



Pós-Graduação em Ciência da Computação

**“A RISK IDENTIFICATION TECHNIQUE FOR
REQUIREMENTS ASSESSMENT”**

Por

LILIANE SHEYLA DA SILVA

Dissertação de Mestrado



Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, MARÇO/2012



UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

LILIANE SHEYLA DA SILVA

“A RISK IDENTIFICATION TECHNIQUE FOR REQUIREMENTS ASSESSMENT”

*ESTE TRABALHO FOI APRESENTADO A POS-GRADUAÇÃO EM
CIÊNCIA DA COMPUTAÇÃO DO CENTRO DE INFORMÁTICA DA
UNIVERSIDADE FEDERAL DE PERNAMBUCO COMO REQUISITO
PARCIAL PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIA
DA COMPUTAÇÃO*

ORIENTADOR: SERGIO CASTELO BRANCO SOARES, Phd.
CO-ORIENTADORA: CRISTINE MARTINS GOMES DE GUSMÃO, Phd.

RECIFE, MARÇO/2012

Catálogo na fonte
Bibliotecária Jane Souto Maior, CRB4-571

Silva, Liliane Sheyla da

A risk identification technique for requirements assessment /
Liliane Sheyla da Silva. - Recife: O Autor, 2012.

x, 120 folhas: il., fig., tab.

Orientador: Sergio Castelo Branco Soares.

Dissertação (mestrado) - Universidade Federal de Pernambuco.
CIn, Ciência da Computação, 2012.

Inclui bibliografia, anexo e apêndice.

1. Engenharia de software. 2. Testes de software. 3. Gerência de
riscos. I. Soares, Sergio Castelo Branco (orientador). II. Título.

005.1

CDD (23. ed.)

MEI2012 – 068

Dissertação de Mestrado apresentada por **Liliane Sheyla da Silva** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título “**A RISK IDENTIFICATION TECHNIQUE FOR REQUIREMENTS ASSESSMENT**”, orientada pelo **Prof. Sergio Castelo Branco Soares** e aprovada pela Banca Examinadora formada pelos professores:

Prof. Márcio Lopes Cornélio
Centro de Informática/UFPE

Profa. Caroline Maria de Miranda Mota
Departamento de Engenharia de Produção/UFPE

Prof. Sergio Castelo Branco Soares
Centro de Informática/UFPE

Visto e permitida a impressão.
Recife, 1 de março de 2012

Prof. Nelson Souto Rosa
Coordenador da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

ACKNOWLEDGEMENTS

I thank to God for everything, without Him I would not be able to accomplish this work.

Thanks to my family, for being the foundations of my life

Thanks to my fiancé, for his patience, words and for bringing me dinner when I was working late at CIn.

Thanks to Nielson, for all the support. Without him this work would not have been finished in time.

I thank Professor Sergio Soares, for encouraging me to finish on time, for brilliant practicality, for being the provider of the needs that have arisen and for always being accessible.

Thanks to Professor Cristine Gusmão for the guided learning in the risk management area and for all advices.

Thanks to Professor Aldemar Santos, a teacher of excellence who have crossed my path during the Master, with his teaching and support.

Thanks to Felipe Buarque for all his available, thanks to Emanuel Barreiros who will be an excellent advisor, for the revisions made on this work. Thanks to Adauto Trigueiro for discussions regarding this work.

Thank you to my friends who remain friends even not being able to be with them as much as I wished.

Thanks to Software Engineering Laboratory, which provided all the infrastructure for development this work.

Thank to CAPES and CNPq for financing this research;

To all, thank you for the support and encouragement for this research to become a reality, which have directly and indirectly contributed to complete this work.

RESUMO

Um dos problemas enfrentados no desenvolvimento de software é a criação de casos de testes eficazes a partir dos requisitos. Várias técnicas e métodos são aplicados com a finalidade de minimizar os riscos associados à construção dos casos de testes, objetivando o correto atendimento das necessidades especificadas. A identificação de riscos para avaliação dos requisitos é primordial para a geração de casos de testes. No entanto, os engenheiros de testes, ainda encontram dificuldades em aplicá-la na prática pela ausência de conhecimentos sólidos sobre as atividades da Gerência de Riscos e também pela ausência de ferramentas de apoio para esta atividade. Este trabalho propõe uma técnica que auxilia os engenheiros de testes, na etapa de identificação de riscos nos requisitos para a realização do teste de software. A partir de estudos que utilizaram o conceito de similaridade para comparar projetos de software, a fim de reutilizar riscos identificados anteriormente. A técnica desenvolvida utiliza esse mesmo procedimento aplicado no domínio de requisitos. Dentro deste contexto, este trabalho: i) define uma técnica, com base em analogia, por meio da categorização de requisitos, podendo assim identificar riscos através de uma base de dados de requisitos semelhantes, e ii) reutiliza riscos identificados em requisitos anteriormente para a avaliação de novos requisitos.

Palavras-chave: testes de software baseado em riscos; identificação de riscos, requisitos de software;

ABSTRACT

One recurrent issue in software development is the effective creation of testcases from requirements. Several techniques and methods are applied to minimize the risks associated with test cases building, aiming to meet the requirements specified correctly. Risks identification for requirements assessment is essential to tests cases generation. However, test engineers still face difficulties to apply it in practice due to the lack of solid knowledge about Risk Management activities and tool support for such activities. This work proposes a technique that helps test engineers in risk identification from requirements for software testing. From studies that used the similarity concept to compare software projects in order to reuse previously identified risks, the developed technique uses the same assertion applied to requirements. Within this context, this work aims to: (i) to define a technique based on analogies by categorizing requirements, thus being able to identify risks through a database of similar requirements, and (ii) to reuse risks previously identified at requirements for the evaluation of new requirements.

Keywords: *Risk based Testing; risk identification; software requirements*

CONTENTS

CHAPTER 1 — INTRODUCTION	1
1.1. MOTIVATION.....	3
1.2. GOALS.....	5
1.2.1. GENERAL GOAL	5
1.2.2. SPECIFIC GOALS	6
1.3. CONTRIBUTIONS	6
1.4. METHODOLOGY	6
1.5. OUTLINE.....	7
CHAPTER 2 — TECHNIQUES FOR RISK IDENTIFICATION IN SOFTWARE TESTING: A MAPPING STUDY ...	8
2.1. METHODOLOGY	8
2.1.1. RESEARCH QUESTION	9
2.1.2. RESEARCH STRATEGY.....	9
2.1.3. SCIENTIFIC SOURCES.....	10
2.1.4. STUDY SELECTION.....	10
2.1.5. DATA EXTRACTION AND QUALITY ASSESSMENT.....	11
2.2. RESULTS	12
2.2.1. RESULTS QUALITY OF PRIMARIES STUDIES.....	13
2.2.2. RESULTS DATA EXTRACTION AND QUALITY SCORE	13
2.2.3. ANSWERS TO RESEARCH QUESTION.....	13
2.3. DISCUSSION	14
2.3.1. ANALYSIS OF THE TECHNIQUES FOUND	14
2.3.2. LIMITATIONS	16
2.3.3. LESSONS LEARNED	17
2.4. SMS CONCLUSION.....	17
CHAPTER 3 — A RISK IDENTIFICATION TECHNIQUE FOR REQUIREMENTS ASSESSMENT	19
3.1. DEFINING THE ATTRIBUTES OF SOFTWARE REQUIREMENTS	21
3.2. RISKS SOFTWARE REQUIREMENTS.....	23
3.2.1. ANALYTIC STUDY OF THE ATTRIBUTES OF REQUIREMENT VERSUS THE RISKS.....	25
3.3. SIMILARITY CALCULATION	27
3.3.1. CALCULATION OF THE SEQUENTIAL SIMILARITY	28
3.3.2. EUCLIDEAN DISTANCE SIMILARITY	30
3.4. RECOVERY AND STORAGE OF REQUIREMENTS	30
3.5. RITRA MODELING.....	31
CHAPTER 4 — EVALUATION.....	35
4.1. SIMILARITY ALGORITHMS EVALUATION	35
4.1.1. DEFINITION	35
4.1.2. RESULTS ANALYSIS	39
4.2. EXPERIMENTAL STUDY	42
4.2.1. DEFINITION	42
4.2.1.1. GLOBAL GOAL	42
4.2.1.2. STUDY GOAL	42
4.2.1.3. MEASUREMENT GOAL.....	42
4.2.1.4. METRICS.....	43
4.2.2. PLANNING.....	44
4.2.2.1. SELECTING THE CONTEXT	44

4.2.2.2.	<i>METRICS DEFINITION</i>	44
4.2.2.3.	<i>SELECTION OF THE VARIABLES</i>	45
4.2.2.3.1.	INDEPENDENT VARIABLES	45
4.2.2.3.2.	DEPENDENT VARIABLES	45
4.2.2.4.	<i>SAMPLE SELECTION</i>	45
4.2.2.5.	<i>EXPERIMENT DESIGN</i>	46
4.2.2.5.1.	TREATMENT	47
4.2.2.5.2.	PROJECT	47
4.2.2.6.	<i>INSTRUMENTATION</i>	47
4.2.2.7.	<i>THREATS TO VALIDITY</i>	48
4.2.2.7.1.	INTERNAL VALIDITY	48
4.2.2.7.2.	CONCLUSION VALIDITY	48
4.2.2.7.3.	CONSTRUCT VALIDITY	49
4.2.2.7.4.	EXTERNAL VALIDITY	49
4.2.3.	<i>OPERATION</i>	50
4.2.3.1.	<i>PREPARATION</i>	50
4.2.3.2.	<i>EXECUTION</i>	50
4.2.4.	<i>DATA ANALYSIS</i>	52
4.2.5.	<i>EXPLORATORY DATA ANALYSIS</i>	67
4.2.5.1.	QRI_{TRAT} VERSUS QRI_{STRA}	67
4.2.5.2.	TRI_{TRAT} VERSUS TRI_{STRA}	71
4.2.6.	<i>GENERAL CONSIDERATIONS AND DIFICULTIES</i>	71
CHAPTER 5 — CONCLUSIONS		73
5.1.	RESEARCH CONTRIBUTION	74
5.2.	THREATS TO VALIDITY	75
5.3.	FUTURE WORK	75
REFERENCES		77

LIST OF FIGURES

FIGURE 1.1. LIFE CYCLE OF TESTING PROCESS. ADAPTED BY [QUALITI 2008]	2
FIGURE 3.1. SEI TAXONOMY FOR SOFTWARE REQUIREMENTS [CARR AT AL. 1993]	23
FIGURE 3.2. RITRA LIFE CYCLE	32
FIGURE 3.3. RITRA CLASS DIAGRAM	34
FIGURE 4.1. EXPERIMENT PROCESS	53
FIGURE 4.2. COMPARISON OF IDENTIFIED RISK'S PERCENTAGE INDEXES OF COVERAGE	70
FIGURE 4.3. COMPARISON OF IDENTIFIED RISKS' PERCENTAGE INDEXES OF EFFICACY.	70
FIGURE 4.4. COMPARISON OF TIME TAKEN FOR THE IDENTIFICATION OF RISKS	71

LIST OF TABLES

TABLE 2.1. SELECTION CRITERIA	11
TABLE 2.2. QUALITY CRITERIA.....	11
TABLE 2.3. PRIMARY STUDY SELECTION.....	12
TABLE 2.4. RESULTS DATA EXTRACTION AND QUALITY SCORE	13
TABLE 2.5. TECHNIQUES FOR RISK IDENTIFICATION IN SOFTWARE TESTING	14
TABLE 2.6. BENEFITS AND CONSTRAINTS OF THE TECHNIQUES TO RISK IDENTIFICATION	16
TABLE 3.1. RISKS DEFINITION. [CARR ET AL. 1993].	24
TABLE 3.2. ATTRIBUTES VALUES	27
TABLE 3.3. ACTIVITY 1:REQUIREMENT ATTRIBUTES CLASSIFICATION	31
TABLE 3.4. ACTIVITY 2: RISK IDENTIFICATION.....	32
TABLE 3.5. SUGGESTIONS FOR RITRA FEATURES	33
TABLE 3.6. SUGGESTIONS FOR THE RITRA TECHNIQUE AUTOMATION.....	33
TABLE 4.1. ACTIVITIES A1 AND A2 PROCEDURES.....	37
TABLE 4.2. PERFORMANCE ANALYSIS OF SIMILARITY ALGORITHMS	38
TABLE 4.3. ACTIVITIES B1 AND B2 PROCEDURES.....	39
TABLE 4.4. RESULT OF THE TECHNIQUE APPLICATION	40
TABLE 4.5. BEHAVIOR OF ALGORITHMS IN THE SELECTION OF SIMILAR REQUIREMENTS.....	41
TABLE 4.6. YEARS EXPERIENCE	46
TABLE 4.7. EXPERIMENT SCOPE.....	46
TABLE 4.8. APPLICATION OF THE TREATMENTS TO THE SAMPLES.....	47
TABLE 4.9. DESCRIPTION OF SCENARIO A	51
TABLE 4.10. DESCRIPTION OF SCENARIO B.....	51
TABLE 4.11. PROFILE'S EXPERIMENT PARTICIPANTS	52
TABLE 4.12. RISKS IDENTIFIED BY RISK EXPERT 1 USING THE RITRA TECHNIQUE	54
TABLE 4.13. RISKS IDENTIFIED BY RISK EXPERT 2 USING THE RITRA TECHNIQUE	55
TABLE 4.14. RISKS IDENTIFIED BY RISK EXPERT 3 USING THE RITRA TECHNIQUE	55
TABLE 4.15. TIME TO IDENTIFY RISKS USING THE RITRA TECHNIQUE BY RISK EXPERTS.....	56
TABLE 4.16. MERGE OF RESULTS OF THE RISK EXPERT USING THE RITRA TECHNIQUE	56
TABLE 4.17. RISKS IDENTIFIED AD HOC BY RISK EXPERT 1	57
TABLE 4.18. RISKS IDENTIFIED AD HOC BY RISK EXPERT 2	57
TABLE 4.19. RISKS IDENTIFIED AD HOC BY RISK EXPERT 3	58
TABLE 4.20. TIME TO IDENTIFY RISKS AD HOC BY RISK EXPERTS.....	58
TABLE 4.21. MERGE OF THE RISK IDENTIFICATION AD HOC	59
TABLE 4.22. INTERSECTION OF THE RISKS IDENTIFIED BY RISK EXPERT (RIE)	60
TABLE 4.23. RISKS IDENTIFIED BY TEST ENGINEER 1 USING THE RITRA TECHNIQUE	61
TABLE 4.24. RISKS IDENTIFIED BY TEST ENGINEER 2 USING THE RITRA TECHNIQUE	62
TABLE 4.25. RISKS IDENTIFIED BY TEST ENGINEER 3 USING THE RITRA TECHNIQUE	63
TABLE 4.26. RISKS IDENTIFIED BY TEST ENGINEER 4 USING THE RITRA TECHNIQUE	64
TABLE 4.27. TIME TO IDENTIFY RISKS USING THE RITRA TECHNIQUE BY TEST ENGINEERS.....	64
TABLE 4.28. RISKS IDENTIFIED AD HOC BY TEST ENGINEER 1	65
TABLE 4.29. RISKS IDENTIFIED AD HOC BY TEST ENGINEER 2	65
TABLE 4.30. RISKS IDENTIFIED AD HOC BY TEST ENGINEER 3	65
TABLE 4.31. RISKS IDENTIFIED AD HOC BY TEST ENGINEER 4	65
TABLE 4.32. TIME TO IDENTIFY RISKS AD HOC BY TEST ENGINEERS.....	66
TABLE 4.33. NUMBER OF CONSISTENT IDENTIFIED RISKS USING THE RITRA TECHNIQUE.....	66
TABLE 4.34. NUMBER OF CONSISTENT IDENTIFIED RISKS WITHOUT RITRA.....	66
TABLE 4.35. IDENTIFIED RISKS' PERCENTAGE INDEXES OF COVERAGE AND EFFICACY APPLYING RITRA.....	68
TABLE 4.36. IDENTIFIED RISKS' PERCENTAGE INDEXES OF COVERAGE AND EFFICACY AD HOC	68
TABLE 4.37. DESCRIPTIVE DATA OF COVERAGE INDEXES USING THE RITRA TECHNIQUE.....	69
TABLE 4.38. DESCRIPTIVE DATA OF EFFICACY INDEXES USING THE RITRA TECHNIQUE.....	69
TABLE 4.39. DESCRIPTIVE DATA OF COVERAGE INDEXES AD HOC.....	69
TABLE 4.40. DESCRIPTIVE DATA OF EFFICACY INDEXES AD HOC.....	69
TABLE 4.41. TIME DESCRIPTIVE DATA FOR THE RISK IDENTIFICATION	71

LIST OF EQUATIONS

EQUATION 3.1. 29

EQUATION 3.2. 29

EQUATION 3.3. 30

EQUATION 4.1. 38

EQUATION 4.2. 67

EQUATION 4.3. 68

CHAPTER 1

INTRODUCTION

Organizational environments have a constant concern about their products and services quality. Thus testing activities had assumed an essential role in Software Quality Assurance – SQA, ensuring that requirements meet stakeholders needs [Myers 2004]. In contrast, to carefully evaluate and guarantee software quality, it is necessary that the set of tests designed to ensure the most critical areas of the system has been functioning correctly. According to Pressman [Pressman 2009] testing activities require a reasonable effort. There is an enormous diversity of system' inputs and outputs, which can impact cost as much as 40% of the software development (in its first version).

