/or underestimate them could bring harmful consequences to the requirements, in first place, and to the whole project in general, in quality as well as in

time and costs. Anticipate threats and take the appropriate mitigation measures could help to achieve the expected results [5].

Therefore, a requirements process based on models written in natural language has been proposed, together with a knowledge management repository, with the purpose of tackle communications problems, distance and idiomatic differences, and lack of domain knowledge. This proposal was created taking into account the difficulties detected and quantified in a GSD project, where actions against threats, typical of the GSD model, were ignored. The proposed process was applied in a real project within the same domain, and by comparing the results achieved on the quality of the requirements and on the timing of the involved activities, it was possible to identify the mitigations obtained over the threats affecting requirements.

The following section describes a requirements process based on natural language models and the potential threats that can affect a GSD project. In section 3, the threats identified in a real project of GSD after a post-mortem analysis are exposed. Section 4 presents the mitigation proposal to requirements threats and the application in other project with similar characteristics, analyzing comparatively the mitigation achieved on these threats. Section 5 discusses similar works on GSD. Finally, conclusions and future works are presented.

2 Requirements and Global Software Development

It is well known that requirements are an essential component for the generation of software, and a proper requirements stage is vital in any software process [7]. If a defect occurs during the requirements production, it will undoubtedly impact on subsequent phases of the software process, damaging the whole product [7]. Additionally, the GSD has distinctive features that complicate the software development process and affect much more the requirements definition activities [8].

Software globalization encourages more and more to seek cost-effective alternatives outside the boundaries of countries [9]. These decisions are motivated by various causes with different benefits [10], [11], [12], such as: possible costs reduction, increased flexibility and professional skills of the teams, rapid adaptability of the company to market changes to achieve short-term goals, competitiveness through cost reduction, hiring models flexibility, risk mitigation related to labor and taxes, and possibility of expansion into new markets.

This set of advantages comes with different threats that must be monitored to avoid negative effects, since these threats mainly affect the requirements [13] and conspire against time, cost and quality of projects. As mentioned by Davis et al. in [14], problems in requirements production are considered the main factor of failure of 90% of large software projects.

The unawareness of these threats and the lack of mitigation actions may affect the requirements and lead to loss of competitive advantages, staff demotivation, and poor software quality [5]. In that sense, requirements engineering, as a discipline, seeks to systematize the requirements process, providing methods, techniques and tools that facilitate the elicitation, modeling, verification, validation and management of high quality requirements, transforming business requirements into software requirements [15]. The requirements engineering process must be adapted depending not only on the type of software to be built, but also the specific development environment. A requirements strategy that focuses on increasing the commitment of stakeholders will obtain better requirement specifications [16], and this commitment is mainly achieved through a proper communication between all stakeholders.

2.1 Requirements Process based on Natural Language Models

Requirements engineering has a collective social ingredient due to the involvement of a variety of stakeholders in a project, who often have different skills, knowledge and vocabularies. These differences cause the understanding of the problem to be a complex activity. In order to have a successful engineering requirements process, it is vital to have good communication among stakeholders and to understand deeply the context of the system to be built [17], [18]. That is the reason of following a requirements process based on natural languages models. The Language Extended Lexicon (LEL) model and the Scenarios model have this communication feature, which provides advantages not only to the requirements process, but also to the entire software life cycle, being an easy understanding media between parties [17].

The LEL is a glossary that defines terms used in the application context [18]. Each term, called *symbol*, is identified with one or more names (synonyms) and defined by a notion (denotation) and a behavioral response (connotation in this context). Both, the notion and the behavioral response, are described by means of one or more sentences driven by two principles: i) circularity principle (to maximize the use of LEL symbols while describing other LEL symbols), and ii) minimum vocabulary (to minimize the use of terms not belonging to the LEL). Fig. 1

presents two symbols of the LEL created in a GSD project of an insurance company, with the intention to mitigate threats to requirements. Underlined terms in Fig. 1 represent hyperlinks to their corresponding definition symbols within the lexicon.

Symbol	**Policy / Contract**
Notion	• A written document containing the terms of the agreement between the insurer and the insured. • Instrument evidencing the agreement between the insured and the insurer.
Behavioral Response	• It defines the rules in general, particular or special ways that regulate the contractual relationship between the insurer and the insured. • It has a unique policy number. • It has an insured and policy holder and can be these two different entities. • The insured receives a printout after contracting the insurance. • It may be modified, terminated or extended. • The Premium can not to be collected by the Insurance Company directly. • All premiums are collected through agents. • The payment of the policy premium could be done by check, deposit, or bank transfer.
Symbol	**Insured**
Notion	• Person, owner of the interest on which insurance risk is taken. • The person or organization that is covered by the insurance policy. • He may or may not be the beneficiary of the policy.
Behavioral Response	• He hires the policy with the insurer. • He can negotiate, terminate or extend the policy.

Figure 1: Example of two LEL Symbols

Scenarios are behavioral descriptions in a given context and at a particular timeframe [17]; the behavior is defined by a set of episodes, performed by actors to achieve a specific goal, and by exceptions that obstruct such achievement. The context of the scenario is described by means of a geographical location, a temporal location and preconditions. Episodes may be simple actions, conditional actions or optional ones, performed in sequential or non-sequential order. A episode may also reference to other scenario, called sub-scenario. Exceptions are described in terms of the cause of the disruption and the treatment or resolution of that disruption. The exception treatment may be described by a simple action or a scenario, depending on the complexity of the treatment. In addition, the actors participating in episodes and the required resources used in episodes are also enumerated in the scenario.

Two different sets of scenarios are created: i) Current Scenarios, which describe the situations observed in the application context, and ii) Future Scenarios, which describe expected situations that will take place in the context of use of the software to be built. Fig. 2 shows a Future Scenario created for the insurance GSD project. In this Figure, underlined terms are hyperlinks to the definitions of the LEL symbols, and treatments of the exceptions invoke other scenario.

In general, every application context has a typical terminology, that is, words or phrases that have a particular meaning in that context. Misunderstanding this vocabulary may lead to defects in requirement specifications, which would result in re-work, repeating verifications and validations. Thus, the LEL improves communication among the stakeholders, standardizing the user's vocabulary as the common terminology to be used, helping to build a good relationship with users, facilitating the validation of models written in natural language, and minimizing ambiguities in the created models, among other contributions [18].

Scenarios enable to understand the behavior in the application context, to unify criteria about the best way to interact with the system, to increase the user's commitment with the development process, and to facilitate the training of new team members about the problem to be solved [17].

A requirements engineering process based on natural language models consists of four phases [15]:

i) Understand the context vocabulary: create, verify and validate the LEL;

ii) Understand the current context: create, verify and validate Current Scenarios;

iii) Define the software context: create, verify and validate Future Scenarios;

iv) Extract the software requirements from the Future Scenarios: create, verify and validate requirement specifications.

It should be noticed that the *creation* of a model involves several activities: elicitation, modeling and managing

requirements. Additionally, it is also important to remark that all the mentioned models use the terminology defined in the LEL, containing hyperlinks to the definitions of the corresponding LEL symbols, in order to reduce the inherent ambiguity of natural language descriptions. Examples of these hyperlinks are depicted in Fig. 2, where some LEL symbols are used in the scenario description.

Title	**Pre Processing Incoming Payments**	
Goal	The Interface should process the payments associated to policies collected by other entities in name of the Insurance Company and registering the accounting entries into Intermediate Data Base.	
Context	**Temporal Location**	• Every day • Brazilian Time should be used for Interface schedule execution time
	Geographical Location	• Brazil
	Preconditions	• Existence of the Intermediate Data Base. • Access (user & Password) to the location where the file is stored in order to allow interface to log and read the file. • Access (user & Password) to the Intermediate Insurance Company Data Base to allow interface to record the registers. • Interface execution scheduled in Execution tool.
Actors	Bank entity (File), Insurance Company (Intermediate data base, processing interface)	
Resources	File with the collection, Intermediate Data Base, processing interface	
Episodes	1. The bank lets the file available for interface reading.	
	2. The interface invokes with the product parameter.	
	3. The interface invokes bank entity logging service.	
	4. The interface reads full file from bank entity.	
	5. The interface filters the rows by motor or other products specified.	
	6. The interface inserts data into the Intermediate Insurance Company Data Base taking into consideration the unique policy number.	
	7. The interface confirms the processing of the data and deliver status log.	
	8. The interface logoffs from the service.	
Exceptions	1. Error trying to access File (log error and INFORM EXECUTION PROBLEM).	
	2. Security error trying to access Intermediate Insurance Company Data Base (log error and INFORM SECURITY PROBLEM).	
	3. Error invoking centralized logging service (log error and INFORM EXECUTION PROBLEM).	
	4. Error invoking centralized error handling service (log error and INFORM EXECUTION PROBLEM).	

Figure 2: Example of a Future Scenario

Fig. 3 shows the structure of a requirement specification. This document presents an introduction, a summary of the involved business process and rules, a functional description of the solution for that business process, and a section involving non-functional requirements, which should be considered for a proper solution. This schema was the one used by the insurance company in every software development project.

1. Introduction
2. Business Process and Business Rules
3. Solution Requirements
3.1 Business Function Requirements
3.1.1 Function 1: <description>.
3.1.2 Function 2: <description>.
……
4. Other Business Process Requirements
4.1 Organizational Requirements
4.2 Location Requirements
4.3 Data Requirements
4.4 Application Requirements
4.5 Technology Requirements

Figure 3: The structure of a Requirement Specification

2.2 Threats in Global Software Development

Although it is considered that distributing work teams is a strategic decision that pursues advantages, such as cost reduction and higher competitiveness, this may lead to big investments and hard challenges [12]. Organizations that tackle this kind of work should have stipulated processes, adequate tools, appropriate means of communication, clear policies, among other aspects as well, in order to face the diverse threats that surround the GSD [19].

The main threats to GSD [3], [4], [5], [12], [19] are: i) Inadequate communication, ii) Language and cultural barriers, iii) Geographical distance and time differences, and iv) Problems with knowledge management. Additionally, other threats derive from the combination of those mentioned above [13] being some of them not specific of GSD, however, they must be addressed (Fig. 4): v) Lack of confidence and engagement; vi) Lack of knowledge about the problem domain; vii) Ambiguity, contradictions and lack of clarity in specifications; vii) Technical or tools problems; ix) Individual goals; and x) Rotation of team members. Underestimating or not being aware of these threats may impact heavily on the quality of the requirements [13], and when these threats get identified in distant phases they will greatly increase the costs of resolution [20].

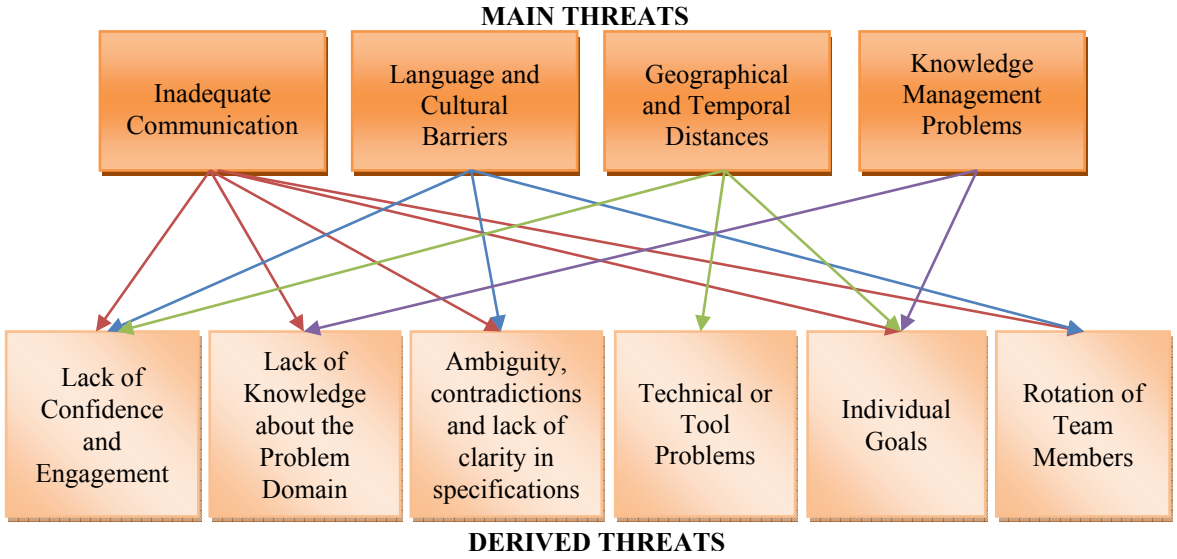

Figure 4: Main and Derived Threats in Global Software Development

Inadequate communication. Communication is an indispensable element that all actors have to daily deal with, during the whole software development process [17]. Particularly, the requirements stage will only be successful if there is an effective communication between the users and the analyst team, avoiding discrepancies in the understanding of the user necessities [7]. Some instruments that improve communication could be those informal and face-to-face ways of communications, which are highly used in projects that are located in only one place.

Usually the communication between members of a team under the distributed model is through electronic means and in an asynchronous way. The abuse of this form of communication is also harmful for the project, because it generates misunderstandings, re-works and ambiguities in the documents that are produced [5], [4]. These problems may decrease productivity by up to 50% in GSD projects, and may increase re-works by 2 to 5 times compared to a project located at the same place [12].

Language and cultural barriers. Culture may differ according to the following areas [4]: organizational, geographical, national culture, religion and power distance. Language is a critical factor that directly impacts on the elicitation and validation activities [4]. This situation can occur both between users and analysts, or between analysts and developers or testers, or between other groups as well. The meaning of some context specific terms and their relation with other technical terminology may differ in some languages or cultures, affecting the requirements quality.

Geographical distance and time differences. Geographical distance among stakeholders becomes a challenge for the face-to-face or informal communication. Temporal distance also attempts against synchronous communication, which is a useful tool to quickly overcome inconsistencies and ambiguities before they become bigger problems [4]. The lack of a fluid communication not only affects the bonds among stakeholders, but also affects the right understanding of the application context and also of the user necessities [4].

Knowledge management problems. The purpose of knowledge management is to absorb the whole knowledge of the organization in order to be used later on, including also the compiling of information of past and actual projects [11]. In the requirements stage, there is usually a large amount and variety of information coming from diverse sources of information, and these sources have to be available to all involved people. This factor should be considered especially in a distributed model, where involved participants are geographically disperse and must have formal channels to access to the information to be shared; this helps to avoid tacit knowledge and the informal transmission of information [8].

Lack of confidence and engagement. Trust is difficult to establish in distributed teams, where relationships and socialization are restricted due to geographical distance and time differences [5], [4], [12]. While profile creation and collaborative tools can help to mitigate this threat [12], it is necessary a proper understanding between those involved in order to work as a team pursuing the same objective.

Lack of knowledge about the problem domain. Having knowledge about the application context is of vital importance in a project, because the lack of knowledge may lead to a round cycle of questions and answers, and continuous and frustrating validations, causing significant delays. The interpretation of information by someone who has no knowledge about the context may be a difficult task [21]. Moreover, those who better know the application domain could be overwhelmed by questions [5].

Technical or tools problems. Connectivity plays a major role when dealing with GSD. The support to teamwork through multi-language tools, groupware tools and shared repositories among other things, are basic elements to consider when approaching this kind of project [12].

Ambiguity, contradictions and lack of clarity in the specifications. When specifying a requirement, ambiguity must be avoided. As the IEEE 29148 [22] standard indicates, there must be only one interpretation for each requirement, and this has to be simple and easy to understand. Having cultural and language differences, as well as a limited knowledge of the domain and abuse of asynchronous communication, may increase the probability of producing ambiguous, inaccurate or contradictory requirements. Requirements with this kind of defects will impact on subsequent phases of the software project, damaging the quality of produced artifacts and the final software product [7].