Companies live with the dilemma of developing affordable products not jeopardizing quality. Testing processes are solutions which can be employed to enhance quality throughout the software development. Tests allow assessing whether the software developed meets the specified requirements for further development, significantly reducing costs after deployment and during system maintenance.

Risk management in software development projects, specifically technical risks, assume joint responsibility along with testing activities. The main advantages of risk management activities included in software testing are to minimize the likelihood of potential problems during development, so that objectives are achieved with less time and cost, without sacrificing product's quality. The Risk-based Testing – RBT [Bach 1995] allows prioritizing efforts and allocating resources for software components that need to be tested more thoroughly. The approach identifies, analyses and controls risks associated with software requirements, reducing time and effort for testing a product [Souza et al. 2008].

A testing process defines activities, practices, and artifacts used to test a software. Whereas the quality of the testing process is directly related to the final quality of the product developed, the testing process aims to define how methods are planned, designed, executed and controlled. Phases of

testing life cycle are directly linked with the software development life cycle see Figure 1.1. Testing activities begin at the same time development activities are initiated. The testing life cycle is iterative, resulting in quality increasing and risk failure decreasing of the final product. The initial testing planning occurs during the planning activity, where the requirements are captured and tests are planned. Then, analysis and design activities focus in test design. At this stage the cases and tests procedures are designed to validate and establish a stable architecture for the system. At the stage of development /implementation, the main focus of activities is the implementation and tests execution of the various requirements implemented. In order to deploy the system, acceptance testing of the product is performed. Acceptance tests are classified as "black box" tests, made by the user in order to demonstrate compliance with the software requirements so that finally there is an assessment of the correctness of the changes made due to defects found.

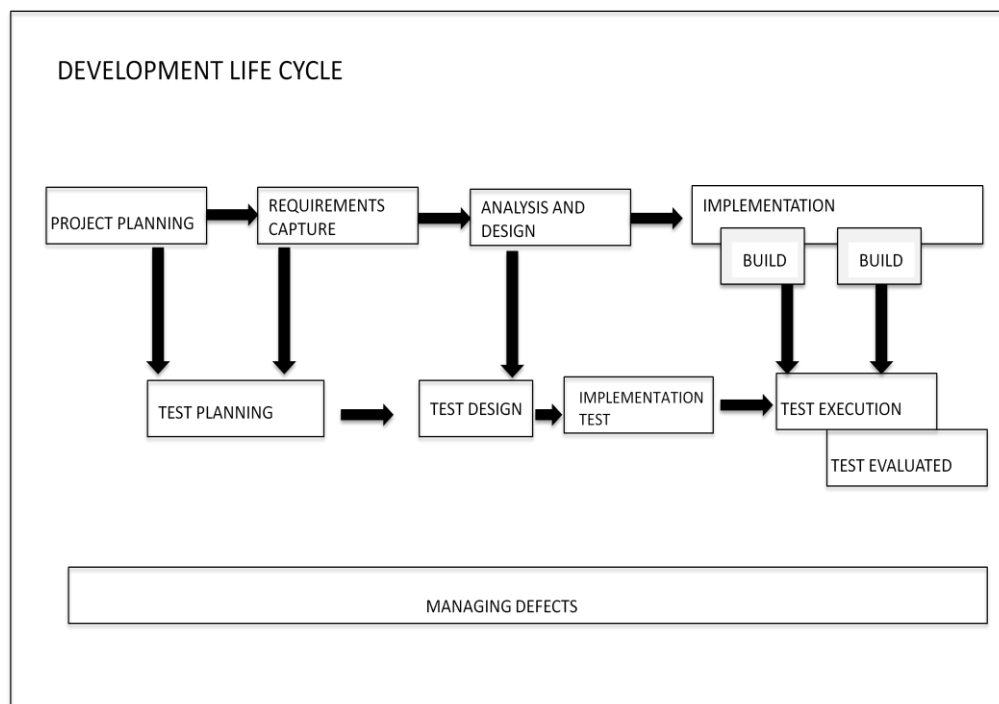


Figure 1.1. Life Cycle of Testing Process. Adapted by [Qualiti 2008]

Software testing cannot guarantee a bug free application, but it can test most of the functionality in minimum space of time. The solution is directing efforts to create tests that identify the system's most critical problems. The

software testing life cycle begins with planning alongside software requirements. The software test-planning activity is characterized by the definition of the entire testing process. This is the main activity, because it defines all testing activities for each iteration such as: requirements to test, priorities, resources and development schedule, in order to direct and control testing activities. If the risks in software requirements can be identified in the planning activity, analysts can create test cases from requirements that presented most critical risks. Such test cases will be more likely to identify problems than planned tests without previous criterion. Each requirement of a system is associated with a set of risks. However, test engineers do not have knowledge in risks. This work presents a technique that helps test engineers in risk identification activities from requirements.

The aim of this introductory chapter is to present the importance of this subject and to show concepts covered in this work: Section 1.1 presents motivation of this work, Section 1.2 presents main goals , Section 1.3 describes contributions and Section 1.4 presents methodology adopted during the construction process.

1.1. MOTIVATION

According to Goldsmith [Goldsmith 2006], unfortunately, the testing process is seen by many as a process implemented in the final activity of product development. The software testing activity is neglected due to the lack of time to test the system. For many in the software engineering field, it is common to hear "everything is ready, we just need to test". However, how can one be sure that everything is ready if there are still tests to be performed? Testing software does not ensure that the product is free of flaws, but at least minimizes the potential faults of the developed product by increasing customer satisfaction.

Dustin [Dustin et al. 1999] mention relevant facts about the time spent on development. According to them, more than 90% of developers have missed shipping dates. Missing deadlines is a routine occurrence for 67% of developers. In addition, 91% have been forced to remove key functionalities late in the development cycle to comply with deadlines.

Since the software testing activity is costly [Myers 2004], it is essential to use a testing process that would increase product's quality and reduce both development time and cost. Due to the limited resources available and the high cost to perform the testing activities, the testing process needs to prioritize efforts for implementing test cases.

Risks identification is an activity that can be used to assist tests cases creation, especially those cases that are automatically generated. The success to detect critical use cases requires good techniques for risks identification, enabling a test engineer to prioritize particular test cases during the planning activity of software testing. In general, test cases are built from software requirements. If the requirements' risks are already known, test engineers could build test cases based on more critical requirements' risks, facilitating the identification of failures in the products and minimizing efforts. However, test engineers still find difficulties in applying the technique in practice, by lacking solid knowledge about risk management activities, and by the lack of tool support.

Methodologies and, in some cases, specific software testing approaches, such as those based on risks, are needed to ensure greater coverage, analysis and test planning. These methodologies allow greater efficacy and better quality.

If there is a database with requirements' risks previously identified, a risk identification technique based on an analogy concept could be developed to aid test engineers. An "analogy" is a comparison of two things that are essentially dissimilar but are shown through the analogy to have some similarity [Higgins 1994]. Analogy techniques are simple and based upon the concept that similar projects have many similarities [Newell et al. 2002]. This concept is extended to requirements in this work. From previous history of others similar requirements, an analogy can be built. Three types of data are required to use this technique: 1) Description and characteristics of the new requirement. 2) Description and characteristics of the past requirements. 3) Identified risks of the past requirements.

In 2007, a method for risks identification called CBR Risk was purposed and developed by Lins [Lins 2007] aiming to help the project risk identification. This method has the assumption that "similar software projects

have similar risks”. The method is based on the artificial intelligence technique named Case-Based Reasoning (CBR), which uses information about previous experiences to resolve new problems. For a new project, CBR Risk tries to find similar projects in a database from which the risks can be adapted in a new project. After its development, the model proposed by Lins [Lins 2007] was evaluated and evolved by Trigo [Trigo et al. 2007] through experimental studies, providing improvements and progress in the used concepts. In the context in this work, it was realized that the fundamental assumption was positively evaluated. We aim to verify if this assumption works in the domain of requirements, i.e., “similar requirements have similar risks”.

This work proposes a technique for risks identification. The technique uses the similarity concept through similarity algorithms to compare similarities among attributes from requirements, taking into account that similar requirements present similar risks. This work intends to provide a technique that allows test engineers to (i) compare one requirement to another, and thus identify risks through a database of similar requirements, and (ii) use the risks identified in previous requirements to identify risks in new requirements.

1.2. GOALS

Following sections presents main general and specific goals of this work:

1.2.1. General goal

The purpose of this work is to define a technique, which helps test engineers in the risk identification activity on software testing requirements. To achieve the overall goal of this work, a set of actions were divided into four steps

- 1) General review on software testing and risk identification ;
- 2) Investigation and study of the main techniques for risks identification;
- 3) Designing and proposing a technique for risks identification;
- 4) Application of the proposed technique.

For more details about methodology see Section 1.4.

1.2.2. Specific goals

For this objective to be achieved some targets had to be identified and planned, enabling the efficient execution of all defined activities. These goals are part of the specific objectives and are detailed in the next section. The specific objectives of this study are:

- To propose modeling of the technique,
- To propose a set of attributes for categorizing requirements,
- To evaluate and analyze the similarity algorithms.

1.3. CONTRIBUTIONS

The main contributions of this work are the following:

- 1- Studying and using the concept of similarity to compare requirements;
- 2- Proposing a set of simple attributes for categorizing requirements;
- 3- Defining a technique to help test engineers in risk identification;
- 4- Performing a mapping study to identify which risk identification techniques are used in software testing;
- 5- Evaluating the proposed technique in a systematic experiment.

1.4. METHODOLOGY

The methodology applied for this work is presented in four phases reported below:

Phase 1: General review

- Literature review and study of techniques and methodologies on the planning area of software tests;
- Review about Risk-based testing, analyze their activities and approaches.

Phase 2: Investigation and study of the main techniques for risks identification

- Studying techniques for risks identification.

Phase 3: Designing and proposing a technique for risks identification

- Selecting a standard for modeling and defining the technique;
- Study of the selected modeling tool;
- Proposal of a technique for risk identification by describing lifecycle;
- Establishing and specifying the assessment requirements of the technique.

Phase 4: Technique application proposed in the planning phase of testing

- Technique evaluation;
- Data collection, analysis and presentation of the assessment.

1.5. OUTLINE

The remainder of this dissertation is organized as follows:

- Chapter 2 describes a mapping study to identify in the literature the existing techniques for identifying risks in the area of software testing. The purpose is to identify and to compare the used approaches
- Chapter 3 proposes a technique for risk identification in the planning phase of testing.
- Chapter 4 presents the evaluation, analysis and discussion about the proposed technique.
- Chapter 5 summarizes the contributions of this research, discusses some related and future work, and presents the conclusions.
- Appendix A presents artifacts developed and used in this work.
- Appendix B presents results of evaluations of this work
- Annex A presents the requirements (versions 1.3 and 1.5, respectively) used in evaluations of this work.
- Annex B presents a Taxonomy-Base Risk Identification Questionnaire for requirements [Carr et al. 1993].

CHAPTER 2

TECHNIQUES FOR RISK IDENTIFICATION IN SOFTWARE TESTING: A MAPPING STUDY

Risk Identification is a discipline of risk management covered by several models and guidelines, including PMBOK Guide (Project Management Body of Knowledge) [PMI 2004], RUP [Jacobson et al. 1998][Kruchten 1998] and CMMI [SEI 1993] level 3 ISO 31000[ISO 31000 2009]. Such task can be inserted into the testing process of the planning phase to identify key features and system's more critical flows to be tested based on their priority. Therefore, it is necessary to know and use methodologies, tools, processes in the identification of risks, ensuring greater productivity, efficiency, and better quality. With this purpose in mind, we performed a systematic mapping study (SMS) [Kitchenham et al. 2007] to collect evidence on risk identification research, following specific guidelines [Kitchenham et al. 2007]. The SMS described in this chapter aimed to identify, in the literature, which are the techniques for identifying risks in the area of software testing.

The rest of the chapter is organized as follows. Section 2.1 describes the research methodology employed. Section 2.2 presents the results and data analysis. Section 2.3 presents closing discussions describing the techniques found, limitations, lessons learned. Conclusions from the mapping are presented in Section 2.4.

2.1. METHODOLOGY

The Systematic Mapping Study (SMS) conducted in this work had the purpose of identifying, in the scientific literature of conferences, and journals, among others, which are the techniques for identifying risks in the area of software testing. The purpose is to identify the approaches used. The main step to perform a SMS is the protocol definition, which must describe the research plan in details [Kitchenham et al. 2007]. The research protocol for the SMS described in this paper is summarized in the following sections.

2.1.1. Research Question

The first step in performing such study is formulating a primary research question as part of the research protocol. The aim of this SMS is to find evidences in the literature to answer the following question:

Which techniques are used to identify risks that guide in software testing?

2.1.2. Research Strategy

The strategy used to construct the search string occurred in the following steps:

- a) Keywords identification for the research question;
- b) Synonyms identification for each term;
- c) Use boolean term “OR” to incorporate the synonyms;
- d) Use boolean term “AND” to link keywords identified in Step a).

Results for Step a): The initial phase of construction of the search string identified the keywords related to the research question. We used three keywords: “techniques”, “risk identification”, and “software testing”;

Results for Step b): The second phase identified, if necessary, the synonyms for each keyword identified in the first phase. “Method” was considered synonym to “technique” because, according Pressman [Pressman 2009], a method is set of techniques. “Test case” and “testing process” were also identified as synonyms to Software Testing. An experienced researcher helped to define such synonyms;

Results for steps c) and d): In the last stages of building the search string, the synonyms for each keyword were aggregated using the boolean term “OR”. The keywords were put together using the boolean term “AND”. Quotation marks (") were used while addressing composed words at the search string to ensure the words were located in the same way and in the same order they were defined.

The final step to define the search string was discussed, reviewed, and approved by expert researchers in the management and identification risk area, considering the construction and the most appropriate synonyms. The

search string was constructed in a restricted way to select only studies that directly answer the research question. Thus, the search string for the research question is:

(Technique OR Method) AND ("Risk Identification") AND
("Software Testing" OR "Test Case" OR "Testing process")

2.1.3. Scientific Sources

The automatic search was performed in four search engines:

- 1) SCOPUS: www.scopus.com;
- 2) SCIENCE DIRECT: www.sciencedirect.com;
- 3) ACM Digital Library: dl.acm.org;
- 4) IEEEExplorer: ieeexplore.ieee.org.

Current automatic mechanisms are not efficient, stable and reliable. So manual search is a way to complement the automatic search. A manual search is performed by inserting the string “risk identification” in the search field. it was performed in the following scientific journals related to software risks and software engineering, respectively:

- 1) Risk Analysis: [onlinelibrary.wiley.com/journal/10.1111/\(ISSN\)1539-6924](http://onlinelibrary.wiley.com/journal/10.1111/(ISSN)1539-6924)
- 2) IEEE Transactions on Software Engineering: www.computer.org/portal/web/csd/transactions/tse

The queries were constrained in both journals to studies published between 2000 and 2010 to obtain recent studies.

2.1.4. Study Selection

Four software engineering MSc students were divided in two pairs. Search sources were equally divided to each pair. The first selection was made based on titles, keywords and abstract, using the study selection criteria detailed in Table 2.1. The lists were merged - one from each pair that performed in the first selection. The second selection was made based on reading introduction and conclusion of all articles obtained in the first selection. Repeated papers

were removed and conflicting ones were resolved in face-to-face meetings. Again, the lists were merged into a unique list with potentially relevant papers. All steps of the selection process were recorded in spreadsheets.

Table 2.1. Selection Criteria

Exclusion criteria	Inclusion criteria
Does not answer research question	Answers research question
Not relevant (title, summary and keywords that are not related to the focus of a SMS)	Empirical studies (quantitative and qualitative) about risks identification
Not Available/accessible	Studies that include techniques for risks identification
Duplicated	
Not written in English	
Focus on identifying risks in a different context than in software testing	

2.1.5. Data Extraction and Quality Assessment

After primary study selection, data extraction and quality assessment was applied to the potentially relevant papers. Data extraction select the following information: (1) Authors; (2) Place of execution; (3) Place of publication; and (4) Techniques described. The criteria and quality metrics used in this SMS were based on references [Kitchenham et al. 2007] [Dybå 2008]. According to them, seven quality criteria were defined (Table 2.2) and three evaluation levels: Agree, Partially Agree and Disagree. Each level has a score (weight) associated: 1, 0.5, and 0, respectively. At the end, studies will obtain a Final Score (weights average) ranging from 0 to 1 for quality classifications: Poor: 0,0 to 0,24; Regular: 0,25 to 0,49; Good: 0,50 to 0,74; Very Good: 0,75 to 1,0

Table 2.2. Quality Criteria

1	Are the aims clearly stated?
2	Are the variables used in the study adequately measured?
3	Were the basic data adequately described?
4	Is the purpose of the analysis clear?
5	Are the findings explicitly stated?
6	Are the conclusions useful for the academic community or to the software industry?
7	Are threats to validity of the study presented?

2.2. RESULTS

We obtained 333 studies from the automatic and manual searches.

Primary study selection was made in two phases:

1. In this step, only papers that were clearly out of scope were removed. This was achieved by analyzing only title, abstract and keywords. The main goal of this step is to keep only potentially relevant studies;
2. In this step, the full papers were obtained and screened. As a result, only papers that were identified as relevant to the mapping were kept. Observe that only three papers remained after this phase. Table 2.3. Primary Study Selection depicts these numbers through the selection steps.