Individual goals. The lack of common goals or objectives may result in the demotivation of the teams or in low productivity, and also may promote the culture of “them against us” [5], affecting significantly the outcome of the project. The objectives may not be the same between the different teams, because they report to different management structures or different companies. This explains why it is so difficult for teams to understand and commit to the same objective [12].

Rotation of the team members. The loss of experienced resources during the development of a project has a direct impact on productivity, knowledge transfer and team motivation, overloading the remaining team members when trying to recover delays [5]. This situation also implies dedicating more time to train new team members.

3 Post-Mortem Analysis of a GSD Project without Mitigation of Threats

The present proposal arises due to the identification of a highly negative impact on the quality of the software, the estimated deadlines and budget, in the late phases of a large software project. The project, which belongs to the insurance domain, had a budget of $9 million dollars, an estimated deadline of 10.5 months and the participation of

approximately 100 people. It was the first of a program of projects whose teams were scattered in several countries: analysts in Buenos Aires (Argentina), developers in Pune (India), testers in Chennai (India), configuration suppliers from Poland and core applications suppliers from Denmark (Fig. 5). The users of this first software project were located in Caracas (Venezuela). The second software project, whose users were in São Paulo (Brazil), had the same geographic distribution of teams, although with other composition and other members. In this second project, the proposal for mitigating threats to requirements was applied, based on the difficulties encountered in the first project.

Figure 5: Teams Distribution in Venezuelan and Brazilian projects

The development strategy for these two projects consisted in moving the analysts from Buenos Aires to the country where the users were located, in order to perform requirements elicitation and preliminary modeling. The planning, requirement specifications and design were performed at Buenos Aires. Subsequently, the transition to the software factory took place in Pune, in order to start the development. In this factory, the requirement specifications were assigned to developers, who fulfilled their doubts with the analysts. After the development, integration testing was made in Chennai; the testing team deliver the software to the destination country, where the software certification tests took place. Resuming, in both projects it was used the cascade life cycle with the following seven phases:

i) *Conceptualization Phase* involves analysis activities, feasibility activities, tools and infrastructure set up, configuration, scope and business requirements definition, and preliminary architecture design;

ii) *Initiation Phase* involves risk management and planning activities, project kick-off meeting, and training of the team;

iii) *Analysis and Design Phase* involves management activities to update project planning, test plan definition, analysis activities, architectural design, data model preparation, classes design, and test cases creation;

iv) *Transition to Software Factory and Development Phase* involves activities of transferring requirement specifications to software factory, code generation, unit test, user manual creation, and elaboration of installation and operation manuals.

v) *Testing Phase* involves the execution of integration tests and user acceptance tests;

vi) Implementation Phase involves project management activities, deliverables acceptance and closing meetings, training activities for end users, generation of support plan, implementation of deliverables in production environments, and knowledge management activities by creating knowledge transfer documentation for the support team;

vii) Post-Implementation Phase involves lessons learned meetings, release of project resources and post-implementation review.

The requirements process in the Venezuela project was performed along the first and third phase, and their activities are detailed below.

In the *Conceptualization Phase*, the scope and business requirements definition activity involves:

- Elicitation and analysis of business requirements
- Creation of a business requirements record
- Creation of a traceability matrix
- Validation of business requirements through sessions
- Formal approval of business requirements

During the *Analysis and Design Phase*, the analysis activity involves:

- Prioritization of business requirements
- Creation of requirement specifications
- Validation of requirement specifications through sessions
- Formal approval of requirement specifications

After approval of requirement specifications, during sub-sequent phases, an activity of requirement specifications evolution was performed, despite being a waterfall model.

In both projects, every model and document was written in English, being the common language adopted due to language differences of all the stakeholders.

Both projects used a set of predefined metrics provided by the insurance company, which were calculated based on the collected information of each activity. Table 1 only shows the subset of metrics related to requirements. The estimation for the activities was based on the historical information of the company. The metrics related to requirements were estimated according to the complexity of the specification: Low, Medium, and High, also distinguishing them by analysts or developers with or without knowledge of the insurance domain. The complexity of the specifications was established with the help of a functional point estimator.

Table 1: Metrics related to requirements, used in Venezuela and Brazil projects

Code	Indicator	Definition	Phase	Unit of Measurement
DFE01	Specification creation	Average time for creating a requirement specification	Analysis and Design	Day
DFA01	Specification approval	Average time for approving a requirement specification	Analysis and Design	Day
DFR01	Specification rejection	Rejection average per specification during user approval	Analysis and Design	Numeric
TRATPT01	Transition time to software factory	Average time for transitioning a specification to the software Factory	Transition to Factory	Hour
TRACPDF01	Questions per Specification	Average amount of questions per requirement specification	Transition to Factory	Numeric
CERTNCFF01	Defect % originated in specification	Percentage of specifications defects compared to the total amount	Testing: Users Certification	Percentage
CERTINCPER01	Defect rate per specification	Defect index per specification due to defect in requirements	Testing: Users Certification	Numeric

During the certification of the software in Venezuela, many rejections from users were identified; those rejections turned on alarms in the new project that was about to start, whose destination country was Brazil. These rejections were properly documented, and it was determined that 41% came from the requirements. Based on the analysis of the calculated metrics, a proposal was elaborated to reduce rejections in the Brazil project.

In the Venezuela project, 20 users from Caracas were involved, 13 analysts from Buenos Aires, 38 developers

from Pune and 8 testes from Chennai. The 69% of the analysts had no knowledge in the insurance domain, and the 66% of developers either. 104 requirement specifications were created: 43% were of Low complexity, 34% Medium, and 23% High. The first ones had an average of 18 pages, the Medium complexity ones had an average of 48 pages and the last ones 82 pages.

3.1 Requirements Metrics Calculation for Venezuela Project

Details of the metrics obtained in the Venezuela project are presented below. The value of the DFE01 metric presents substantial differences between the estimation and the actual times for creating specifications. Table 2 also shows that the creation time of a specification made by an analyst with knowledge in the domain is clearly lower than the one created by the inexperienced one.

Table 2: Variation between effective and estimated time for creating specification (days)

Complexity	Estimated Time		Effective time		Variation	
	Experts	Non-experts	Experts	Non-experts	Experts	Non-experts
Low	6	8	7	14	17 %	75 %
Medium	10	18	13	29	30 %	61 %
High	15	20	22	32	47 %	60 %

The average time to get the formal approval of the users (DFA01 metric) was significantly higher than the estimated value, 60% higher on average (Table 3).

Table 3: Variation between effective and estimated time for specification approval (days)

Complexity	Estimated Time		Effective time		Variation	
	Experts	Non-experts	Experts	Non-experts	Experts	Non-experts
Low	2	5	3	9	50 %	80 %
Medium	4	9	6	13	50 %	44 %
High	6	10	10	19	67 %	90 %

The average of rejections during the approval of the requirements (DRF01 metric) was considerably higher in those specifications that were created by inexperienced analysts than those made by the experienced ones (Table 4), 132% higher on average.

Table 4: Average variation of rejections by specification and analyst's category

Complexity	Rejection average by specification		Rejection variation
	Expert analysts	Inexpert analysts	
Low	2	5	150%
Medium	4	9	125%
High	5	11	120%

During the transition phase to the software factory, the overlapping of the working hours between Buenos Aires and Pune was only of 2 hours a day and the chosen language was English, which was not the native language of any of the teams. Every developer made a previous reading of the assigned requirement specification to familiarize with it. Then, they clarified their doubts with the analyst during a session limited to those 2 hours of overlapping. The communication tools used were: Phone conference, Video conference and a Q&A management tool.

Table 5 represents the estimated and effective times for the transition to the factory by type of specification and by analyst category (TRATPT01 metric). The transitions of analysts with domain knowledge were close to the estimated values, however the transitions of the analysts without knowledge was, on average, 19% higher than the estimation. The developers made on average 13.2 questions by specification to clarify doubts (TRACPDF01 metric). Inexperienced developers asked on average 37% more questions than the experienced ones (Table 6). That lack of domain knowledge and of the terminology resulted in losing a significant part of the synchronous time in clarifying basic concepts.

Table 5: Variation between effective and estimated time for transitioning specification to software factory (hours)

Complexity	Estimated Time		Effective time		Variation	
	Experts	Non-experts	Experts	Non-experts	Experts	Non-experts
Low	8	11	8	13	0 %	18 %
Medium	18	24	19	28	6 %	17 %
High	22	29	24	35	9 %	21 %

Table 6: Average variation of questions by specification and developer category

Complexity	Questions average per specification		Questions variation
	Expert developers	Inexpert developers	
Low	7	9	29 %
Medium	11	15	36 %
High	15	22	47 %

During the software certification phase, 238 defects were reported divided in 9 types, 98 of which came from requirements (41% of the total, CERTNCFF01 metric) (Fig. 6). This 98 defects were originated in 75 of the 104 created requirement specifications (72% of the total), which implies that some of them had more than 1 defect. From these 75 specifications, 73% of them were created by inexperienced analysts. The rate of defects in requirements (CERTINCPER01 metric) was 0.94 defects by specification.

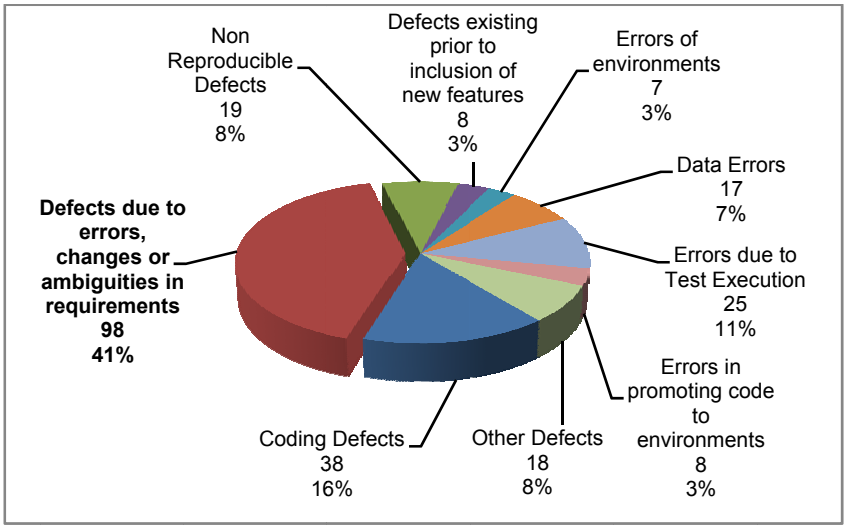

Figure 6: Detected Defects in the Software Certification of Venezuela Project

The defects mentioned in Fig. 6 represent:

- Non-reproducible defects: they are not able to be reproduced in the testing environments.
- Defects existing prior to inclusion of new feature: they are not related to the functionality introduced with the last promoted code.
- Errors of environment: they are defects due to limitations or characteristics of the environment, such as lack of memory, disk size, among others.
- Data errors: they are a consequence of inconsistency of data.
- Errors due to test execution: they are due to deficient execution of the test cases, manual processing, not running, among others.
- Errors in promoting code to environments: they are due to errors in the promotion process, such as corrupt packages, wrong promotion sequence, among others.
- Coding defects: they are a consequence of wrong code generation.
- Defects due to errors, changes or ambiguities in requirements: they are introduced in the requirement

specifications, such as ambiguities, omissions, inconsistencies and changes to requirements, and are not identified in earlier phases of the development process.

- Other defects: they are those defects not included in any other category, such as errors of permissions and users.

Table 7 summarizes the threats that affected the Venezuela project, identifying the requirement metrics associated to those threats.

Table 7: Threats identified in Venezuela Project

Threat	Impact on:
Inadequate Communication	• Requirement specification creation (DFE01) • Requirement specification approval (DFA01) • Requirement specification rejection (DFR01) • Questions per requirement specification (TRACPDF01) • Transition time to software Factory (TRATPT01)
Geographical and Temporal Distances	• Transition time to software Factory (TRATPT01) • Questions per requirement specification (TRACPDF01)
Language and Cultural Barriers	• Average time for transitioning to software factory (TRATPT01) • Questions per requirement specification (TRACPDF01)
Knowledge Management problems	• Average time for transitioning to software factory (TRATPT01) • Questions per requirement specification (TRACPDF01)
Lack of Confidence and Engagement	• Requirement specification creation (DFE01) • Requirement specification approval (DFA01) • Requirement specification rejection (DFR01)
Lack of Knowledge about Problem Domain	• Requirement specification creation (DFE01) • Requirement specification rejection (DFR01) • Questions per requirement specification (TRACPDF01) • Transition time to software Factory (TRATPT01)
Ambiguity, Contradictions and lack of clarity in specifications	• Requirement specification rejection (DFR01) • Transition time to software Factory (TRATPT01) • Questions per requirement specification (TRACPDF01) • Defect % due to defect specifications (CERTNCFF01) • Defect rate per specification (CERTINCPER01)

It is assumed that many of the threats affected the requirements, since the majority of the defects detected in the certification phase had their origin in the requirement specifications. On the other hand, the project had severe deviations in costs (53% increment over budget), as well as, in deadline (48% increment over planning time). The largest variation in costs was due to the delay in the deadline. Due to the kind of contract with the supplier, it was necessary to extend the contracts of the whole team, which had a strong impact on the total cost. The time extension indirectly arose due to the rejection of the requirement specifications during formal approval, the amount of developer questions, and transition times to the software factory. While the amount of defects by specifications seems to be low, the most time-consuming activities seem to come from the requirements stage.

There were no problems of rotation of team members, since specific strategies have been implemented for the retention of the staff. Moreover, there were penalties for suppliers in case the established attrition rate would increase a certain percentage.

Individual goals of the distributed teams did not affect the project. The supplier of development and testing, selected for both projects, had been working with the company for several years. Therefore, the supplier knew in detail the regional strategy and the leadership of the insurance company and of other suppliers, and was aligned with the company strategies allowing him to make the right decisions in order to achieve the commitment and alignment of every team members with the shared project goals. Management committee meetings were done monthly; in these meetings not only the project results were presented, but also the company and the suppliers exposed their concerns, risks and respective mitigation actions in order to align goals and priorities of the project teams.

There were no problems regarding tools, such as video conference, shared desks, and management tools, since they were available to the team. This means that no technical problems have appeared.

4 Proposal for Mitigating Threats to Requirements in GSD

The Brazil project was similar to the Venezuela one, regarding the characteristics of the software to be built. Although the number of requirement specifications was higher, the process to follow and the planning were similar. This project was planned with an estimated deadline of 16.5 months and a budget of $ 14.5 million, participating 50% more people than in the Venezuela project. The project was initially proposed to start 8 months later than the beginning of the Venezuela project, but finally began 13 months later due to the need to implement appropriate mitigation actions.

Although, the software development process followed in Brazil was similar to the one in Venezuela, some activities were incorporated in the different phases to mitigate the possible occurrence of threats that had affected requirements in the Venezuela project. The proposal was based on the use of natural language models: LEL, Current Scenarios and Future Scenarios, whose contributions were described on subsection 2.1. The highlights of these models are communication among stakeholders, reduction in ambiguities and inaccuracies by using LEL symbols in every created model, understanding of the problem through the Current Scenarios, and understanding of the interaction between the future software and its context through the Future Scenarios. The definition and management of these models required additional activities in several phases of the development process (described in section 3). Besides, a repository to share these models was incorporated in the Brazil project.