Table 2.3. Primary Study Selection

Primary Study Selection						
Resources	Source Results	1 st Phase (Title, abstract and keywords)	2 nd Phase (Introduction and conclusion)			
		Potentially Relevant Studies	Deleted			Selected
			Not relevant	Repeated	Unavailable	
IEEE Xplore	3	2	1	0	0	1
ScienceDirect	32	9	9	0	0	0
Scopus	6	3	1	0	0	2
ACM	8	0	8	0	0	0
Manual Search	284	14	13	1	0	0
TOTAL	333	28	32	1	0	3

The search results show that there are few techniques reported in the literature to identify risks in software testing.

Table 2.4. Results Data Extraction and Quality Score

Study	Identifying and managing risks for automatic test systems [Calhoun et al. 2000]	Risk based testing: A case study [Souza et al. 2010]	Measurement and control for risk- based test cases and activities [Souza et al. 2009]
Author	Cynthia C. Calhoun	Ellen Souza, Cristine Gusmão, Júlio Venâncio	Ellen Souza, Cristine Gusmão, Keldjan Alves, Júlio Venâncio, Renata Melo
Execution Place	USA	Brasil	Brasil
Publication place	IEEE AES Systems Magazine	7th International Conference on Information Technology	10th IEEE Latin American Test Workshop
Publication date	2000	2010	2009
Technique description	Brainstorming, periodic risk reporting, project profile questions, taxonomy-based questionnaire and voluntary risk reporting	Taxonomy-based questionnaire, checklist and brainstorming	Taxonomy-based questionnaire and brainstorming
Quality Score	0,57	1	0,93

2.2.1. Results Quality of primaries Studies

A team of four evaluators was divided into pairs, who performed the quality evaluation of the three primary studies. Each member of the pair assessed the articles individually to avoid bias in the result. In the case of disagreements in the pair assessment, a third expert in the field area steps in to make a consensus.

2.2.2. Results Data Extraction and Quality Score

Table 2.4 shows the data extracted from the primary studies and the final classification of the selected articles, according to calculation of the quality score defined in Section 2.1.5. It is important to notice there is one good article and two classified like very good. It means that the found evidence have a reasonable level of reliability.

2.2.3. Answers to Research Question

This work presents the results of a systematic literature review in order to list techniques for risk identification that guided in software testing. Table 2.5

shows these techniques reported in the primary studies providing answers to the Request Question.

Table 2.5. Techniques for Risk Identification in Software Testing

Techniques	Studies
Periodic risk reporting	[Calhoun et al. 2000]
Project Profile questions	[Calhoun et al. 2000]
Voluntary risk reporting	[Calhoun et al 2000]
Taxonomy based questionnaire (TBQ)	[Calhoun et al 2000];[Souza et al. 2010];[Souza et al. 2009]
Brainstorming	[Calhoun et al 2000];[Souza et al. 2010];[Souza et al. 2009]
Checklist	[Souza et al. 2010]

2.3. DISCUSSION

This Section presents discussion about the results found. Section 2.3.1 presents an analysis of the techniques found, Section 2.3.2 describes limitations of this study, Section 2.3.3 presents lessons learned and, Section 2.3.4 presents conclusions taken from the SMS.

2.3.1. Analysis of the techniques found

Six techniques to identify risks in software testing were identified from the primary studies (See Table 2.5). These techniques are:

- Brainstorm: a meeting aiming to identify risks through a discussion among participants;
- Taxonomy based questionnaire (TBQ): consists in a set of questions when asked denote if risks exist or not. These questions are based on a risk taxonomy defined by the SEI [Carr et al. 1993];
- Checklist: a list of risks that were previously identified in other projects;
- Periodic Risk Reporting: integrates directly into project activities by requiring each individuals or selected key individuals to periodically submit (mandatory and scheduled) risk forms or a status report that addresses any risks they have identified [Dorofee et al. 1996];

- Project Profile Question: a description of how to tailor the taxonomy-based questionnaire based on project characteristics, eliminating the questions that aren't consistent with the project context. The answers to the project profile questions help the teams conducting interviews by giving them a better understanding of the project [Dorofee et al 1996];
- Voluntary Risk Reporting: is the systematic distribution and regular submission of risks forms as part of routine projects activities [Dorofee et al. 1996].

All these techniques were analyzed to understand their benefits and constraints (see Table 2.6).

From the analysis of the studies, it was observed that brainstorming and TBQ were the most used techniques to identify risks in software testing. All relevant articles cited these techniques as useful for this task. One possible explanation for this is that these techniques are quite widespread and used as references in the literature.

Table 2.6. Benefits and Constraints of the Techniques to Risk Identification

Technique	Benefits	Constraints
Brainstorming	<ul style="list-style-type: none"> •Generates a lot of ideas in a short amount of time; •It does not require training of participants. 	<ul style="list-style-type: none"> •Best in a small group (fewer than nine people [Lumsdain et al. 1990]); •Required a skilled facilitator to deal with the conflict and negative emotion that may surface and must be controlled.
Taxonomy based questionnaire (TBQ)	<ul style="list-style-type: none"> •Use a questionnaire based in risk taxonomy to identify the risks; •Guides the team for risk taxonomy and provides knowledge for the team about the project weaknesses. 	It contains questions that may not be relevant for specific software domains or for specific project organizations.
Checklist	<ul style="list-style-type: none"> •Set of commons risks in several projects; •Set of risks that occurred previously on the project. 	Must have a previous project or managers who have already had experience with risk identification.
Periodic Risk Reporting	<ul style="list-style-type: none"> •Continuous risk identification; •Foster risk awareness throughout the project; •Easily integrated into routing project. •Uses a form to identify risks, enables defining the risk impact and the occurrence probability. 	This technique will not be effective into a routine project that does not have open communication or where there is a little trust or rapport between managers and other personnel [Dorofee et al. 1996]. In this case, the Voluntary Risk Reporting can be used.
Project Profile Question	<ul style="list-style-type: none"> •Help the team to understand the project; •Questionnaires are used to identify risks; 	Use the TBQ as a base to identify risks eliminating some questions that cannot be relevant for the project organization. This elimination should be done carefully to not eliminate the questions incorrectly, turning the questionnaire illegible.
Voluntary Risk Reporting	<ul style="list-style-type: none"> •Continuous risk identification; •Enables everyone in the project to contribute to the risk identification process; •Available to all personnel; •Uses a form to identify risks and enables to define the risk impact and the occurrence probability. 	A central repository or collection person is required to collect and process the risks.

2.3.2. Limitations

This SMS found the risk identification techniques being used in the context of software testing. There is a large set of techniques to identify risks, but most of them are used in other study areas.

Although this study has followed rigorous methodological guidelines [Kitchenham et al. 2007], the generalization of the results is a threat to validity. There might be other techniques that were not identified in this work due to the automatic search that was conducted only in four search engines and, the manual search that was performed in only two journals. To eliminate bias in assessing the studies quality, when two appraisers disagreed at any quality criteria, a third appraiser eliminated the disagreement.

The number of studies returned was considered small. The string can be improved in order to identify more techniques. It is important to note, however, that even if the string is modified, there can be little change in the results regarding the number and relevance of papers returned if the literature really lacks such techniques. This cannot be disregarded as a limitation, though.

2.3.3. Lessons Learned

During the execution of this SMS it was possible to note the benefits and constraints of the risks identification techniques and how they are not widely used in software testing. A small number of studies have been returned as relevant, indicating that research in this area is still limited and there is a gap between risks and testing. This opens a door for new research, such as, new techniques to identify risks in software testing and processes to plan and execute tests to mitigate risks.

2.4. SMS CONCLUSION

The results of this SMS support the conclusion that current techniques for risk identification in software testing are poorly explored. Thus, there are gaps in this area of research towards the proposition of new techniques. On the other hand, in the literature of software engineering [PMI 2004], there are various methods and techniques for identifying risks applied in Project Management such as; brainstorming, checklist, comparison by analogy, analysis of premises, decomposition, diagramming techniques, Delphi technique, revision

of documents and interviews. From this gap, there is a lack of technique in risk-based testing.

CHAPTER 3

A RISK IDENTIFICATION TECHNIQUE FOR REQUIREMENTS ASSESSMENT

This chapter presents the Risk Identification Technique for Requirements Assessment (RITRA). This technique aims to contribute to the identification of risks in requirements based on previously identified risks, in other words, based on past experiences. In this way, it can contribute to a better use of risk identification efforts directing them to identify new risks. To better define the scope of the work being presented, (it is defined Technique (Greek τέχνη- techne) 'art, technical, craft' as a procedure, or all of these, in order to obtain a certain outcome [Houaiss 2001]). Our technique is presented through a life cycle model that indicates the procedures for its implementation. Our technique is inspired from studies Lins and Trigo [Lins 2007; Trigo 2007], however our domain is risk identification on software requirements. This technique is based on SEI risk taxonomy that belongs to the class of product engineering, which is found in the named requirements element. So, the risks identified in the requirements will serve as basis for generating test cases thus enabling the reduction and/or the removal of risks. The RITRA can be used alone, however it will only identify known risks in the database. Its use along with other risk identification techniques makes the identification process complete, as it does not have to spend time identifying previous risks.

The RITRA is presented graphically using the BPM (Business Process Modelling). BPM is a holistic management approach [Brocke et al. 2010] focused on aligning all aspects of an organization with the wants and needs of clients. It promotes business effectiveness and efficiency while striving for innovation, flexibility, and integration with technology. BPM attempts to improve processes continuously. It can therefore be described as a “process optimization process” It is argued that BPM enables organizations to be more efficient, more effective and more capable of change than a functionally focused, traditional hierarchical management approach.

According to Kohlbacher an empirical study indicates that BPM helps organizations to gain higher customer satisfaction, product quality, delivery

speed and time-to-market speed [Kohlbacher 2009]. An empirical study by Vera & Kuntz conducted in the German hospital sector indicates that BPM has a positive impact on organizational efficiency [Vera et al. 2007].

After some research on free BPM Standalone (Installable) tools, BizAgi Free BPMN Process Modeler [BizAgi 2011] was chosen because is the most friendly tool to diagram and document business processes using BPMN notation [Grosskopf et al. 2009]. It allows creating BPMN 1.1 diagrams, sharing them (XPDL and Visio import & export), and publishing documentation.

In this chapter, the details of the proposed modeling technique are presented. The attributes and representation of its values are defined in Section 3.1. The risks of requirements used in this technique and relationship between the risks and requirements attributes are presented in Section 3.2. The calculation of similarity used is presented in Section 3.3. Section 3.4 describes how the requirements are retrieved and stored. Finally, the modeling technique is presented in Section 3.5.

3.1. DEFINING THE ATTRIBUTES OF SOFTWARE REQUIREMENTS

This section presents the attributes of each requirement in the proposed technique used in identifying indices to be used in the calculation of similarity between requirements. The attributes described only the features that will define the requirements. However, before presenting the attributes that represent requirements, an overview of requirements is presented, with its types and finally the attached attributes.

In the literature, several definitions and concepts are found for requirements. The IEEE Standard Glossary of Software Engineering Terminology [IEEE 1990], defines requirement as a “condition or capability needed for a user to solve a problem or to achieve a goal (...)”, that is, software requirement is the property that the software possess to solve a problem.. According to Aurelius, requirement is (“i) Condition that must be met to achieve a certain end; (ii) Requirement of a legal order for that particular process to have progress”. "Requirement is something (verifiable) that the product should do or some quality (measurable) that it should present, and that (by its risk of compromising the success of the project worthiness) should be tested" [Nannini 2010]. High quality requirements are clear, complete, unambiguous, and implementable, consistent and testable [Wiegiers 2003]. Requirements may be functional or nonfunctional [Sommerville 2004]. Functional requirements are descriptions of the various operations that customers and users want or need to be performed by the system. In other words, they are connected directly to the features of a system. For example: activities related to registrations (Insert, Delete, Edit, Query), management reporting, among others. Non-functional requirements define the activities of control and quality attributes of a system. The non-functional requirements describe the system's features. For example: System response time, storage capacity. Non-functional requirements are critical to the success of software systems because they are directly related to user's satisfaction. The quality of the software requirement is the first step towards software quality [Fabbrini et al. 2001]. Both requirements, functional and non-functional have associated attributes that measure the quality requirement. The attributes are important

information on the status of requirements. Just as each person has personal attributes such as: date of birth, gender, color of eyes, among others. Each requirement also presents a set of attributes associated to it. Thus, similarity, if we can run queries through the attributes of a person in a database to find, for example; brown-eyed women and younger than 24 years, in the same way, we can consult the requirements of a software across the attributes to find the similar requirements that are stored in the database. The requirements may have different aspects of quality such as completeness, consistency, clarity, entirety, etc. [Mora et al. 2003]. Various types of attributes used in the requirements are found in the literature [Fabbrini 2001, Fantechi et al. 2003, Firesmith 2003, IEEE 1998, Telelogic 2006, Wilson et al. 1997]. Among them, five attributes were chosen for characterization in this work and the requirements are described below:

- Correctness/Accuracy: Ability of the requirement to be exact concerning the real intentions of customers.
- Ambiguity: Indicates whether the requirement has more than one interpretation.
- Completeness: Indicates the ability of the requirement contain all information necessary for development.
- Stability: This parameter is used to define priorities and which items should be postponed to be explored and analyzed in the next iterations. The higher the stability the less the probability of changing in the requirement.
- Difficulty: Indicates the estimated effort to implement and validate the requirement since some requirements require more time and resources than others to be captured and implemented.

The attributes can be described by numbers, dates, boolean algebra or plain text. Each attribute is represented by ordered symbols. The advantage of using the symbols sorted in this technique is that the attributes possess a default order, so that values more similar are closer. Such as the accuracy attribute can take values: Very High, High, Medium, Low, Very Low, in this case the values have an order of precedence that indicates that the symbol "High" is more like a symbol "Very High" than symbol "Very Low". The

similarity is calculated based on the distance between them using a numerical measure. By using ordered symbols, we can assign numerical values to them and thus use the same measures used in the numeric types, Example: Very High-> 1, High-> 2, Medium-> 3, Low-> 4, Very Low-> 5, in this case, values ranging from one (1) and five (5) will be used. The similarity can be found through a simple calculation that will be presented in Section 3.3.

3.2. RISKS SOFTWARE REQUIREMENTS

The classification used in this work is defined by the Software Engineering Institute (SEI) [Carr et al. 1993], which provides a taxonomy for software risks. The SEI taxonomy organizes software risks in three classes: Product engineering, Development environment and Program constraints. These classes are subdivided into elements, which in turn are divided into attributes, according to **Erro! Fonte de referência não encontrada.**

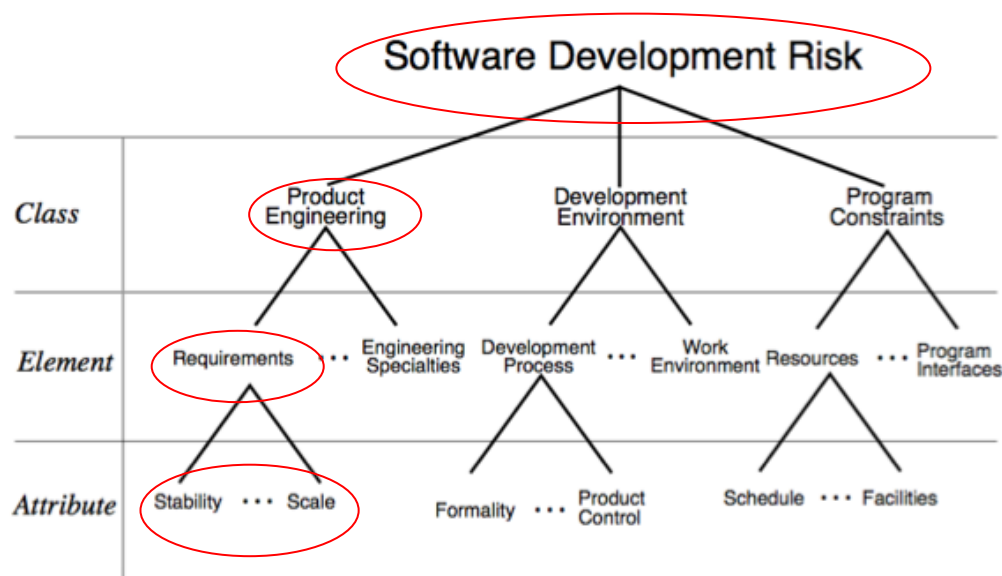


Figure 3.1. SEI taxonomy for software requirements [Carr et al. 1993]

As the scope of this work has to classify risks on requirements, also known as technical risks, this work is limited to use only the risk rating of the Product Engineering class. In the Product Engineering class, we have the element requirements. The following attributes characterize the requirements element: Stability, Completeness, Clarity, Validity, Feasibility, precedent and

Scale. The definition of each attribute is presented in the Table 3.1 extracted from Taxonomy-Based Risk Identification [Carr et al. 1993].

Table 3.1. Risks Definition. [Carr et al. 1993].