In the preparation of infrastructure and tools activity within the *Conceptualization Phase*, a sub-activity was included for the selection, installation and configuration of knowledge management tools. Another sub-activity incorporated in this Phase was an induction of the analysts into the creation of LEL, Current Scenarios and LEL Scenarios.

Additionally, in the *Conceptualization Phase*, the scope and business requirements definition activity was re-designed adding several new activities. Therefore, this activity involves the following sub-activities (new sub-activities are marked in italics):

- *Creation of Language Extended Lexicon*
- *Validation of LEL through sessions*
- *Creation of Current Scenarios*
- *Validation of Current Scenarios through sessions*
- *Storage of LEL and Current Scenarios in de knowledge management repository*
- *Formal approval of LEL*
- *Formal approval of Current Scenarios*
- Elicitation and analysis of business requirements
- Creation of a business requirements record
- Creation of a traceability matrix
- Validation of business requirements through sessions
- Formal approval of business requirements
- *Creation of Future Scenarios*
- *Validation of Future Scenarios through sessions*
- *Storage of Future Scenarios in de knowledge management repository*
- *Formal approval of Future Scenarios*

In the *Initiation Phase*, the project kick-off meeting activity included the presentation of the LEL and Scenarios models, and of the knowledge management repository to the stakeholders. Within the same Phase, regarding the team's training, the LEL and Scenarios models and the knowledge management repository were presented and explained to the developers and to new analysts. Training took place through meetings held in each of the locations. The main Current Scenarios were presented in detail, providing a high-level view of the essential characteristics of the business. Then, it was delved into the rest of those scenarios to give a correct view of the interaction of different areas, such as, claim and policy administration. It was presented the derivation towards Future Scenarios, which

provided a global view of the system goals. During these sessions, the necessary information related to the LEL, to Current Scenarios and to Future Scenarios was provided, such as the location of these models within the repository of information and the access to them, among other information.

In the *Analysis and Design Phase*, the creation, validation and formal approval of the requirement specifications continued being performed, but these ones were created taking into account the vocabulary defined in the LEL and based on the created Future Scenarios. These specifications were also stored in the knowledge management repository.

During the following phases of the process, the activity of evolution of LEL, of Current Scenarios and of Future Scenarios was added, aligned with changes in business requirements or corrective changes due to a better understanding of the problem domain.

In the *Implementation Phase*, the LEL, Current Scenarios and Future Scenarios were made available for the support team, together with the requirement specifications.

The team was composed of 60 users in São Paulo, 16 analysts in Buenos Aires, 47 developers in Pune and 8 testers in Chennai. Just like in the Venezuela project, there was a similar proportion of analysts and developers with no experience in the Insurance domain. This was, in both projects, as consequence of the lack of available resources in the labor markets of Buenos Aires and Pune.

In the Brazil project, an LEL with 46 symbols, 26 Current Scenarios, 97 Future Scenarios and 154 requirement specifications were created. The amount of pages by specification was similar to the Venezuela project, with a similar composition regarding the complexity. All models were written in English, due to the diversity of languages that the teams had, and were also written using a unified terminology provided by the LEL model.

4.1 Metrics Calculated for the Brazilian Project

Tables 8 to 12, show the calculated values for the Brazil project, which are equivalent to the Tables 2 to 6 of sub-section 3.1. The same time estimations were used in both projects. Comparing Tables 2 and 8, improvements in the time used for creating a specification may be observed.

Table 8: Variation between estimated and effective time for creating a specification (days)

Complexity	Estimated Time		Effective time		Variation	
	Experts	Non-experts	Experts	Non-experts	Experts	Non-experts
Low	6	8	6	10	0 %	25 %
Medium	10	18	13	21	30 %	17 %
High	15	20	21	25	40 %	25 %

Table 9 in contrast with Table 3, shows an important reduction in time consumed for requirements approval versus the estimation for inexperienced analysts. Analyzing the duplication of time for specification approval, related to low complexity specifications created by expert analysts, has only been able to conclude that the estimation of 2 days was an optimistic one.

Table 9: Variation between estimated and effective time for approving a specification (days)

Complexity	Estimated Time		Effective time		Variation	
	Experts	Non-experts	Experts	Non-experts	Experts	Non-experts
Low	2	5	4	6	100 %	20 %
Medium	4	9	6	11	50 %	22 %
High	6	10	9	13	50 %	30 %

On the other hand, a substantial decrease in the rejections average for medium and high complexity specifications created by inexperienced analysts versus experts, may be observed in Table 10.

Table 10: Average variation of rejections by specification and analyst category

Complexity	Rejection average per specification		Rejection variation
	Expert analysts	Inexpert analysts	
Low	2	5	150 %
Medium	5	8	60 %
High	6	9	50 %

In the transition to the software factory in the Brazil project, the effective versus estimated times were reduced although in low proportion compared to Venezuela project (see Tables 5 and 11).

Table 11: Variation between estimated and effective time for transitioning a specification to software factory (hours)

Complexity	Estimated Time		Effective time		Variation	
	Experts	Non-experts	Experts	Non-experts	Experts	Non-experts
Low	8	11	7	12	-13 %	9 %
Medium	18	24	17	29	-6 %	21 %
High	22	29	24	34	9 %	17 %

In the Brazil project, the number of questions from developers compared to Venezuela was considerably reduced, being on average 8.5 questions by specification. It was feasible to equal the same number of questions for inexperienced and expert developers, except for highly complex specifications (Table 12).

Table 12: Average variation of questions by specification and developer category

Complexity	Questions average per specification		Questions variation
	Expert developers	Inexpert developers	
Low	5	5	0 %
Medium	9	9	0 %
High	9	14	56 %

During the certification phase, 254 defects were detected, being the 32% of them defects coming from requirements (Fig. 7). These defects were originated in 69 specifications, from a total of 154 ones. This leads to conclude that the 45% of the specifications contain defects, a lower value than in the Venezuela project. As in Venezuela, inexperienced analysts have created the biggest amount of specifications with defects. The rate of defect in requirements per specification was 0.53, almost half of Venezuela project.

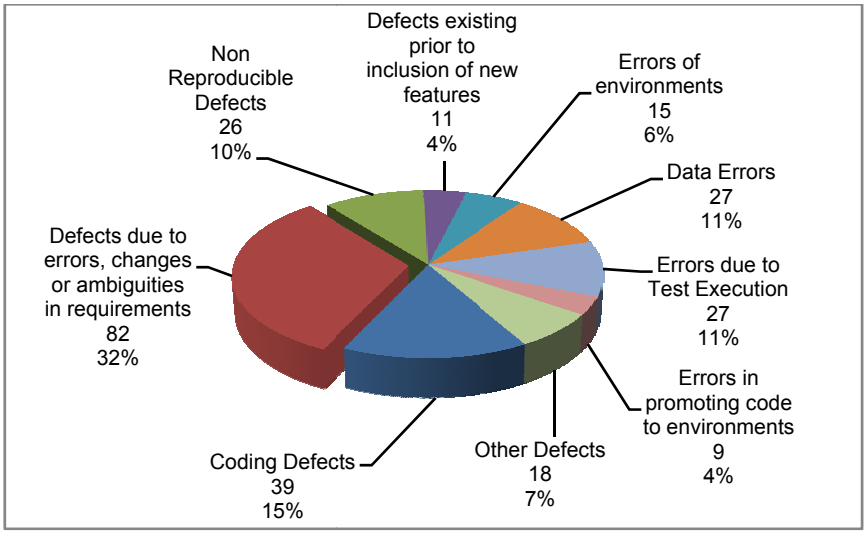

Figure 7: Detected Defects in Software Certification of Brazil Project

4.2 Comparative Analysis of Metrics from both Projects

Table 13 shows that Venezuela's teams and Brazil's teams have a similar composition of resources regarding those with knowledge in the insurance domain and those without that knowledge, representing the last ones a high percentage of the total of resources. Considering the total of stakeholders involved in each project, the Brazil project had 50% more participants than the Venezuela project. This was mainly due to the number of users: 20 in Venezuela and 60 in Brazil, resulting in tougher tasks in elicitation, in validation of specifications, and then in the certification tests with users.

Table 13: Composition of analysts and developers teams in Brazil and Venezuela projects

	Venezuela		Brazil	
Resources	Analysts	Developers	Analysts	Developers
With domain knowledge	4	13	5	15
Without domain knowledge	9	25	11	32
Total Resources	13	38	16	47
% of resources without domain knowledge	69 %	66 %	69 %	68 %

In the Brazil project, both analysts with and without expertise in the domain decreased the time needed for creating requirement specifications, independently of the complexity of the specification, in comparison with the Venezuela project (Table 14). In case of the inexperienced analysts, the reduction in the creation time compared to Venezuela was in average 26%, while for the experts was in average 6%. In the Brazil project, there was a significant reduction in effective creation time respect to estimation creation time, mainly for those specifications created by non-experts.

Table 14: Variation in effective creation time of requirement specifications between Brazil and Venezuela

	Venezuela (days)		Brazil (days)		Variation time	
Complexity	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts
Low	7	14	6	10	-14 %	-29 %
Medium	13	29	13	21	0 %	-28 %
High	22	32	21	25	-5 %	-22 %
			Average Variation		-6 %	-26 %

The average approval times for those specifications created by expert analysts almost kept unchanged between the two projects, being even close to the estimated time. On the contrary, approval times for specifications created by inexpert analysts decreased in the Brazil project.

As shown in detail in Table 15, the time used for the formal approval of the specifications was reduced in the Brazil project, except for low complexity specifications created by expert analysts. Approval time for those specifications have increased in average 33%, which means in absolute values an increment of 1 day per specification; this could be considered insignificant (see numbers in red in Table 15). Furthermore, analyzing in absolute values the 33% of average reduction in time approval for non-experts, that reduction implies in average 3 days less for approving specifications than the time taken in the Venezuela project (see numbers in green in Table 15).

Table 15: Variation in effective approval time of requirement specifications between Brazil and Venezuela

	Venezuela (days)		Brazil (days)		Variation time	
Complexity	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts
Low	3	9	4	6	33 %	-33 %
Medium	6	13	6	11	0 %	-15 %
High	10	19	9	13	-10 %	-32 %
			Average Variation		8 %	-27 %

Table 16 shows the variation of number of rejections per requirement specification during the formal approval task. There was only a minimal reduction in the number of rejections of specifications created by inexperienced analysts, while an increment occurred for medium and high complexity specifications created by experts in

comparison with the Venezuela project. As pointed out in case of approval time, the average values show that the number of rejections increased in one per type of specification. However, it is important to observe that the rejections in the Brazil project were originated in changes or minor errors.

Table 16: Variation in number of rejections per requirement specification between Brazil and Venezuela

	Venezuela		Brazil		Variation in rejections	
Complexity	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts
Low	2	5	2	5	0 %	0 %
Medium	4	9	5	8	25 %	-11 %
High	5	11	6	9	20 %	-18 %
			Average Variation		15 %	-10 %

The time consumed for transitioning the specifications to the software factory were slightly decreased respect to the Venezuela project (Table 17), while there was a significant reduction in the number of questions (Table 18). The first is due to the limitation of overlapping hours between Buenos Aires and Chennai for synchronous communication. The Venezuela's developers did on average 13.2 questions per specification, while in the Brazil project the average was 8.5 questions. There was an important decrease in questions done by Brazilian inexpert developers, on average 40 % less than those done by Venezuelan inexpert developers. Moreover, in the Brazil project it was narrowed the gap between the number of questions asked by inexperienced developers face to those asked by experts.

Table 17: Variation in transition time per requirement specification between Brazil and Venezuela

	Venezuela (hours)		Brazil (hours)		Variation time	
Complexity	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts	Expert analysts	Inexpert analysts
Low	8	13	7	12	-13 %	-8 %
Medium	19	28	17	29	-11 %	4 %
High	24	35	24	34	0 %	-3 %
			Average Variation		-8 %	-2 %

Table 18: Variation between Brazil and Venezuela in number of questions per specification during transition to factory

	Venezuela		Brazil		Variation questions	
Complexity	Expert developers	Inexpert developers	Expert developers	Inexpert developers	Expert developers	Inexpert developers
Low	7	9	5	5	-29 %	-44 %
Medium	11	15	9	9	-18 %	-40 %
High	15	22	9	14	-40 %	-36 %
			Average Variation		-29 %	-40 %

Although it seems that the number of non-conformities on requirement specifications reported by users during software certification in the Brazil project is slightly lower than the one of the Venezuela project, it should be noticed that a greater number of requirement specifications (increment of 48 %) were developed in the Brazil project (Table 19). Therefore, the defect rate per specification in Brazil was almost half of the Venezuela rate.

Table 19: Comparison of defect rate per requirement specification between Brazil and Venezuela

	Venezuela	Brazil	Variation
Number of specifications	104	154	48 %
Number of detected defects	98	82	-16 %
Defect Rate	0,94	0,53	-44 %

The result of the software certification in the Brazil project identified that the 45 % of the total requirement specifications (154) had defects; this percentage was much lower than in the Venezuela project, where 72 % of the

specifications contained defects (Table 20).

Table 20: Comparison of specifications reported with defects in software certification between Brazil and Venezuela

	Venezuela	Brazil
Number of specifications	104	154
Number of specifications with defects	75	69
Percentage of specifications with defects	72 %	45 %

In both projects, inexperienced analysts created most specifications with detected defects. Table 21 shows that the number of defects in specifications created by inexperienced analysts was highly reduced in the Brazil project against the Venezuela project. The nonconformities found in specifications created by inexperienced analysts of the Brazil project was 93% higher than those detected in specifications created by experts, while in the Venezuela project this value was 163%. That is, there was a significant reduction in the percentage of the Brazil project against the one of the Venezuela project.

Table 21: Variation between Brazil and Venezuela in number of defects reported according to specifications created by category of analysts

	Venezuela	Brazil
Number of detected defects in specifications	98	82
Number of defects in specifications created by expert analysts	27	28
Number of defects in specifications created by inexperienced analysts	71	54
Variation in number of defects between experts and non-experts	163 %	93 %

Table 22 summarizes the metrics obtained for both projects, showing important improvements in several of these metrics. It is significant the reduction in time and cost deviation for Brazil project, as well as the reduction, almost by half, of the defect rate per specification during the software certification phase. On average, there was a slight reduction in the number of rejections during approval, as well as in the transition time, being more significant for inexperienced analysts.

Table 22: Metrics comparison between Venezuela and Brazil

Metric	Venezuela	Brazil
Average days for creating a specification	20	16
Average days for approving a specification	10	8.2
Amount of rejection during specification approval	6	5.8
Average hours for transitioning to software	21.2	20.5
Questions per requirement specification	13.2	8.5
Defect rate per specification (certification)	0.94	0.53
% of defect specifications considering the total amount (certification)	41 %	32 %
Percentage of specification with defects compared to all specifications (certification)	72 %	45 %
Percentage of cost deviation	53 %	26 %
Percentage of time deviation	48 %	18 %

Regarding the validity of the results shown above, some issues should be remarked:

- Though both projects were developed for the same insurance company, they were performed independently of each other.
- Analysts, developers and testers were hired independently for each project at each location, since there was some overlap in the development of the two projects. The same process activities were done at the same

locations, thus the teams at each location had the same culture. Obviously, users of each project came from each software target country, with their own culture.

- The same type of development process was performed in both projects, despite the additional activities included in the Brazil project to help mitigate threats to requirements.
- The estimation of each activity time, depending on the complexity of the requirement specification, was done using function points with a common estimator, independent of the different locations, suppliers and projects.
- The same metrics were applied in both projects, and measures were obtained in the same way.
- Estimated time and cost of the Brazil project did not include the creation and validation activities of the LEL, Current Scenarios and Future Scenarios, since the budget and planning was made before knowing the threats that had affected the Venezuela project.