Stability
The stability attribute refers to the degree to which the requirements are changing and the possible effect changing requirements and external interfaces will have on the quality, functionality, schedule, design, integration, and testing of the product being built. The attribute also includes issues that arise from the inability to control rapidly changing requirements. For example, impact analyses may be inaccurate because it is impossible to define the baseline against which the changes will be implemented.
Completeness
Missing or incompletely specified requirements may appear in many forms, such as a requirements document with many functions or parameters “to be defined”; requirements that are not specified adequately to develop acceptance criteria, or inadvertently omitted requirements. When missing information is not supplied in a timely manner, implementation may be based on contractor assumptions that differ from customer expectations. When customer expectations are not documented in the specification, they are not budgeted into the cost and schedule.
Clarity
This attribute refers to ambiguously or imprecisely written individual requirements that are not resolved until late in the development phase. This lack of a mutual contractor and customer understanding may require rework to meet the customer intent for a requirement.
Validity
This attribute refers to whether the aggregate requirements reflect customer intentions for the product. This may be affected by misunderstandings of the written requirements by the contractor or customer, unwritten customer expectations or requirements, or a specification in which the end user did not have inputs. This attribute is affected by the completeness and clarity attributes of the requirements specifications, but refers to the larger question of the system as a whole meeting customer intent.
Feasibility
The feasibility attribute refers to the difficulty of implementing a single technical or operational requirement, or of simultaneously meeting conflicting requirements. Sometimes two requirements by themselves are feasible, but together are not; they cannot both exist in the same product at the same time. Also included is the ability to determine an adequate qualification method for demonstration that the system satisfies the requirement.
Precedent
The precedent attribute concerns capabilities that have not been successfully implemented in any existing systems or are beyond the experience of program personnel or of the company. The degree of risk depends on allocation of additional schedule and budget to determine the feasibility of their implementation; contingency plans in case the requirements are not feasible as stated; and flexibility in the contract to allocate implementation budget and schedule based on the outcome of the feasibility study. Even when unprecedented requirements are feasible, there may still be a risk of underestimating the difficulty of implementation and committing to an inadequate budget and schedule.
Scale
This attribute covers both technical and management challenges presented by large complex systems development. Technical challenges include satisfaction of timing, scheduling and response requirements, communication among processors, complexity of system integration, analysis of inter-component dependencies, and impact due to changes in requirements. Management of a large number of tasks and people introduces a complexity in such areas as project organization, delegation of responsibilities, communication among management and peers, and configuration management.

3.2.1. Analytic study of the attributes of requirement versus the risks

This study was realized to observe the behavior of the attributes in relation to the risks. As previous defined in the Section 3.1, the attributes of the chosen requirement to classify a requirement are: Correctness/Accuracy, Ambiguity, Difficulty, Completeness, Stability and the risks defined according to the SEI taxonomy are: Stability, Completeness, Clarity, Validity, Feasibility, Precedent, and Scale. The analysis was performed by two researchers with expertise in the risks area using the brainstorming technique. Following is presented an analysis of the relationship between attributes and risks.

Correctness/Accuracy: Attribute that defines if the requirements reflect the real intentions of the client, i.e., if the requirements satisfy the real needs expressed by him/her. If the requirement does not represent the user needs, this attribute can generate great risks regarding validity, enabling the delivery of a product that is different from client expectations. This attribute is split in five levels:

- Very High: The requirement presents all the client's real needs;
- High: The requirement presents the client's main needs;
- Average: The requirement presents some of the client's necessities;
- Low: The requirement presents few client's needs;
- Very Low: The requirement presents none of client's needs.

Stability: Attribute that defines the degree of stability of the requirement.

Some change factors such as errors, inconsistencies and conflicts in the requirements should be considered stability indicators [Hazan et al. 2003].

This attribute is split in five levels:

- Very High: Remote probability of change;
- High: Low probability of change;
- Average: There is no indicator of change in the requirement.
- Low: It is possible that the requirement changes.
- Very Low: It is highly probable that the requirement changes.

This attribute presents risks related to the stability risk of a requirement.

Difficulty: Attribute that defines the level of effort to implement the requirement. This attribute is split in five levels:

- Very High: Far above the average of effort to implement a requirement;
- High: Above average effort to implement the requirement;
- Average: Average effort to implement the requirement;
- Low: Below average effort to implement the requirement;
- Very Low: Far below the average effort to implement a requirement;

Depending on the level of this attribute, the requirement may present risks of feasibility, precedent, and scale impacting on compliance with the deadline set by the customer.

Ambiguity: Attribute that defines how much the requirement admits more than one interpretation. The ambiguity is related to understanding the requirement. Requirements that demand interpretation or are not clear, presents indicators of clarity risks. The writing of the requirement can make dubious sense or different from the real intention. This attribute is split in five levels:

- Very High: Requirement is totally misunderstood.
- High: Requirement admits many interpretations.
- Average: Requirement admits more than one interpretation.
- Low: Requirement asks for effort in the interpretation of the reading.
- Absent: Requirement is totally understood.

The ambiguity presents risks related to the clarity generating negative impacts without sufficient time to get around the situation. A study performed by Kendall [Kendall et al. 2007] states that the lack of clarity is the third biggest source of risks in the software projects.

Completeness: Attribute that defines the degree of completeness of the requirement. Incomplete requirements do not allow the correct implementation of requirements and will have to be modified in a posterior moment.

Complete requirements increase the quality of codification, thus helping the fulfillment of the deadline, correct execution under the budget and project's resources. Incomplete requirements make more difficult the test execution and the code verification. This attribute is split in five levels:

- Very High: Requirement has all necessary information;
- High: Requirement has just important information;
- Average: Requirement has just the essential information;
- Low: Requirement has little information;
- Very Low: Requirement has no relevant information;

Thus, the attribute completeness can be associated with risks of completeness, clarity and stability.

The Table 3.2 shows the relationship between attributes and risks previously defined and the values of each attribute are defined in Table 3.3.

Table 3.2. Attributes versus risks

Attributes	Risks						
	Stability	Completeness	Clarity	Validity	Feasibility	Precedent	Scale
Correctness				X			
Ambiguity			X				
Difficulty					X	X	X
Completeness	X	X	X				
Stability	X						

Table 3.3. Attributes levels

ID	Attributes	Attributes levels				
		1	2	3	4	5
A ₁	Correctness/Accuracy	Very High	High	Average	Low	Very Low
A ₂	Ambiguity	Very High	High	Average	Low	Absent
A ₃	Difficulty	Very High	High	Average	Low	Very Low
A ₄	Completeness	Very High	High	Average	Low	Very Low
A ₅	Stability	Very High	High	Average	Low	Very Low

3.3. SIMILARITY CALCULATION

To recover a similar requirement stored in a database we will analyze it based on the concept of similarity. According to Martins [Martins 2000], Similarity is the formalization of a certain philosophy of judging similarity using a concrete mathematical model. In general, we take as true the hypothesis: "Similar problems have similar solutions" in the same application domain.

The objective of this technique is to recover requirements whose risks can be the same given a new requirement, therefore the similarity between

the requirements is directly connected to their risks. In this technique, we propose to calculate the similarity between requirements through a set of common attributes between the requirements defined in Section 3.1. The comparison between requirements is done firstly between the attributes of the new requirement regarding a given requirement on the database. Each requirement compared will receive a value of similarity. Similar requirements will be returned, each one with its value of similarity.

In the study developed by Lins [Lins 2007] the calculation of similarity used was the sequential. This work was evaluated by Trigo [Trigo et al. 2007] through experimental studies. In the Trigo work the calculation of original similarity was compared in relation to its performance to other three algorithms of similarity. The best result obtained in the experiment was the algorithm of similarity with Euclidean Distance. In relation to requirement, we cannot say that using the Euclidean distance, the results will be a better performance. Due to this uncertainty, in Chapter 4 an evaluation that will analyze the performance of two algorithms will be presented: Sequential and Euclidean Distance to verify which calculation identifies more similarity among the requirements. From these results, the calculation of similarity used in the technique RITRA will be defined. In the following subsections, the calculation of the sequential similarity and the calculation of the similarity with Euclidean distance will be presented.

3.3.1. Calculation of the sequential similarity

The model Calculation of sequential similarity is split into two main parts, (i) Calculation of local similarity and (ii) Calculation of global similarity. The first calculates the similarity attribute by attribute, while the second calculates the average on the values calculated previously. From this, one value of similarity regarding the present requirement will be assigned to the recovered requirement. From that value, the requirement with a greater similarity in relation to the new one will be chosen. The formula used in this work is based on the calculation of similarity used in case based reasoning from Martins [Martins 2000]. However, instead of cases, the formulas were adapted to be

an applied requirement, which does not influence the efficiency of the equation.

i) Calculation of local similarity

The calculation of similarity proposed in this work is based on technique of local similarity (measure used in comparing attributes levels, where each attribute level of a requirement is compared individually with the contents of the current requirement). In other words, the comparisons will be made between level attribute and not between requirements. The requirement attributes used and attributes levels were defined in the Table 3.3.. The calculation of similarity between two attributes is defined through the Equation (3.1):

$$S = (N - 1) - |A_n - A_s| \quad (3.1)$$

Where:

S = Similarity between attributes

N = Quantity of values that the attributes can take

A_n = Values of the attributes of the present requirement

A_s = Values of the attributes of the stored requirement.

ii) Calculation of global similarity

The following is the equation to calculate the global similarity. It will provide a value to compare the present requirement with the requirement recovered from database. The global similarity is the arithmetic mean of the local similarities that are found. The Equation (3.2) presents the calculation of global similarity

$$S_g = \frac{(S_1 + S_2 + S_3 + S_4 + S_5)}{5} \quad (3.2)$$

Where:

S_g = Similarity between requirements. The higher the value of S_g , the greater the similarity between the requirements. This way, it is possible to create a ranking of similarity from the requirements stored in the database.

S_1, S_2, S_3, S_4, S_5 = Similarities between attribute values obtained from equation 3.1

Both similarities, global and local present values ranging from 0 (zero) to 4 (four). Being 0 the minimum similarity, and 4 the maximum similarity.

3.3.2. Euclidean Distance similarity

In this variation, instead of using the calculation in two steps (local similarity and global similarity) the calculation of similarity is performed in just one step. This step consists in calculating the Euclidean distance among all attributes. To do it the equation (3.3) (expression for the calculation of global similarity through Euclidean Distance) is used:

$$S_g = \sqrt{\sum_{i=1}^n (N_i - S_i)^2} \quad (3.3)$$

In Equation (3.3), S_g is the global similarity, N_i are the attributes levels of the requirement, S_i are the attributes levels of the requirement which is in the database. The method of the Euclidean distance will make the classification by the shortest distance found. The similarity between the requirements is bigger as close as zero. The similarity values range from 0 (zero) to 8.9 (eight dot nine). It is important to note that both similarity algorithms presented do not have weights associated. In some cases it is important to differentiate the importance of a given attribute in a specific scenario, similar to what has been observed in other studies [Lins 2007, Trigo et al. 2007]. It is known that requirements' attributes are different from each other and might yield different importance in different scenarios, which might indicate the use of weights. However, in this work it was considered that requirements have the same importance due to a lack of detailed study about the importance of specific requirements' attributes.

3.4. RECOVERY AND STORAGE OF REQUIREMENTS

The main goal of requirements recovery is to obtain from the database, all the risks identified from requirements that may be relevant to the new

requirement. In this work, the similarity value is calculated to all requirements on the database sequentially, in other words, one after the other. This way, there is an assurance that the requirements will be compared one by one. Then the requirements are placed in descending order according to the similarity to the problem. The three requirements with higher similarity values will be retrieved and risks associated to them will be added to the new one. New risks can be identified in the new requirement through other techniques, and inappropriate risks that have been imported can be removed. At this point, this requirement is inserted into the database.

3.5. RITRA MODELING

The following tables present activities defined by the RITRA technique. The pre-condition for executing the technique is a requirements database with identified risks associated to them. Table 3. and Table 3. present RITRA activities:

Table 3.4. Activity 1: Requirement Attributes Classification

Goal	To classify attributes of each requirement
Responsible	Project manager and/or Business Analyst
Artifacts input	<ul style="list-style-type: none"> - Software requirement document - Attributes classification spreadsheet (Appendix A1) - Risks requirements Description (Appendix A3)
Artifacts output	- Attributes classification spreadsheet Filled
Steps	The project manager and/or business analyst will evaluate the requirements in accordance with the values of the attributes defined in the document (Appendix A3)

Table 3.5. Activity 2: Risk identification

Goal	Risk identification in new requirement
Responsible	Test engineers
Artifacts input	- New requirements with classified attributes by project manager and/ or Business Analyst - Data base with identified risks
Artifacts output	- New requirement with identified risks
Steps	Test engineer insert new requirement with values of attributes associated (Correcteness/Ambiguity/Difficulty/Completeness/Stability) identified previously by the Project Manager and/or Business Analyst. Then the new requirement is compared with all previous requirements by similarity. After this, the similarity value of all base requirements is recorded. The 3 (three) most similar requirements in the database are returned with their associated risks. The test engineer extracts the risks for the new requirement and adds new risks related to it using other techniques for risk identification and finally the new requirement is inserted into the database.

RITRA life cycle is shown in Figure 3.2 and it is modeled in BPMN through the tool free BizAgi.¹

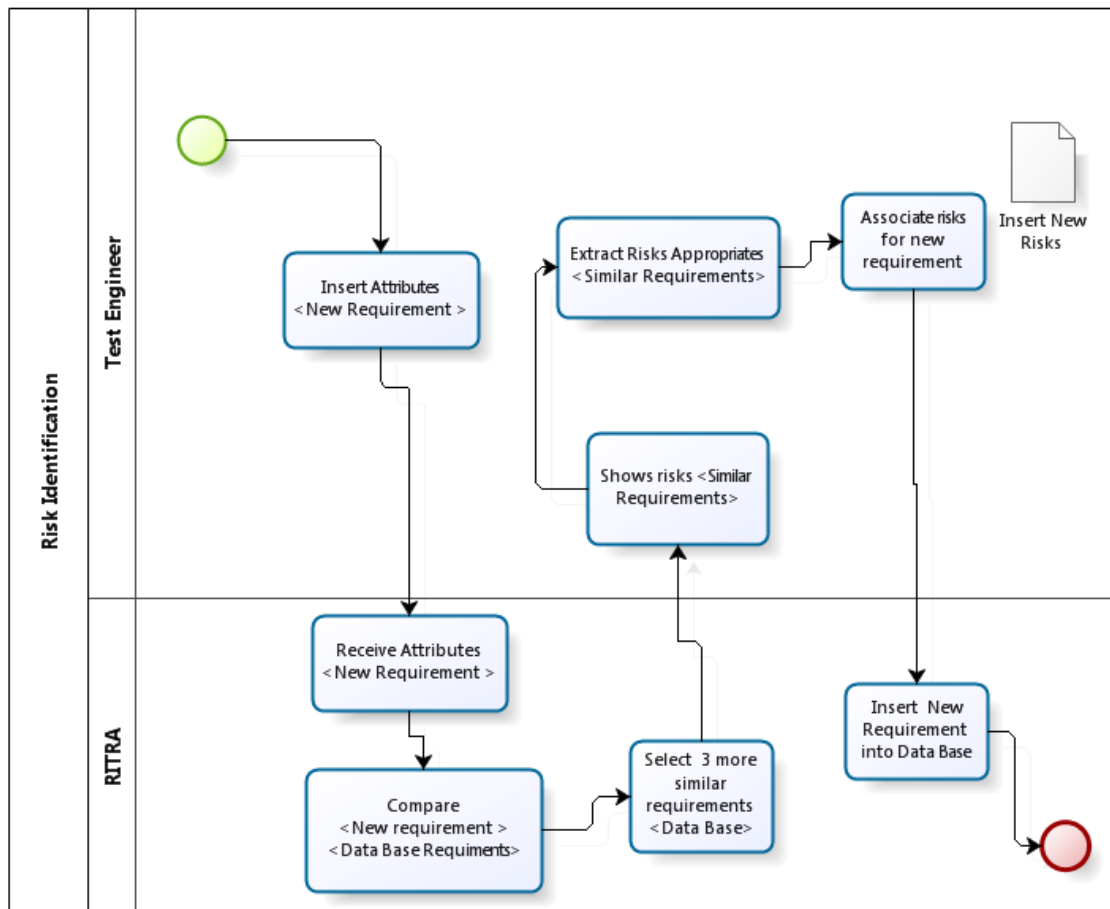


Figure 3.2. RITRA LifeCycle

¹ BizAgi <http://www.bizagi.com>

For greater efficiency and quality, RITRA automation is indicated, however, there was insufficient time to implement it. Tables 3.6 and 3.7 describes the features and technologies respectively that can be used in the development for future implementation, as well as the class diagram to represent the structure and relationships between classes serving as a model for objects.

Table 3.6. Suggestions for RITRA features

Features	Description
Register Requirement	This feature allows the user to classify the requirement according to the attributes specified in the technique.
Show requirements associated with the risk	Allows viewing of the requirements associated with the risks
Risks Register	Allows registering new risks
View registered Risks	Allows viewing of registered risk
Identify risks in the new requirement	Allows calculating similarity between the new requirement and all Database.
Insert requirement with risks identified in the database	This feature inserts the requirement in the database.
Show Requirements in order of similarity	Allows viewing of the requirements by presenting the similarity value.

Table 3.7. Suggestions for the RITRA technique automation

Programming Language: Java
Java is an object-oriented programming language, created in the early 90's by Sun Microsystems. Java was recognized in 1995 due to the growth of the web [Cornell et al. 2000]. The main language features beyond object orientation are: portability, independent platform execution, and security due to the support of programs via networking, and similar syntax to the languages C/C++. Besides being free and widely used by software developers.
Database MySQL:
MySQL [MySQL 2011] is a database management system (DBMS) that uses the SQL (Structured Query Language) as an interface. MySQL was chosen because it provided a great popularity among software developers, support almost all platforms today, have a great performance and stability, being easy to use and free software based on GPL (General Public License).
Persistence: JPA:
Java Persistence API [Biswas et al. 2011], called JPA for short, is a Java API for data persistence. JPA defines a way to perform object-relational mapping for Java objects called entity beans.
Development Environment: Eclipse
The Eclipse IDE (Integrated Development Environment) for Java developers is a platform focused on developing tools and software applications. It provides Java editing with validation, incremental compilation, cross-referencing, code assistance and an XML Editor [Eclipse 2011]. The Eclipse IDE is adaptable to different needs of each user and very used tool in the world [Eclipse 2011].
Case Tool: astah*/JUDE
The tool Astah*/JUDE [Astah 2011] is used for the creation of analysis and project diagrams. Astah (previously named JUDE) is a CASE tool for UML modeling created by the Japanese company ChangeVision. Astah was chosen because it supports the creation of several diagrams, such as, class diagrams, use cases diagrams, and sequence diagrams, etc.