4.3 Mitigation to Requirements Threats in Brazil Project

Based on the comparative analysis of results between the Venezuela and Brazil projects, signs of significant improvements on certain aspects and in other cases partial mitigations of requirements threats were observed. In summary, the values of the Brazil project in contrast with the Venezuela give evidence of:

- Improvement in quality communication, using the Language Extended Lexicon to standardize the vocabulary of the domain throughout the entire software process. Both, Current Scenarios and Future Scenarios, also provided a communication medium, allowing a better understanding for the stakeholders, in addition those models were also written in the domain terminology. Such terminology was the regular used by users and other human resources, which know the insurance domain, while in the case of those without such knowledge, they had these clear terminology definitions through the LEL model.
- Improvement in trust and engagement of the project teams, using models written in natural language with a unified vocabulary through the LEL.
- Better knowledge of the problem domain, through scenarios that gave a proper view of the problem: the current behavior (Current Scenarios) and the planned behavior (Future Scenarios). Moreover, understanding the context vocabulary is the beginning of the cognitive process. This was mainly observed in those resources with poor or no insurance domain knowledge.
- Reduction of ambiguities, contradictions and lack of clarity in the specifications, by means of the Future Scenarios, since these ones were created in natural language, collaborating on the requirements validation with users, preventing the inclusion of defects in requirement specifications and improving the deadlines. In addition, creating specifications using the symbols included in the LEL helped to reduce ambiguity. This was notoriously evident with the decrease of the defect rate per specification.
- Improvements in knowledge management, through the use of the knowledge repository, which contains descriptions in natural language of the of vocabulary (LEL), the problem under study (Current Scenarios) and the system to be built (Future Scenarios).
- Partial mitigation of the problems related to geographical and temporal differences through the LEL and Future Scenarios, since these models were a valid source for creating requirement specifications and also during the transition to the software factory by allowing clearing doubts or by taking faster decisions, minimizing the number of questions from developers to analysts and focusing on relevant matters or issues that were more complex to understand.
- Partial mitigation of the problems related to cultural and language barriers, through the LEL model, which unified the vocabulary for stakeholders interaction, providing shared terms with a precise meaning.

5 Related Work

The threats found in the Venezuela project are mostly typical of GSD. When analyzing the work of Hanisch and Corbitt in the project Sapphire Software House [23], similarities can be found related to the difficulties faced by the Venezuela project. The team was scattered in various locations, the UK and New Zealand, with cultural and temporal differences, combined with inexperienced resources; this resulted in inaccurate interpretations by analysts,

who introduced several defects in the specifications. This is similar to the consequences suffered in the Venezuela project.

In the Sapphire project [23], it was considered that the major threat was related to communication, which also generated misunderstanding problems, affected the trust relation among those involved, arriving to a similar conclusion as in Venezuela project. As a mitigation action, the most experienced resources were relocated at user offices in UK, which affected the remaining team by delaying the understanding of the requirement specifications coming from UK, and increasing considerably the amount of questions. This is a similar consequence that happened in the Venezuela project, mainly in the transition to the software factory. To mitigate this new problem, they included new tools in order to improve communication and knowledge management, since they were only using email. The implemented tools, already in place in both Venezuela and Brazil projects, enabled the Sapphire project to perform the tasks more efficiently, although there is not a detailed explanation about the improvements.

In short, the problems identified in both projects, Sapphire and Venezuela were similar, but the mitigation options implemented were different. While in the Sapphire project the relocation of the most experienced resources was chosen in order to improve the communication and increase the confidence, and subsequently synchronous communication tools were included; in the Brazil project, the models LEL, Current Scenarios and Future Scenarios were used, complemented with a repository containing those models. Unlike the Sapphire project, the implemented models allowed not only improve communication but also the knowledge of those involved, providing useful tools for the execution of their tasks in a remotely and asynchronous way. It is important to mention that in the Brazil project the improvements obtained were quantified and compared to the Venezuela project.

In the work of Aranda et al. [24], the communication is also mentioned as the main problem faced in GSD, being time differences, cultural differences and knowledge management mentioned as well. An important concept, emphasized by the authors, is the necessity of a common understanding of the domain; to do so, ontologies as facilitators of communication were used. This proposal is relatively similar to the advantages provided by the LEL model, although the last one is strictly focused on the vocabulary used in the context, and it is complemented by the Current Scenario model to provide a proper understanding of the context's behavior [18].

Aranda et al. [3] mention the necessity of carefully select the requirements elicitation techniques and groupware tools, depending on the cognitive characteristics of the involved team, considering especially the lack of the GSD model to allow a fully synchronous communication. In Venezuela and Brazil projects, tools were already implemented and available for the team members, but it is interesting the approach mentioned by Aranda et al. related to the affinity of some teams to a particular way of communication, such as visual and verbal ones, and the use of this analysis to define the most appropriate instruments. A problem presented in this proposal is how to handle the diversity of antagonist cognitive characteristics of those involved, probably making almost impossible the use of a large amount of tools. They have presented some results based on controlled experiments [3]. In short, the proposal provided by Aranda et al. [3] focuses on the use of ontologies and groupware tools, mainly centralized in improving communication, understanding of the problem and knowledge management.

6 Conclusions

The GSD is widely spread in the software industry [9], and this is the reason to pay special attention to this working model. Even though it has great benefits, these ones could be reduced when ignoring or underestimating the particular characteristics of this model. Therefore, these peculiarities should be considered appropriately in the development process, since they could become threats that put in risk the quality of the process and the product [3], [4], [5], [6], [8], [12], [19], [24]. Many of these threats affect the requirements in different levels and ways, causing undesirable effects, which could be spread to the next development phases if not identified from the beginning [7]. The identification of existing, new or potentials threats, when selecting the distributed model, is the first step towards a proactive strategy to mitigate them. The proposal that has been presented, using the LEL, Current Scenario and Future Scenario models [15], [18] tries to provide benefits in mitigating requirements threats in GSD projects.

Both projects, Venezuela and Brazil presented here, had similar features from a technical perspective, due to the use of the same processes, metrics, estimation model and tools. Based on the calculated metrics for the Brazil project, it is quite evident that the use of these natural language models, together with a common repository containing them, allowed mitigating threats previously identified in the Venezuela project. The main threats attacked by the proposal were problems on: communication, domain knowledge, ambiguity and imprecision of specifications, and teams involvement. The evidence has shown that the proposal has provided greater benefits for those resources, analysts and developers, who had no knowledge of the problem domain.

This proposal is aligned with what was exposed in [8], which claims out the necessity of a different requirements

engineering process for those GSD projects. The characteristics of Venezuela and Brazil projects are similar to many other GSD projects, as it could be seen in the literature, which would suggest that the improvements in the quality of requirement specifications using the LEL model, the Scenario model and the repository might also be obtained in similar projects. Furthermore, it is quite common to have teams with different level of experience or knowledge about the problem domain, mainly when they come from several countries or even cities, and the present proposal may level off the knowledge of the team members.

A potential improvement in the present proposal is the creation of an additional LEL, containing the vocabulary used in the Future Scenarios and in the requirement specifications, since new terms not defined in the initial LEL may appear or the meaning of existing terms may require modification. This may happen due to either changes in business processes or technological changes by the incorporation of the software [25].

A subject not analyzed in this work, and proposed for future is how the LEL and Future Scenarios could assist the testing team in the understanding of terminology and system flows, since this testing task is often entrusted to a supplier different to the one who carries out the development. A potential alternative to facilitate testing could be the generation of test cases from Future Scenarios, an activity that has already been developed using natural language descriptions [26], [27], [28]. It is considered likely that the improvements achieved in the Brazil project could be enhanced in the testing phases.

References

1. B. Nuseibeh and S. Easterbrook, “Requirements Engineering: A Roadmap”, in *Proc. Future of SE Track 2000*, Limerick, Ireland, pp. 35-46, 2000. Available: http:// dx.doi.org /10.1145/336512.336523
2. T. Bucher, M. Klesse, S. Kurpjuweit and R. Winter, “Situational Method Engineering”, in *Situational method engineering: fundamentals and experiences*, Springer, pp. 33-48, 2007. Available: http://dx.doi.org/10.1007/978-0-387-73947-2_5
3. G. N. Aranda, A. Vizcaíno, A. Cechich and M. Piattini, “A Methodology for Reducing Geographical Dispersion Problems during Global Requirements Elicitation”, in *Proc. 11th Workshop on Requirements Engineering (WER2008)*, Barcelona, Spain, pp.117-127, September 2008.
4. D. E. Damian, and D. Zowghi, “RE challenges in multi-site software development organizations”, *Requirements Engineering Journal*, vol. 8, no. 3, pp. 149-160, 2003. Available: http://dx.doi.org/10.1007/s00766-003-0173-1
5. I. Richardson, V. Casey, F. McCaffery, J. Burton, and S. Beecham, “A Process Framework for Global Software Engineering Teams”, *Information and Software Technology*, vol. 54, no. 11, pp. 1175-1191, 2012. Available: http://dx.doi.org/10.1016/j.infsof.2012.05.002
6. M. Niazi and S. Shastry, “Role of requirements engineering in software development process: an empirical study”, in *Proc. 7th International Multi Topic Conference*, pp. 402-407, 2003. Available: http://dx.doi.org/10.1109/INMIC.2003.1416759
7. N. Nahar, P. K wora, and S. Kumaresh, “Managing Requirement Elicitation Issues Using Step-Wise Refinement Model”, *International Journal of Advanced Studies in Computers, Science and Engineering*, vol. 2, no. 5, pp. 27-33, 2013. Available: https://arxiv.org/ftp/arxiv/papers/1311/1311.1729.pdf
8. D. Zowghi, “Does Global Software Development Need a Different Requirements Engineering Process?”, in *Proc. International Workshop on Global Software Development (in conjunction with ICSE 2002)*, Orlando, Florida, pp. 53-55, 2002.
9. J. D. Herbsleb, “Global Software Engineering: The Future of Socio-technical Coordination”, in *Proc. 2007 Future of Software Engineering (FOSE '07)*, IEEE Computer Society, Washington, pp. 188-198, 2007. Available: http://dx.doi.org/10.1109/FOSE.2007.11
10. G. N. Aranda, A. Vizcaíno, A. Cechich and M. Piattini, “Strategies to minimize problems in global requirements elicitation”, in *Proc. CLEI electronic journal*, vol. 11, no. 1, paper 3, 2008.
11. R. Prikladnicki, J. L. N. Audy, and R. Evaristo, “Global Software Development in Practice Lessons Learned”, *Software Process: Improvement and Practice*, vol. 8, no. 4, pp. 267-281, 2003. Available: http://dx.doi.org/10.1002/spip.188
12. K. Fryer and M. Gothe, “Global Software Development and Delivery. Trends and Challenges”, IBM Developer Works, The Rational Edge, 2008. http://www.ibm.com/developerworks/rational/library/edge/08/jan08/fryer_gothe/
13. J. M. Bhat, M. Gupta and S. N. Murthy, “Overcoming requirements engineering challenges: Lessons from offshore outsourcing”, *IEEE Software*, vol. 23, no. 5, pp. 38-44, 2006. http://dx.doi.org/10.1109/MS.2006.137
14. C. J. Davis, R. M. Fuller, M. C. Tremblay and D. J. Berndt, “Communication challenges in requirements elicitation and the use of the repertory grid technique”, *Journal of Computer Information Systems*, vol. 46, no. 5, pp. 8-86, 2006. Available: http://www.tandfonline.com/doi/abs/10.1080/08874417.2006.11645926
15. G. D. S. Hadad, *Uso de Escenarios en la Derivación de Software*, Universidad Nacional de La Plata, Facultad de Ciencias Exactas, Doctoral Thesis, 2008. Available: http://www-di.inf.puc-rio.br/~julio/Tesis-Graciela Hadad.pdf
16. L. Macaulay, “Requirements capture as a cooperative activity”, in *Proc. IEEE 1st International Symposium on Requirement Engineering*, IEEE Computer Society Press, EEUU, pp. 174-181, 1993. Available: http://dx.doi.org/10.1109/ISRE.1993.324820
17. G. D. S. Hadad, G. N. Kaplan, A. Oliveros and J. C. S. P. Leite, “Integración de escenarios con el léxico extendido del

lenguaje en la elicitación de requerimientos: aplicación a un caso real", *Revista de Informática Teórica y Aplicada (RITA)*, Brazil, vol. 6, no. 1, pp. 77-103, 1999.

18. J. C. S. P. Leite, J. H. Doorn, G. N. Kaplan, G. D. S. Hadad and M. N. Ridao, "Defining System Context using Scenarios. Perspectives on Software Requirements", in J. S. C. P. Leite, and J. H. Doorn (Eds.), *Perspectives on Software Requirements*, Norwell, MA: Kluwer Academic Press, pp.169-199, 2004.
19. S. ul Haq, M. Raza, A. Zia and M. N. A. Khan, "Issues in global software development: A critical review", *Journal of Software Engineering and Applications*, vol. 4, no. 10, pp. 590-595, 2011. Available: http://dx.doi.org/10.4236/jsea.2011.410069
20. J. C. Westland, "The cost of errors in software development: evidence from industry", *Journal of Systems and Software*, vol. 62, no. 1, pp.1-9, 2002.
21. M. Dharmadas, *Global Software Development: A Case Study of Knowledge Management Challenges and Industry Approaches*, Norwegian University of Science and Technology, Department of Computer and Information Science, Master Thesis, 2008. Available: http://www.diva-portal.org/smash/get/diva2:348671/FULLTEXT01.pdf
22. IEEE_29148: 2011. *Systems and software engineering — Life cycle processes — Requirements engineering.* Piscataway, NJ, USA: IEEE, 2011.
23. J. Hanisch and B. J. Corbitt, "Requirements Engineering During Global Software Development: Some Impediments to the Requirements Engineering Process-A Case Study", in *Proc. European Conference on Information Systems 2004*, paper 68, 2004.
24. F. Nonyelum Ogwueleka, "Requirement elicitation problems in software development - A case study of a GSM service provider", in *Indian Journal of Innovations & Developments*, Nigeria, vol. 1, no. 8, pp. 599-605, 2012.
25. G. N. Kaplan, J. H. Doorn, N. Gigante, "Evolución semántica de glosarios en los procesos de requisitos", in *Proc. XVIII Congreso Argentino de Ciencias de la Computación*, Mar del Plata, Argentina, 2013. Available: http://hdl.handle.net/10915/32399
26. C. Wang, F. Pastore, A. Goknil, L. Briand and Z. Iqbal, "Automatic generation of system test cases from use case specifications", in: *Proceedings of the 2015 International Symposium on Software Testing and Analysis* (ISSTA 2015), ACM, New York, NY, USA, 385-396, 2015. Available: http://dx.doi.org/10.1145/2771783.2771812
27. J. J. Gutiérrez, M. J. Escalona, M. Mejias, J. Torres and A. H. Centeno, "A case study for generating test cases from use cases", in *Proc. Second International Conference on Research Challenges in Information Science (RCIS 2008)*, IEEE, pp. 209-214, June 2008. Available: http://dx.doi.org/10.1109/RCIS.2008.4632109
28. E. Sarmiento, J. C. S. P. Leite and E. Almentero, "C&L: Generating model based test cases from natural language requirements descriptions", in *Proc. IEEE 1st International Workshop on Requirements Engineering and Testing (RET 2014)*, IEEE, pp. 32-38, 2014. Available: http://dx.doi.org/10.1109/RET.2014.6908677