Figure 3.3 shows modeling of RITRA through the class diagram to represent the structure and relationships between classes serving as a model for objects.

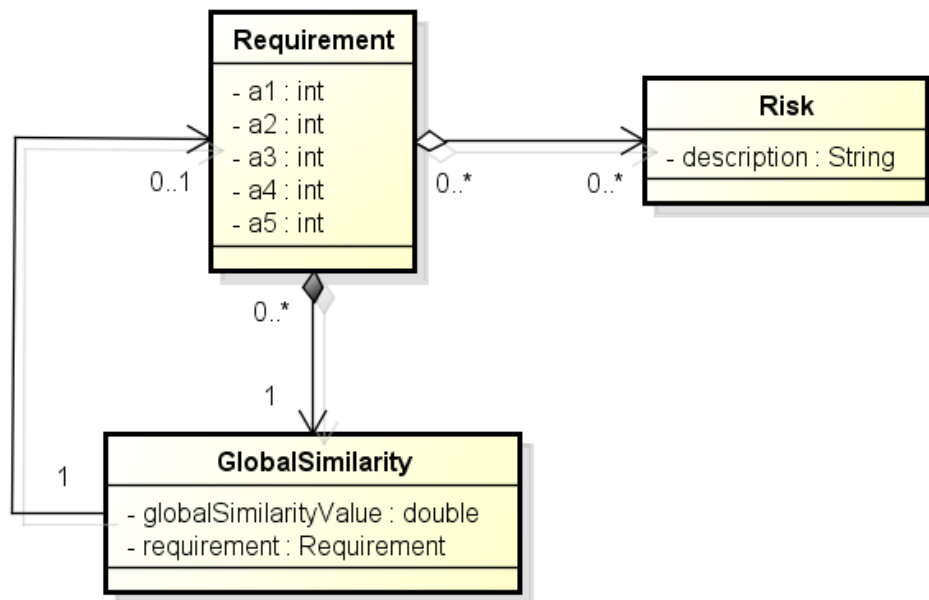


Figure 3.3. RITRA class diagram.

Figure 3.4 shows the overview of the proposed technique.

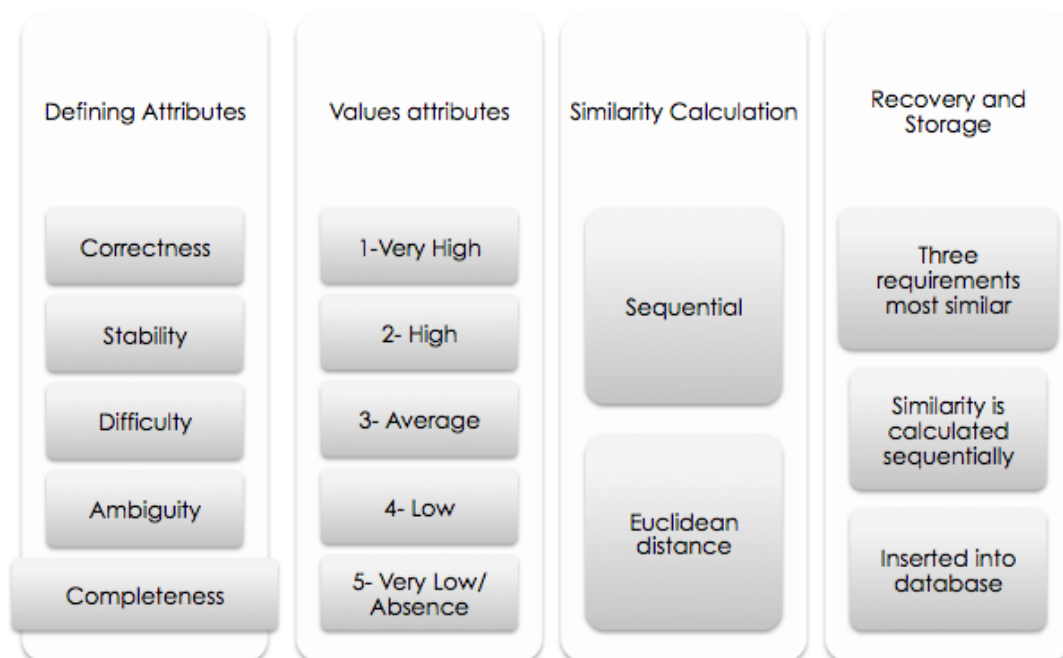


Figure 3.4. RITRA technique

This chapter aims at presenting two evaluations of the RITRA technique. Section 4.1 contains the evaluation between the similarity algorithms. The result of this evaluation will define the algorithm, which is going to be used with the RITRA technique. In Section 4.2 a controlled experiment is performed in order to verify the usage of the technique.

4.1. SIMILARITY ALGORITHMS EVALUATION

This evaluation aims at verifying the performance of two similarity algorithms: sequential and Euclidean distance. These algorithms are presented in Sections 3.3.1 and 3.3.2 respectively. Similarity algorithms evaluation is presented in the Section 4.1.1 describing the evaluation's definition and Section 4.1.2 showing the analysis of results.

4.1.1. Definition

This section is divided in two phases and presents how data were collected and evaluated. Each phase presents two activities. In what follows, the entire process is described.

Phase A: Data Collection

This phase defines how data were collected. It aims at classifying the attributes of each requirement and building a requirements database with associated risks. Activities A1 and A2 from this phase are presented next.

Activity A1: Classification of the requirements attributes

The data were collected through electronic (and online) spreadsheets by project managers due to their knowledge about the systems. These spreadsheets contain the values of the attributes the requirements defined in the Section 3.1. The used requirements came from the system which belongs to the InteliMed Mobile Project, versions 1.3 and 1.5 (Annex A). The InteliMED Mobile² is a product to support medical diagnostics using smart technologies in mobile equipments. It is integrated in a way that supports the diagnostication services and a second medical opinion. InteliMED is a decision support system (DSS) based on intelligent computing techniques.

Activity A2: To build a database

Due to lack of a real requirements database with identified risks, after activity one was performed, the current activity was executed by three risk experts with at least experience of two years in risks. The risks were associated to the requirements using the questionnaire of risks taxonomy (Annex B) and according to the experience of each risk expert. The results of this collection were inserted in an electronic spreadsheet. Table 4.1 presents activities A1 and A2 procedure.

² Project InteliMED – <http://promise.cin.ufpe.br/intelimed>

Table 4.1. Activities A1 and A2 Procedures

A1: Classification of the requirements' attributes	
Sponsor	Project Manager and/or Business Analyst
Steps	1- Read description of requirements' risks (Appendix A3) 2- Read requirements document (Annex A) 3- Classify requirements' attributes 4- Fill in the attributes classification Spreadsheet (Appendix A1)
Input Artifacts	-Attributes classification spreadsheet (Appendix A1) - Requirements document (Annex A) - Requirements' risks description (Appendix A3)
Output Artifacts	- Attributes classification spreadsheet Filled
A2: To build Data Base	
Sponsor	Risks experts
Steps	1- Read data base spreadsheet (Appendix A5) 2- Identify risks using questionnaire TBR or personal experience (Annex B) 3- Fill in the data base spreadsheet (Appendix A5)
Input Artifacts	- Attributes classification spreadsheet filled -Questionnaire of risk Taxonomy- Based Risk identification -TBR(Annex B) - Requirements document (Annex A) -Data base spreadsheet (appendix A5)
Output Artifacts	Data base spreadsheet

Phase B.Data Evaluation

The goal of the data evaluation is to compare the performance of two similarity algorithms using the data collected. The sponsor for this phase is a risk expert with at least a one-year experience. The evaluator cannot be one of the three experts that participated in the data collection phase. Next, activities B1 and B2 will be presented.

Activity B1: Executing Algorithms

A new requirement with attributes classified by the project manager is compared with all requirements from the database. The algorithms were inserted into a MS Excel™ Spreadsheet. First the sequential algorithm (Algorithm One) is used, then the euclidean distance (Algorithm Two). After the algorithm execution, the top three most similar requirements will be selected. The identified risks in each similar requirement is compared and associated to new requirement. This comparison must be done with other two similar requirements. After this, run the same procedure using Algorithm Two.

Activity B2: Results Assessment

The evaluator fills Table 4.2 to analyze which algorithm obtained the best similarity performance. The used metric is defined by Equation (4.1): Following the metric to evaluate the performance of the similarity calculation of the individually requirement is presented:

$$M_{1...3} = \frac{\text{risks total in the new requirements from similar requirement}}{\text{Risks total in the similar requirement}} \quad (4.1)$$

Table 4.2. Performance analysis of similarity algorithms

	Algorithm 1	Algorithm 2
Similar Requirement 1	M_1	M_1
Similar Requirement 2	M_2	M_2
Similar Requirement 3	M_3	M_3
Total Metric	$\frac{\sum_{i=1}^3 M_i}{3}$	$\frac{\sum_{i=1}^3 M_i}{3}$

The metric ranges from 0 (zero) to 1(one). The closer to one, the more similar the requirement is. Table 4.3 presents activities B1 and B2:

Table 4.3. Activities B1 and B2 Procedures

B1: Executing Algorithms	
Sponsor	Evaluator (Profile Risk Expert)
Steps	Execute for three new requirements 1- Insert attributes values for the requirement into spreadsheet 2- Execute the algorithm (sequential and Euclidean) 3- Select the top three most similar requirements 4- Back for step 1
Input Artifacts	- Three new requirements (Appendix A4) - Database spreadsheet - Spreadsheet with algorithms
Output Artifacts	- Spreadsheet with data executed
B2: Results Assessment	
Sponsor	Evaluator (Profile Risk Expert)
Steps	- Assessment of results from metric 4.1
Input Artifacts	- Spreadsheet with data executed - Metrics
Output Artifacts	- Spreadsheet results

4.1.2. Results Analysis

Next, results from activities defined in Section 4.1.1 are presented and analyzed.

Activity A1 and A2 Results: Result of the attributes classification and risks identification in requirements.

The functional requirements for the two versions of IntelliMED were classified by the project manager. Versions 1.3 and 1.5 had a total of 36 functional requirements, from which 16 requirements for version 1.3 do not have risks identified by the risks experts in Activity A2. The attributes classification of the 20 requirements is showed in Appendix B1. The requirements with identified risks that were analyzed in this phase can be visualized in the Appendix B2.

Activity B1 Results: Result of the evaluator's analysis

Three requirements from version 1.5 of the project InteliMED, that did not have their risks previously identified, were used in this analysis. The attributes of these requirements were classified by the project manager and they are presented in Appendix A4.

The requirements were submitted for testing with the technique using the two algorithms. The analysis was carried out using the MS Excel™. The result is a requirement list sorted in a descending order by similarity value. The sequential algorithm presents the value 4 as the highest similarity and 0 as lowest similarity, whereas, the Euclidean distance algorithm presents 0 as the greatest similarity and 8,9 as the lowest similarity. Table 4.4 presents the result of the comparison of 3 requirements with the database. The complete result is shown in Appendix B3.

Table 4.4. Result of the technique application

Requirement	Similar Requirement			
	Sequential		Euclidean	
REQ_021	REQ_05	3,8	REQ_05	1,0
	REQ_06	3,4	REQ_06	1,7
	REQ_02	3,4	REQ_02	1,7
	REQ_07	3,4	-	-
REQ_022	REQ_06	3,6	REQ_06	1,4
	REQ_02	3,6	REQ_02	1,4
	REQ_01	3,6	REQ_01	1,4
REQ_023	REQ_06	3,4	REQ_07	1,7
	REQ_02	3,4	REQ_17	1,7
	REQ_16	3,4	REQ_12	2,0
	REQ_07	3,4	-	-
	REQ_17	3,4	-	-

After the execution of the technique using the two similarity algorithms, it was observed that the similar requirements identified were practically the same, invalidating the analysis from the risks, as defined in the Section 4.1. Among the three requirements analyzed after execution of the two algorithms, the REQ_023 presented a greater difference between similar requirements returned, thus chosen for the analysis of the algorithms' performance in relation to the requirement database. Table 4.5 presents the position,

attributes' values for each requirement, its respective distance in relation to new requirement and the values of each algorithm.

Table 4.5. Behavior of algorithms in the selection of similar requirements

REQ_2322334												
Position	Sequencial					Value	Euclidean					Value
1	REQ_0222252					3,4	REQ_0711333					1,7
	Distancia00122						Distancia11001					
	REQ_0622353						REQ_1711234					
	Distancia00021						Distancia11100					
	REQ_0711333											
	Distancia11001											
	REQ_1622342											
	Distancia00012											
	REQ_1711234											
	Distancia11100											
2	REQ_1223223					3,2	REQ_1223223					2,0
	Distancia01111						Distancia01111					
3							REQ_0222252					2,2
							Distancia00122					
							REQ_0622353					
							Distancia00021					
							REQ_1622342					
							Distancia00012					

Observing Table 4.5, while the sequential algorithm defined the same value of similarity for requirements 2, 6, 7, 16, and 17, the Euclidean algorithm had a different result. In the latter, requirements 7 and 17 are in a higher position than requirements 6 and 16, due to the requirements 6 and 16 presenting 3 equals attributes. They present attributes with a two-level distance, whereas the requirements 7 and 17 present 2 equal attributes, but the other attributes are just one level away. Thus, it is possible to note that the Euclidean algorithm has a better homogeneity between the attributes than the sequential one. Because of this, the Euclidean distance algorithm was chosen to be used in the RITRA technique.

4.2. EXPERIMENTAL STUDY

This section reports an experimental study plan specifying the definition, planning, execution and results analysis to assess the technique proposed in Chapter 3. This study analyzes if the technique can help professionals who do not have expertise in risk management to identify risks from software requirements.

4.2.1. Definition

The following sections define the global purpose of this experimental study.

4.2.1.1. Global goal

The global goal of this study is to evaluate the proposed technique in order to verify its behavior in identifying risks in requirements from the perspective of test engineers who do not have knowledge in risks.

4.2.1.2. Study goal

Analyze the proposed technique, aiming at observing its behavior in the identification of risks from software requirements, considering the lack of knowledge in the area of risk from the perspective of test engineers. All subjects had at least one year of experience in software testing.

4.2.1.3. Measurement Goal

Considering the technique that will be analyzed, the following factors must be observed:

- Number of risks identified consistent. (The concept of consistent identified risks is presented in Section Metrics)
- Time required identifying risks from requirements.

4.2.1.4. Metrics

- Data that represent the number of risks identified "consistent" from the requirements.
- Data that represent the time required to identify risks from the requirements. Data will be collected in minutes.

In this study "consistent" is defined as the risks identified from the intersection of the results found by specialists using the RITRA technique and ad hoc . These data will be collected through the following defined metrics that compare the number of equal risks identified as the result of the intersection of risks found by specialist and risks identified by the test engineers with no knowledge in risks.

Number of correct risks using the RITRA technique:

$$QRI \Rightarrow RIE_{TRAT} = RIL_{TRAT}$$

Number of correct risks using ad hoc. :

$$QRI \Rightarrow RIE_{STRA} = RIL_{STRA}$$

Where,

QRI – Number of risks consistent identified.

RIE – Risks identified by risk experts resulting from the intersection of the risks identified using the RITRA technique and ad hoc.

$$RIE = RIE_{TRAT} \cap RIE_{STR}$$

Where,

RIE_{TRAT} – Risks Identified by risk experts using the RITRA technique.

RIE_{STRA} – Risks Identified ad hoc by risk experts.

RIL_{TRAT} – Risks Identified by the test engineers using the RITRA technique.

RIL_{STRA} – Risks Identified ad hoc by the test engineers.

4.2.2. Planning

The following sections describe the steps taken during the planning of this experiment, defining how this study will be structured.

4.2.2.1. *Selecting the Context*

The experiment will be executed in the context of software engineering professionals of the state of Pernambuco. The team of participants will be split in 4 groups:

- Group 1: Risk Experts who will execute the RITRA technique in the risk identification of requirements;
- Group 2: Risk Experts who will identify risks ad hoc, having just the requirements, SEI taxonomy questionnaire and risk classification;
- Group 3: Test Engineers without knowledge in risks, who will execute the RITRA technique in the risk identification of requirements;
- Group 4: Test Engineers without knowledge in risks who will identify risks ad hoc, having just the requirements, SEI taxonomy questionnaire and risk classification.

The intersection of results of risks identified by the group of specialists will be compared to the results of test engineers without knowledge in risks.

4.2.2.2. *Metrics Definition*

Next some symbols will be introduced to represent the metrics that will be collected and analyzed. These symbols will be used in this chapter.

QRI – Number of risks consistent identified.

TIR – Time identifying risks

Each metric will have two variations and will be analyzed for professionals without knowledge in risks.

QRI_{TRAT} – Number of risks consistent identified using the RITRA technique.

QRI_{STRA} – Number of risks consistent identified ad hoc.

TIR_{TRAT} – Time identifying risks using the RITRA technique.

TIR_{STRA} – Time identifying risks ad hoc.

4.2.2.3. *Selection of the Variables*

This section defines the dependent and independent variables of the experiment.

4.2.2.3.1. Independent Variables

The independent variables are the manipulated and controlled ones:

- Experience of the subjects (risk experts and test engineers);
- Risk identification of requirements;
- Software requirements used (Appendix B2);
- Risk Definition [Carr et al. 1993]
- Questionnaire Taxonomy Based Risk (Annex B);
- Using of Excel spreadsheets.

4.2.2.3.2. Dependent Variables

The dependent variables are the variables obtained from the change of independent variables.

- Number of identified risks;
- Time identifying risks.

4.2.2.4. *Sample Selection*

To participate in this study, 14 professionals of software engineering were chosen by convenience. Table 4.6 presents years experience of each professional in their respective areas. Six professionals are risk experts and eight professional are test engineers.

Table 4.6. Years Experience

	Participants	Experience Time
Risk Expert	1	Two years
	2	Two years
	3	Six years
	4	Two years
	5	Two years
	6	Nine years
Test Engineer	1	Four years
	2	One year
	3	Two years
	4	One year
	5	One year and half
	6	Two years
	7	Four years
	8	One year

These professionals were split into four groups defined in Section 4.2.2.1.

4.2.2.5. Experiment Design

An experiment consists of a set of tests for the treatments. The design of an experiment describes how these tests will be organized and executed. The experiment will be divided into seven activities, each one having the purpose of collecting specific metrics. Table 4.7 introduces the scope of the experiment with its activities, procedures, responsible, and metrics to be collected.

Table 4.7. Experiment Scope

Activity	Procedure	Responsible	Metric
A ₁ : Risk Identification with the RITRA technique	Identify Risk with the RITRA technique	Risk Expert	RIE _{TRAT} TIR _{TRAT}
A ₂ : Risk Identification ad hoc	Identify Risk ad hoc	Risk Expert	RIE _{STRA} TIR _{STRA}
A ₃ : Definition of the Correct Risks	Intersection between the risks of the activities A ₁ and A ₂	Evaluator	RIE
A ₄ : Risk Identification with the RITRA technique	Identify risks with the RITRA technique	Test Engineer	RIL _{TRAT} TIR _{TRAT}
A ₅ : Risk Identification ad hoc	Identify risks ad hoc	Test Engineer	RIL _{STRA} TIR _{STRA}
A ₆ : Number of Risks Identified With the RITRA technique.	Comparison between the risks of the activities A ₃ and A ₄	Evaluators	QRI _{TRAT}
A ₇ : Number of Risks Identified ad hoc	Comparison between the risks of the activities A ₃ and A ₅	Evaluators	QRI _{STRA}

The activities A₁ and A₂ are defined in this experiment as a control object that will be compared to the data obtained by test engineers. The comparison

between the values resulting from the group of experts and test engineers is required to measure the RITRA technique impact. The following sections describe how the experiment was designed.

4.2.2.5.1. Treatment

This part of the study consists of an experiment with one factor and two treatments. The factor is the way to identify risks and the treatments are the use of the RITRA technique and ad hoc.

In the experiment the sample is split in four groups, according to the Section 4.2.2.1. Each group will run the same set of requirements. Table 4.8 shows the application of the treatments to the samples.

Table 4.8. Application of the treatments to the samples

Sample	With RITRA	Ad hoc
Group 1	X	
Group 2		X
Group 3	X	
Group 4		X

4.2.2.5.2. Project

A database with identified risks from the first evaluation(Appendix B2) will be used in this experiment.

4.2.2.6. Instrumentation

The purpose of the instrumentation it is provide a way to execute the experiment and monitoring it without affecting its results. The object of this experiment is the set of project requirements that will be used to perform activities related to the metrics defined in Section 4.2.2.2, from which the data of the execution will be collected. The activities will be described and kept in an Excel spreadsheet. In this spreadsheet the data of execution of each activities will also be stored. The time will be collected using a conventional chronometer.

4.2.2.7. *Threats to validity*

This section describes the study's threats to validity. In this study are defined four types of threats to validity: internal, conclusion, construct and external validities. Internal validity ensures that the obtained outcomes from the variables are not being influenced by any uncontrolled factor. Conclusion validity is related to the ability to establish a correct relationship between the treatment applied and the results obtained. The construct validity is to ensure that the treatment reflects the cause and the results reflect the effect. Finally, the external validity is related to the ability to generalize results to other contexts.

4.2.2.7.1. Internal Validity

The subjects of the study are professionals in Software Engineering, six are specialists in risks and do research in the area for at least one year. The other eight subjects do not have knowledge in risks and are test engineers. To ensure that no external factor will influence the experiment, the technique will be previously explained for the group with no knowledge in risks, so they can understand the technique lifecycle and execute it without doubts. Another threat to the internal validity can be the quality factor in the experience of risk experts. However, a questionnaire will be previously applied about their knowledge and their experience in the risk area.

4.2.2.7.2. Conclusion Validity

The subjects will be instructed about the importance of reporting real and coherent data to the experiment. Besides, a coordinator will be always present during the execution of the experiment to ensure that the data will be collected in a correct way. This experiment does not employ any statistical test because of the small sample size. An exploratory data analysis is performed.

4.2.2.7.3. Construct Validity

The subjects will apply the treatment using requirements of a real project. A factor that could cause confusion is the fact of participants reporting a time for activities less than what was really spent, to demonstrate greater efficiency. To get around this, participants will be informed that their names will be omitted from this study and will not be possible to identify who performed the activities in the shortest time.

4.2.2.7.4. External Validity

An expected result of this study is to evaluate the behavior of professionals that are not risk experts in the identification of risks using the proposed technique. The results obtained with the experiment cannot be generalized to other contexts because of the limited number of participants and their profiles.

4.2.3. Operation

4.2.3.1. Preparation

The subjects of this study were previously informed that they would be participating on the experiment. To ensure their commitment, the subjects will be advised of the importance of their participation in all stages of the experiment. To ensure confidentiality of the study, their names will be omitted in the results.

4.2.3.2. Execution

The experiment was planned to run in the laboratories of CIn - UFPE. The risk experts, however, due to other professional activities, executed the experiment in different locations, but with appropriate conditions. The test engineers as professionals in a company incubated within CIn performed the experiment in person. All participants were instructed to perform the experiment's activities. The tool used was MS Excel™.

Members of both groups executed the predefined scenarios, measuring the time required for each step. Before performing the experiment, a dry run was performed with duration of one hour. The samples collected during the pilot execution were not used in statistical tests, they only served to ensure that the developed scenarios were possible to be accomplished. It was reserved one hour to run the scenarios. Table 4.9 and Table 4.10 show the description of the scenarios that were executed by participants of this study.

Table 4.9. Description of Scenario A

Scenario A	
Activity:	Identify risks using the RITRA technique
Subjects	Group 1 and Group 3
Duration	one hour
Tool	MS Excel
Steps	
<ul style="list-style-type: none"> • Read the requirement [021]*; • Insert the values of the attributes of the requirements; • Run RITRA; • Sort the requirement <ascendant> • Identify the first three requirements; • Select the risks associated with the three requirements in the database; • Associate the risks to the new requirement; • Choose the risks you think that belong to the requirement [021]. • Insert the requirement [021] in the database.** <p>* Start counting the time * **Stop counting the time</p> <p>When this scenario is over, execute it for requirements [022] and [023].</p>	
Input	Output
<ul style="list-style-type: none"> - Database of requirements with risks previously identified; - Requirements [021], [022], [023] with values of the attributes;(Appendix A4) - Spreadsheet with the RITRA technique. -Data base spreadsheet (Appendix A5) 	<ul style="list-style-type: none"> - Requirements [021], [022] and [023] with identified risks.

Table 4.10. Description of Scenario B

Scenario B	
Activity:	Identify risks ad hoc
Subjects:	Group 2 and Group 4
Duration:	one hour
Tool:	MS Excel
Steps	
<ul style="list-style-type: none"> • Read requirement [021]*; • Identify the risks using the taxonomy questionnaire of SEI or personal experience; • List of identified risks** <p>* Start counting the time * **Stop counting the time</p> <p>When this scenario is over, execute it for requirements [022] and [023].</p>	
Inputs	Output
<ul style="list-style-type: none"> - Document with the definitions and classification of requirements' risks(Table 3.1) - Requirements [021], [022], [023] (Appendix A4) - Taxonomy questionnaire from SEI [Carr et al. 1993] (Annex B) - Spreadsheet for risk identification -Data base spreadsheet (appendix A5) 	<ul style="list-style-type: none"> - Requirements [021], [022], [023] with risks identified.

4.2.4. Data Analysis

The purpose of this analysis is to verify the behavior of RITRA based on the collected data during the experiment execution. The following criteria were used:

QRI – Number of risks consistent identified.

TIR – Time identifying risks

A group of 14 Software Engineering professionals participated in this experiment. From this group, eight were test engineers and six were risk experts. Table 4.11 shows the profiles of the subjects.

Table 4.11. Profile's experiment participants

Profile: Tests Engineers
Software test experience from one to four years. Four of them have bachelor degrees and the others have master degrees. Only one had a test certification. None of the participants has ability about risk based testing neither have knowledge about risk identification.
Profile: Risks Experts
Risks experience from two to nine years. Four of them have bachelor degrees and two have a master degree. Only one has a test certification. All of them have already used risk based testing approach. They present advanced knowledge advanced in risks.

To analyze the metric QRI, the concept of correct risk was defined in section 4.2.1.4. So, for this study it was defined as parameter of correct risk, the intersection between the results of the risks identified by the risk expert using technique and ad hoc. The process used in this experiment is presented in the following activities and it can be visualized in the Figure 4.1.

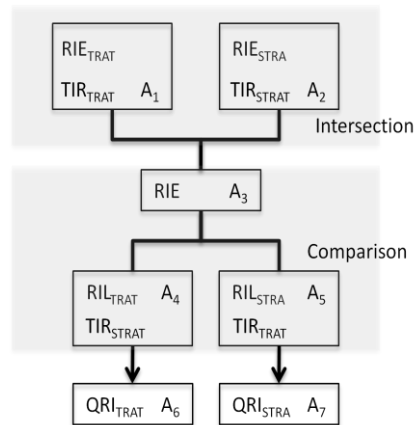


Figure 4.1. Experiment Process

Activity 1: Risks identified by risk expert using the RITRA technique

In the first activity, the subjects will run scenario A, defined in the Table 4.9. The results are presented in the Tables Table 4.12, Table 4.13, Table 4.14, and Table 4.15. This activity results in the values of the metrics RIE_{TRAT} – Risks identified by risk expert using the RITRA technique – and TIR_{TRAT} – Time identifying risks using the RITRA technique.

Table 4.12. Risks identified by risk expert 1 using the RITRA technique

Risks identified in Req_021
The requirement does not report the amount of characters that each attribute must have
The connection to the database can be stopped and the data have not been updated in the system.
The system can validate the user just with login
No information related to the size of the fields
No information related to the data type
No information related to the formatting
Does not report if validation in the data will be needed.
No information about transaction security .
No specification of the implementation details.
Risks identified in no Req_22
Does not report if validation in the data will be needed.
No information related to the size of the fields
The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.
Does not define the external interface
No information about transaction security .
No specification of the implementation details.
Define the type of the sent file.
Risks identified in no Req_023
The requirement does not contain the essential information, such as items of the questionnaire.
The information contained in the questionnaire can be improper, causing its invalidation.
Invalid information can be passed, causing an inappropriate use of it.
No information about transaction security .
Does not specify if all diagnostics will be sent to the server.

Table 4.13. Risks identified by risk expert 2 using the RITRA technique

Risks identified in Req_021
The requirement does not report the amount of characters that each attribute must have
The connection to the database can be stopped and the data have not been updated in the system.
No information related to the size of fields
No information about transaction security .
Define the type of the sent file.
Does not report if validation in the data will be needed.
Risks identified in Req_22
Does not define the external interface
No information about transaction security .
Define the type of the sent file.
Short time available for implementation
Risks identified in Req_023
The requirement does not contain the essential information, such as items of the questionnaire.
No definition of the criteria of the implementation
No experience in the implementation

Table 4.14. Risks identified by risk expert 3 using the RITRA technique

Risks identified in Req_021
The requirement does not report the amount of characters that each attribute must have
No information related to the data type
Does not define the external interface
No information about transaction security .
No specification of the implementation details.
Risks identified in Req_22
Does not report how the data will be stored
Does not define the external interface
No information about transaction security .
No specification of the implementation details.
Define the type of the sent file.
Does not report if will be need validation in the data.
The requirement does not report the amount of characters that each attribute must have
No information related to the size of fields
The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.
Risks identified in Req_023
The requirement does not contain the essential information, such as items of the questionnaire.
The information contained in the questionnaire can be improper, causing an invalidation of the questionnaire.
Invalid information can be passed, causing an inappropriate use of it.
No definition of the criteria of the implementation
No information about transaction security .
No experience in the implementation

Table 4.15. Time to identify risks using the RITRA technique by risk experts

	RE1	RE2	RE3	Average Time
Req_021	7	5	15	9
Req_022	6	3	9	6
Req_023	4	4	7	5

Legend: RE- Risk Experts

The results of previous Tables are summarized in the Table 4.16 showing the risks identification and the occurrence of each risk in the results of each Exper.

Table 4.16. Merge of results of the risk expert using the RITRA technique

Requirement	Identified risks	Occurrence
Req_021	The requirement does not report the amount of characters that each attribute must have	3
	The connection to the database can be stopped and the data have not been updated in the system.	2
	The system can validate the user just with login	1
	No information related to the size of fields	2
	No information related to the data type	2
	No information related to the formatting	1
	Does not report if will be need validation in the data.	2
	No information about transaction security .	3
	No specification of the implementation details.	2
	Define the type of the sent file.	1
	Does not define the external interface	1
Req_022	Does not report if data validation will be needed.	2
	No information related to the size of fields	2
	The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.	2
	Does not define the external interface	3
	No information about transaction security .	3
	No specification of the implementation details.	2
	Define the type of the sent file.	3
	Short time available for implementation	1
	Does not report how the data will be stored	1
	The requirement does not report the amount of characters that each attribute must have	1
Req_023	The requirement does not contain the essential information, such as items of the questionnaire.	3
	The information contained in the questionnaire can be improper, causing an invalidation of the questionnaire.	2
	Invalid information can be passed, causing an inappropriate use of it.	2
	No information about transaction security .	2
	Do not specify if all diagnostics will be sent to the server.	1
	No specification of the implementation details.	2
	No experience in the implementation	2

Activity 2: Risks identified ad hoc by risk expert

In this activity, scenario B, defined in the Table 4.10, was applied for a set of 3 risk experts. The result is presented in the Tables Table 4.17, Table 4.18, Table 4.19, and Table 4.20. This activity results in values for the metrics RIESTR – Risks identified ad hoc by risk expert – and TIRSTRA – Time identifying risks ad hoc.

Table 4.17. Risks identified ad hoc by risk expert 1

Risks identified in Req_021
The requirement does not inform the amount of characters for each field.
The information passed can be incoherent with the type of data that the database will store.
Access the system without inserting the password
No definition of the security policy
Risks identified in Req_22
No information in the requirement
The system may not have evidences realized in the day
No experience in implementation of the required communication
Send invalid files
Risks identified in Req_023
There is no important information in the requirement
The health agent is not the actor on the requirement
There is no essential information for the implementation.
No information about the security

Table 4.18. Risks identified ad hoc by risk expert 2

Risks identified in Req_021
Insertion of unwanted characters
There is no definition of space reserved for each field
Send login and password with empty fields
Risks identified in Req_22
Send invalid data
Does not define the external interface
Risks identified in Req_023
Data in the questionnaire can be invalid
There is no security definition

Table 4.19. Risks identified ad hoc by risk expert 3

Risks identified in Req_021
The requirement does not inform the amount of characters for each field.
Access the system without inserting the password
Does not define the external interface
Risks identified in Req_22
There is no definition of the amount of characters for each defined field
There is no knowledge in the used technology for the development
There is no information about the implementation
Risks identified in Req_023
There is no information about the implementation
Insert invalid data in the questionnaire
Show other different screen in the diagnostic

Table 4.20. Time to identify risks ad hoc by risk experts

	RE1	RE2	RE3	Average Time
Req_021	10	3	14	9
Req_022	13	5	12	10
Req_023	14	3	10	9

Legend: RE- Risk Experts

After the risk identification ad hoc by risk experts, one different expert analyzed the identified risks in order to eliminate duplicates. The result can be seen in Table 4.21.

Table 4.21. Merge of the risk identification ad hoc

Requirements	Identified Risks
Req_021	The requirement does not inform the amount of characters of each field.
	The information passed can be incoherent with the type of data that the database will store.
	Access the system without insert the password
	No definition of the politic of security
	Insertion of unwanted characters
	Send login and password with the fields empty.
	Does not define the external interface
	There is no information about the implementation
Req_022	No information in the requirement
	The system may not have evidences realized in the day
	No experience in implementation of the required communication
	Send invalid files
	Send invalid data
	Does not define the external interface
	There is no definition of the amount of characters for each defined field
	There is no knowledge in the used technology for the development
Req_023	There is no information about the implementation
	There is no important information in the requirement
	The health agent is not the actor on the requirement
	There is no essential information for the implementation.
	No information about the security
	Data in the questionnaire can be invalid
	There is no security definition
	Insert invalid data in the questionnaire
	Show other different screen in the diagnostic

Activity 3: Definition of the correct risks

From the results of the activities 1 and 2 presented respectively in the Tables Table 4.16 and Table 4.21, the risks identified were analyzed by two evaluators with risk experts profile to realize the correct risk definition. As defined in the Section 4.2.1.4, in this study, correct risk is an intersection of risks identified by risk experts using the RITRA technique and ad hoc. The result is shown in the Table 4.22. This activity results in values for the metric RIE – Risks identified by Risk Expert – resulting from the intersection of the risks identified using the RITRA technique and ad hoc.

$$RIE = RIE_{TRAT} \cap RIE_{STR}$$

RIE_{TRAT} – Risks Identified by risk experts using the RITRA technique.

RIE_{STR} – Risks Identified ad hoc by risk experts .

Table 4.22. Intersection of the risks identified by Risk Expert (RIE)

Requirement	Intersection of the risks identified
Req_021	The requirement does not report the amount of characters that each attribute must have
	No information related to the size of the fields
	No information related to the data type
	No information related to the formatting
	Does not report if data validation will be needed
	Define the type of the sent file.
	The system can validate the user just with login
	No information about transaction security
	There is no information about the implementation
	Does not define the external interface
Req_022	Does not report how the data will be stored
	No information about transaction security
	Does not report if data validation will be needed
	No information related to the size of fields
	The requirement does not report the amount of characters that each attribute must have
	The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.
	No specification of the implementation details.
	Short time available for implementation
	Does not define the external interface
	Defines the type of the sent file.
Req_023	The requirement does not contain the essential information, such as items of the questionnaire.
	The information contained in the questionnaire can be improper, causing an invalidation of the questionnaire.
	No specification of the implementation details.
	No information about transaction security
	Invalid information can be passed, causing an inappropriate use of it.

Activity 4: Risks identified by test engineers using the RITRA technique

In this activity of scenario A, defined in the Table 4.9, a set of four test engineers without knowledge in the risks was used. The result is shown in Tables Table 4.23, Table 4.24, Table 4.25, Table 4.26, and Table 4.27. This activity results in the values of metrics RIL_{TRAT} – Risks identified by Test Engineers using the RITRA technique – and TIR_{TRAT} – Time identifying risks using the RITRA technique.

Table 4.23. Risks identified by Test Engineer 1 using the RITRA technique

Identified risks in Req_021
The requirement does not report the amount of characters that each attribute must have
The information passed can be incoherent with the type of data that the database will store.
The system can validate the user just with login
No information related to the size of fields
No information related to the data type
No information related to the formatting
Does not report if data validation will be needed
The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.
Identified risks in Req_22
Does not report if data validation will be needed
The requirement does not report the amount of characters that each attribute must have
No information related to the size of fields
The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.
Does not inform how the user's opinion through the form will be inserted in the system.
No information about transaction security
Define the type of the sent file.
Identified risks in Req_023
The available functionalities for each profile (Doctor and ADS) are not indicated. The sentence of the flux for the valid data is incomplete.
The requirement does not contain the essential information, such as items of the questionnaire.
The information contained in the questionnaire can be improper, causing an invalidation of the questionnaire.
Invalid information can be passed, causing an inappropriate use of it.
No definition of the criteria of the implementation
Does not define the external interface
The interval of time is not specified
Does not specify if all diagnostics will be sent to the server.

Table 4.24. Risks identified by Test Engineer 2 using the RITRA technique

Identified risks in Req_021
Does not report how the data will be stored
Does not inform how the user's opinion through the form will be inserted in the system.
The requirement does not report the amount of characters that each attribute must have
No information related to the size of fields
Identified risks in Req_22
No information about the security of the transaction
No specification of the implementation details.
Define the type of the sent file.
Does not report if data validation will be needed.
Identified risks in Req_023
The available functionalities for each profile (Doctor and ADS) are not indicated. The sentence of the flux for the valid data is incomplete.
The requirement does not contain the essential information, such as items of the questionnaire.
The information contained in the questionnaire can be improper, causing an invalidation of the questionnaire.
Invalid information can be passed, causing an inappropriate use of it.
The interval of time is not specified
Does not specify if all diagnostics will be sent to the server.
No experience in the implementation
No definition of the criteria of the implementation
Does not define the external interface

Table 4.25. Risks identified by Test Engineer 3 using the RITRA technique

Identified risks in Req_021
No information about transaction security
The requirement does not report the amount of characters that each attribute must have
No information related to the size of fields
The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.
Identified risks in no Req_22
Does not report how the data will be stored
Does not define the external interface
No information about transaction security
No specification of the implementation details.
Define the type of the sent file.
Does not report if data validation will be needed.
The requirement does not report the amount of characters that each attribute must have
No information related to the size of the fields
The system can receive text without coherence or a set of characters without content, causing storage of unnecessary information.
Identified risks in Req_023
The direction of the download is not clear. (If the download is from server or from mobile)
The requirement does not contain the essential information, such as items of the questionnaire.
The information contained in the questionnaire can be improper, causing an invalidation of the questionnaire.
Invalid information can be passed, causing an inappropriate use of it.
Does not specify if all diagnostics will be sent to the server.
No definition of the implementation criteria
Does not define the external interface
No information about transaction security

Table 4.26. Risks identified by Test Engineer 4 using the RITRA technique

Identified risks in Req_021	
The requirement does not report the amount of characters that each attribute must have	
No information related to the size of fields	
No information related to the data type	
No information about transaction security	
The requirement does not report the amount of characters that each attribute must have	
Identified risks in Req_022	
No information about transaction security	
Define the type of the sent file.	
Does not report if data validation will be needed.	
Identified risks in Req_023	
The requirement does not contain the essential information, such as items of the questionnaire.	
The information contained in the questionnaire can be improper, causing an invalidation of the questionnaire.	
Invalid information can be passed, causing an inappropriate use of it.	
The interval of time is not specified	
Does not specify if all diagnostics will be sent to the server.	
No information about transaction security	

Table 4.27. Time to identify risks using the RITRA technique by test engineers

	TE1	TE2	TE3	TE4	Averaged Time
Req_021	13	16	14	17	15
Req_022	10	7	9	10	9
Req_023	12	5	5	6	7

Legend: TE- Test Engineers

Activity 5: Risks identified ad hoc by Test Engineers

In this activity of scenario B, defined in the Table 4.10, was applied a set of four test engineers without knowledge in risks. The result is showed in Tables Table 4.28, Table 4.29, Table 4.30, Table 4.31, and Table 4.32. This activity results in the values of metrics RIL_{STR} – Risks identified ad hoc by Test Engineers – and TIR_{STRA} – Time identifying risks ad hoc.

Table 4.28. Risks identified ad hoc by test engineer 1

Identified risks in Req_021
The login button may not function
Identified risks in Req_22
Stability is not so concrete when the internal data processing is informed.
Information is missing when the requirement is presented in a specific way for a specialist in the area.
Identified risks in Req_023
There is no information about the items of the questionnaire. Where is filled this questionnaire? In the system?

Table 4.29. Risks identified ad hoc by test engineer 2

Identified risks in Req_021
The requirement needs more information
Identified risks in Req_22
The information presented is clear, but incomplete.
The requirement needs more information
Identified risks in Req_023
It is necessary to know how the questionnaire will be provided to the medical team. Will the health agent choose the questionnaire from a list of a models stored in the system, or the questionnaire should be generic and "with many text fields"?

Table 4.30. Risks identified ad hoc by test engineer 3

Identified risks in Req_021
The information passed can be incoherent with the data type that the database will store.
Allow validation of the user just for the login, being prevented by SQL injection.
Identified risks in Req_22
The evidences can be corrupted, thus not completing the sending of the information
The data may be with formats not accepted by the system, and the information may be generated with bad formatting and in a difficult way to understand.
Identified risks in Req_023
The system may refer another questionnaire out of the specified context.

Table 4.31. Risks identified ad hoc by test engineer 4

Identified risks in Req_021
The system may allow the access of unauthorized people.
Identified risks in Req_22
The requirement is associated with the risk of implementation. The requirement is not complete. The difficulty of the requirement may be raised, depending on the chosen communication between the systems.
The presented information is clear, but incomplete.
Identified risks in Req_023
Confused and unclear requirement. It may make the implementation harder.
The requirement is unclear, because there is ambiguity. It is not evident which actor is related to the requirement.

Table 4.32. Time to identify risks ad hoc by test engineers

	TE1	TE2	TE3	TE4	Average Time
Req_021	15	15	11	15	14
Req_022	16	10	15	15	14
Req_023	14	10	7	17	12

Legend: TE- Test Engineers

Activity 6: Number of risks identified using RITRA technique

In this activity, the number of correct risks found by test engineers using the RITRA technique, result of the activity 4 in comparison with activity 3, are shown in Table 4.33. This activity results in the values of the metric QRI_{TRAT} .

Table 4.33. Number of consistent identified risks using the RITRA technique

Requirement	RC	Number of consistent risks by test engineers							
		Test Engineer 1		Test Engineer 2		Test Engineer 3		Test Engineer 4	
		TO	CO	TO	CO	TO	CO	TO	CO
21	10	8	6	4	2	3	2	4	4
22	10	7	6	4	4	9	9	3	3
23	5	8	4	9	3	8	5	6	4

Legend: RC- Consistent risks by experts; TO- Risks identified; CO- Correcty risks.

Activity 7: Number of risks identified without RITRA

Activity 7 presents the number of correct risks found by test engineers without using RITRA, result of activity 5 in comparison with activity 3. The outcomes are shown in the Table 4.34. This activity results in the values of the metric QRI_{STRA} .

Table 4.34. Number of consistent identified risks without RITRA.

Requirement	RC	Number of consistent risks by test engineers							
		Test Engineer 1		Test Engineer 2		Test Engineer 3		Test Engineer 4	
		TO	CO	TO	CO	TO	CO	TO	CO
21	10	1	0	1	0	2	2	1	0
22	10	2	0	2	0	2	1	2	0
23	5	1	1	1	0	1	0	2	0

Legend: RC- Consistent risks by experts; TO- Risks identified; CO- Correcty risks.

4.2.5. Exploratory Data Analysis

From the values presented in Section 4.2.4, an exploratory analysis was performed to verify the dispersion of the sample data for each metric used in the study. Five descriptive measures will be analyzed: minimum, maximum, mean, median and standard deviation. The minimum value helps to observe the worst values from the sample data. The maximum value allows researchers to verify the number of consistent identified risks. The median is an equilibrium point of the sample data. It indicates the value that splits the sample in two halves and, finally, the standard deviation indicates the dispersion of data in the sample indicating how the results differ from the mean. The smaller the standard deviation value, the more the sample is homogeneous. In the following sub-sections the data values for the metrics QRITRAT, QRISTRA , TIRTRAT , TIRSTRA are presented.

4.2.5.1. QRI_{TRAT} versus QRI_{STRA}

In this section, the number of risks identified by test engineers using the RITRA technique (QRITRAT) and the number of risks identified ad hoc (QRISTRA) are analyzed. The purpose is to observe the risk identification behavior using the RITRA technique by test engineers who do not have knowledge in risks. For the analysis, the QRITRAT and QRISTRA metrics are divided in two indexes: (i) Identified risks' percentage index of coverage and (ii) Identified risks' percentage index of efficacy. In what follows we describe each of the percentage index, constructed from Tables Table 4.35 and Table 4.36. The percentage ranges from 0 (zero) to 1 (one).

i) Identified risks' percentage index of coverage

This index is calculated through the relation between the correct risks defined by test engineers and the correct risks defined by risk experts. The relation is presented in Equation ((4.2)):

$$I_{CR} = \frac{CO}{RC} \quad (4.2)$$

ii) Identified risks' percentage index of efficacy

This index is calculated through the relation between the correct risks defined by test engineers and the overall risks also defined by test engineers. The relation is presented in Equation ((4.3)):

$$I_{ER} = \frac{CO}{TO} \quad (4.3)$$

Each index will present two variations (1) Applied to the technique (ICRT e IERT) and (2) applied to ad hoc (ICRS e IERS).

Table 4.35 and Table 4.36 show the indexes that will be used as a sample for the analysis of descriptive data:

Table 4.35. Identified risks' percentage indexes of coverage and efficacy applying RITRA

ICRT	IERT
0,6	0,75
0,2	0,5
0,2	0,67
0,4	1
0,6	0,86
0,4	1
0,9	1
0,3	1
0,8	0,5
0,6	0,33
1	0,63
0,8	0,8

Table 4.36. Identified risks' percentage indexes of coverage and efficacy ad hoc

ICRS	IERS
0	0
0	0
0,2	1
0	0
0	0
0	0
0,1	0,5
0	0
0,2	1
0	0
0	0
0	0

Tables Table 4.37 to Table 4.40 were built with the descriptive data from the indexes using the RITRA technique and ad hoc.

Table 4.37. Descriptive data of coverage indexes using the RITRA technique

Metric	Minimum	Maximum	Mean	Median	Standard Deviation
ICRT	0,20	1,00	0,57	0,60	0,27

Table 4.38. Descriptive data of efficacy indexes using the RITRA technique

Metric	Minimum	Maximum	Mean	Median	Standard Deviation
IERT	0,33	1,00	0,75	0,78	0,23

Table 4.39. Descriptive data of coverage indexes ad hoc

Metric	Minimum	Maximum	Mean	Median	Standard Deviation
ICRS	0,00	0,20	0,04	0,00	0,08

Table 4.40. Descriptive data of efficacy indexes ad hoc

Metric	Minimum	Maximum	Mean	Median	Standard Deviation
IERS	0,00	1,00	0,21	0,00	0,40

Figure 4.2 shows the descriptive data behavior of the identified risks' percentage indexes of coverage using the RITRA technique and ad hoc and, Figure 4.3 shows the descriptive data behavior of the identified risks' percentage of indexes of efficacy using the RITRA technique and ad hoc.

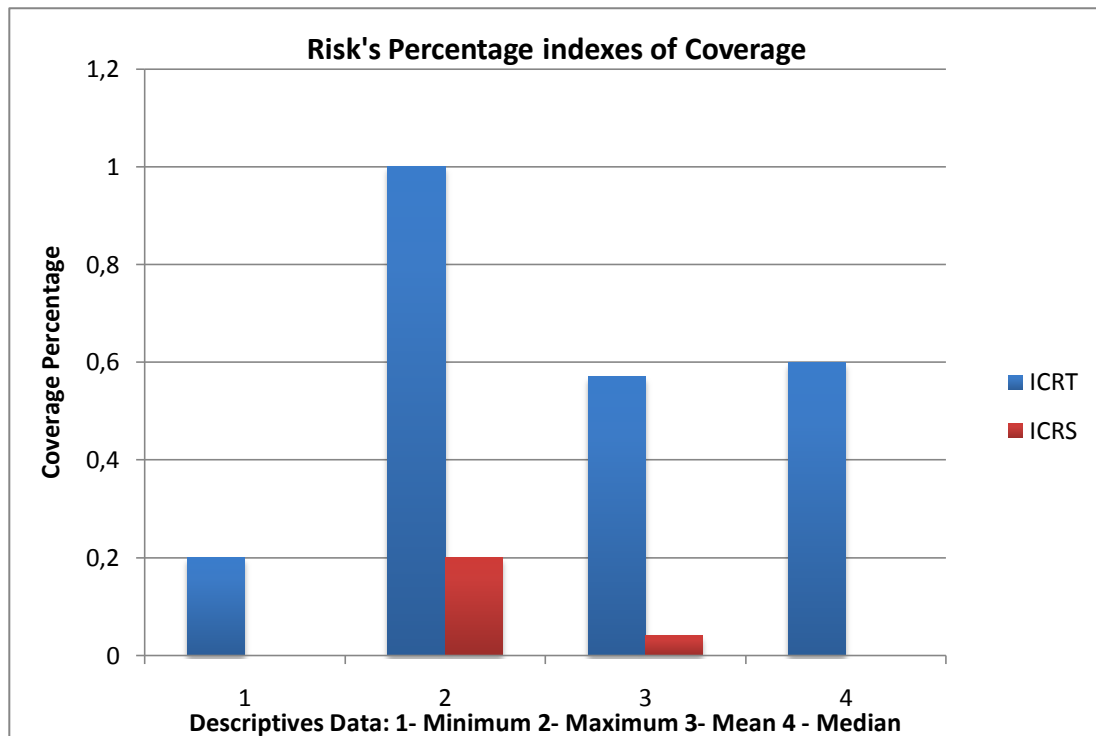


Figure 4.2. Comparison of identified risk's percentage indexes of coverage

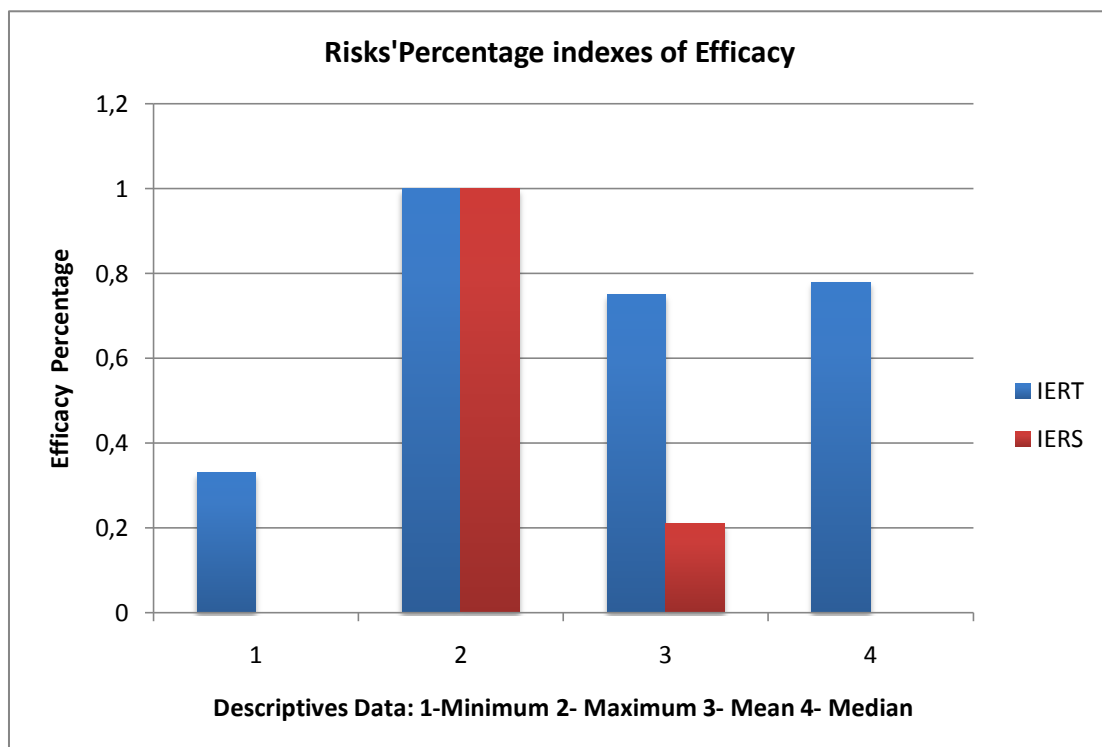


Figure 4.3. Comparison of Identified risks' percentage indexes of efficacy.

After the analysis of the percentage of indexes, it is observed in both cases that results obtained in this experiment using the RITRA technique application were more significant than using ad hoc. The closer to 1(one), the

better, thus, the QRI_{TRAT} metric presented percentage indexes of coverage and efficacy in the risks identification greater than the QRI_{STRA} metric.

4.2.5.2. TRI_{TRAT} versus TRI_{STRA}

In this Section is analyzed the time (minutes) taken to identify the risks with RITRA (TRI_{TRAT}) and the time (minutes) taken to identify the risks without it (TRI_{STRA}). Table 4.41 presents the data.

Table 4.41. Time descriptive data for the risk identification

Metric	Minimum	Maximum	Average	Median	Standard Deviation
TRI_{TRAT}	5	17	10,3	10	4,1
TRI_{STRA}	7	17	13,3	15	3,1

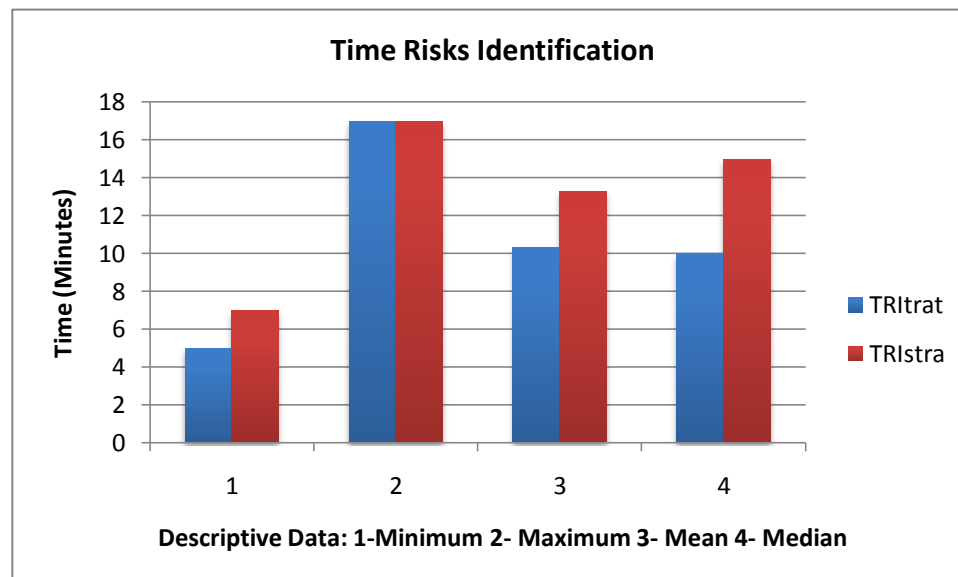


Figure 4.4. Comparison of time taken for the identification of risks

In this Experiment, the time taken to identify risks with the RITRA technique (TRI_{TRAT}) presents better results (less time taken) than identifying risks ad hoc (TRI_{TRAT}).

4.2.6. General Considerations and Difficulties

In this experiment, the proposed activities have been executed successfully. Within the set of samples, this pilot experiment was able to assess the RITRA behavior when used by professionals without risk knowledge. The results can

be considered satisfactory. However, some aspects must be improved. In Chapter 5 these aspects will be presented as lessons learned.

Test engineers classified the experience of using RITRA in the experiment regarding the time spent in the risk identification activity as normal and acceptable. The execution of the experiment presented some difficulties, however, considering that the test engineers did not have experience in risks, the RITRA technique performed well. According to them, without the help from the technique they would not be able to identify most of the risks listed due to the lack of knowledge in risks. Since the technique was performed using an MS Excel™ spreadsheet, many of the participants found it very uncomfortable because of the repetition of steps, suggesting the automation of the technique. They observed a lack of visual information in the script, and since they did not have experience on how to identify risks, they found it a bit of difficult and uncertain about the final selection of risk for each requirement.

The risk experts' perception regarding the time spent in the activity performed with RITRA was classified by them as acceptable. The task presented some difficulties regarding the use of MS Excel™ spreadsheets and also suggested the automation of the technique. Experts indicated the applicability of the technique in small, medium and large projects in companies as long as the technique presents tool support.

CONCLUSIONS

This Chapter presents the scope of this work, main contributions, threats to validity and indication for future works.

After some research on techniques for risk identification applied to software testing, we observed a lack of methods, techniques and tools applied to this area. We observed that testing is such an important activity for software quality, however, tests engineers usually do not have much time to perform it. Usually, software failures are detected through the execution of test cases. In general, test cases are built from software requirements. Even though tests engineers execute test cases, they have many difficulties in choosing them. The ideal scenario would be to choose test cases that enable finding as many errors as possible (ideally all). There are several ways to prioritize tests cases, but if the requirements' risks are already known, test engineers could build test cases based on more critical requirements' risks. Nonetheless, in general, test engineers do not have knowledge in risks area.

In this context, we proposed a technique that helps test engineers in the task of identifying risk from requirements. The technique is based on reuse of risks previously identified from other requirements. Requirements are categorized by attributes to identify and populate a database, which allows their retrieval according to a similarity analysis. A pilot experiment was executed to assess the technique's behavior when used by test engineers. An exploratory analysis was performed on descriptive data. Although this analysis indicated good results, it cannot be generalized due to the small sample size.

It is important to note that the results from the identified risks by the test engineers tightly coupled to the quality of risks already present in the database. It is important to observe that it was not possible to conclude that RITRA is better at finding risks than an expert. The technique is not able to identify new risks on requirements, but it is very helpful at identifying risks that are present in the database and associated to related requirements.

5.1. RESEARCH CONTRIBUTION

The RITRA technique proposed advises test engineers, who usually do not have knowledge on risk management to identify risks from software requirements. Risk identification on software requirements contributes to build test cases based on risks, increasing the likelihood of finding software failures. This work presented some contributions:

Mapping Study: We described, through the mapping study, the state of the art in techniques, methods and tools for risk identification applied to software testing. We verified some gaps on the returned studies and also on the needs of test engineers in the currently used risk identification techniques. We also observed the lack of knowledge in the risks from such engineers.

RITRA: In this work we proposed a risk identification technique to help non-risk-aware test engineers. There are many attributes associated to requirements, but only a subset of them was selected for categorizing. We proposed the use of two similarity calculation algorithms to compare similarity among requirements. The modeling technique was BPMN, and the tool used was Bizagi.

Evaluation: In this work two assessments were performed. First, a comparison between two similarity algorithms: Sequential and Euclidean Distance. After this, we observed that even though the two algorithms returned almost the same similar requirements, the Euclidean distance algorithm presented more homogeneity with all attributes. Others algorithms might be used with this technique and it can be improved associating weights on the attributes. Next, a test engineers took part in a pilot experiment. Although the results cannot be generalized due to sample size, the experiment verified the requirements attributes influence, by using the similarity algorithm and it sowed a good indication to analogy.

Final product: After evaluation some contributions were detected by experiment participants. They are:

1. Ability to systematically identify risks on requirements based on other requirements that have been previously identified;

2. Ability to identify action plans that address the same problems for several requirements;
3. Accelerating the process of identifying risks in recent requirements;
4. Reduction of time spent in identifying risks in requirements;
5. Use of time and effort in identifying new risk requirements.

5.2. THREATS TO VALIDITY

During the study execution some gaps were identified due to inability to say that the proposed technique is efficient. They are described below:

Risk List: Although the risk experts participated in the first part of the evaluation, most of them had little knowledge and experience in the activity of identifying risks from requirements. The risk list defined by the experts present some flaws due to the lack of a careful assessment and filtering that were not performed on the answers given by experts. For example, some risks were conflicting, others cannot be considered risks at all.

Database requirements with identified risks: During this study there was not a database of requirements with identified risks. Then the base was built from the analysis of risk experts. Because of this, the number of requirements with the risks identified was small. The ideal is to build an actual database populated with risks identified in the requirements from several projects.

Manual technique: The technique was employed manually using MS Excel™, creating discomfort to participants, generating unwanted effects that might have influenced the results.

Statistics: The results of this pilot experiment cannot be generalized due to the undersized sample. The results used in the analysis were exploratory, i.e., analyzing only the data of descriptive metrics: minimum, maximum, mean, median and standard deviation.

5.3. FUTURE WORK

For replications of this study some points were identified and must be improved:

Statistics: It is suggested that the study is applied to larger groups of risk experts and non risk-aware test engineers and it is indicated the use of statistical tests to perform inferences on the results so that the results obtained from the sample can be generalized.

Define a Structure of Risk Factors – To use a database of risks is important to define an unique vocabulary structure of risk terms. To use analogy based techniques, it is necessary to develop an accurate data base, where requirements and their respective risks found will be available. For this, it is necessary to develop a risks factors framework for requirements, ensuring that the risk checklist follow a descriptors standard and generic nomenclature.

Automation of the technique: For better technique performance it shall be implemented in the form of a tool. Some suggestions for the RITRA implementation were given in Chapter 3.

Categorization of requirements: The categorization has general requirement's attributes, suggesting a study to cast a new set of attributes to more specific requirements in order to better filter the search for similar requirements.

Add features to the technique:

1. The current approach does not take into account the requirements' functional information.
2. A search algorithm to find similar requirements by keywords. This way, besides the requirements' attributes, their content (text) is considered.

Update Systematic Mapping study (SMS): Its important to update the SMS in further research to identify new techniques developed. The search string should be improved in order to find more results of risk identification techniques applied to software testing.

REFERENCES

[Astah 2011] Modelling tool astah Official Page. Available at <<http://astah.net/>> Acessed: 22 december 2011.

[Bach 1995] Bach, J. The Challenge of Good Enough Software. In: American Programmer, 1995.

[Biswas et al. 2011] Biswas, R.; Ort, E. The Java Persistence API - A Simpler Programming Model for Entity Persistence. Available at <<http://java.sun.com/developer/technicalArticles/J2EE/jpa/>> Acessed: 22 December 2011.

[BizAgi 2011] Modelling tool Biz Agi Official Page. Available at <<http://www.bizagi.com/>> Acessed: 22 december 2011.

[Brocke et al. 2010] Brocke, V.J., Rosemann, M., Handbook on Business Process Management: Strategic Alignment, Governance, People and Culture (International Handbooks on Information Systems) (Vol. 1). Berlin: Springer, 2010.

[Calhoun et al. 2000] Calhoun, Cynthia C. Identifying and Managing Risks for automatic Test Systems, IEEE AES Systems Magazine, 2000.

[Carr et al. 1993] Carr, M. J., Konda, S.L., Monarch, I., Ulrich, F.C., Walker, C.F. Taxonomy Based Risk identification. Tecnical Report CMU/SEI-93-TR-6. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.USA, 1993.

[Cornell et al. 2000] Cornell, G.; horstman, S.; Core Java 2: Fundamentos. 1a edição, Makron Books, 2000.

[Dorofee et al. 1996] Audrey Dorofee et al. Continuous Risk Management Guidebook. Carnegie Mellon University, Software Engineering Institute, 1996.

[Dustin et al. 1999] Dustin, E.; Rashka, J.; Paul, J. "Automated Software Testing: Introduction, Management, and Performance." New York: Addison-Wesley, 1999.

[Dybå 2008] Tore Dybå and Torgeir Dingsøyr. Empirical studies of agile software development: A systematic review. IST, 50, 2008.

[Eclipse 2011] Eclipse Project. available at <<http://www.eclipse.org/>>. Accessed: 22 Dezembro de 2011.

[Fabbrini et al. 2001] Fabbrini, F.; Fusani, M.; Gnesi, Stefania.; Lami, Giuseppe. An automatic quality evaluation for natural language requirements. In: Proceedings of the 7th International Workshop on Requirements Engineering: Foundation for Software Quality REFSQ'01, 2001.

[Fantechi et al. 2003] Fantechi, A.; Gnesi, S.; Lami, G.; Maccari, A. Applications of linguistic techniques for use case analysis, Requirements Engineering, Vol. 8, No. 3, pp 161-170, 2003.

[Firesmith 2003] Firesmith, D. Specifying Good Requirements, Journal of Object Technology, Vol. 2, No. 4, pp 77-87, Jul-Aug 2003.

[Grosskopf et al. 2009] Grosskopf, Alexander., Decker, Gero., Weske, Mathias., The process: Business Process Modeling using BPMN, 1st ed. Meghan-Kiffer Press, Tampa, Florida, USA, 2009.

[Goldsmith 2006] Goldsmith, R. Early and Effective: The Perks of Risk-based Testing. In: Software Test and Performance Magazine, v.3, p.24-30, 2006.

[Houaiss 2001] Houaiss, A. Dicionário eletrônico Houaiss da língua portuguesa . Rio de Janeiro: Objetiva. Versão 1.0. 1 [CD-ROM]. 2001.

[Hazan et al. 2003] Hazan, C. L., Leite, J. C. S. P., Indicadores para a Gerencia de Requisitos, In WER 2003 - VI Workshop em Engenharia de Requisitos, São Paulo, Brazil, 2003.

[Higgins 1994] Higgins, James M, Creating Creativity, Training & Development, 48(11), pg11-15, 1994.

[IEEE 1990] IEEE Std 610.12-1990. “IEEE Standard Glossary of Software Engineering Terminology”. New York, USA: The Institute of Electrical and Electronics Engineers. Sep 1990.

[IEEE 1998] IEEE Std 830-1998. “IEEE Recommended Practice for Software Requirements Specifications”. New York, USA: The Institute of Electrical and Electronics Engineers. Jun 1998.

[ISO 31000 2009] ABNT NBR ISO 31000:2009. Gestão de riscos - Princípios e diretrizes, 2009.

[Jacobson et al. 1998]Ivar Jacobson, Grady Booch, James Rumbaugh. The Unified Software Development Process. Addison Wesley, 1998.

[Kendall et al. 2007]Kendall, Richard P., et al. A Proposed Taxonomy for Software Development Risks for High-Performance Computing (HPC) Scientific/Engineering Applications. Pittsburgh, PA: Software Engineering Institute, Carnegie-Mellon University. USA, 2007.

[Kitchenham et al. 2007] Barbara Kitchenham and Stuart Charters. Guidelines for performing Systematic Literature Reviews in Software Engineering. Keele and Durham University Joint Report, 2007.

[Kohlbacher 2009] M. Kohlbacher. The Effects of Process Orientation on Customer Satisfaction, Product Quality and Time-Based Performance. Paper presented at the 29th International Conference of the Strategic Management Society, Washington DC, October 11–14, 2009.

[Kruchten 1998] Philippe Kruchten. The Rational Unified Process – an Introduction, ISBN: 0-201-60459-0. Addison-Wesley, 1998.

[Lins 2007] Lins, A. V. Um modelo para identificação de riscos de projeto utilizando raciocínio baseado em casos – Trabalho de conclusão de curso, Universidade de Pernambuco, 2007.

[Lumsdain et al. 1990] Edward Lumsdain, Monica Lumsdain. Creative Problem Solving. McGraw-Hill, New York, 1990.

[Martins 2000] Martins, A. Computação Baseada em Casos: Contribuições Metodológicas aos modelos de Indexação, Avaliação, Ranking, e similaridade de casos. Tese de Doutorado em Engenharia Elétrica, Universidade Federal da Paraíba, 2000.

[Mora et al. 2003] Mora, M.; Denger, C. Requirements Metrics - An initial literature survey on measurement approaches for requirement specifications. IESE-Report No. 096.03/E Fraunhofer Institut Experimentelles Software Engineering, Oct 2003.

[Myers 2004] Glenford J. Myers, The Art of Software Testing, 2^a Ed. John Wiley & Sons, Inc., Hoboken, New Jersey, 2004.

[MySQL 2011]. MySQL Data Base Page official. Available at <<http://www.mysql.com>>. Accessed: 22 December 2011.

[Nannini 2010] Nannini, Paulo. Uma taxonomia de Requisitos, Test White Paper, Instituto Brasileiro de qualidade em testes de software, 2010.

[Newell et al. 2002] Newell, M.W., PMP, ENP, Preparing for the project management professional (PMP) certification exam. 2nd ed, 2002.

[PMI 2004] PMI – Project Management Institute. A guide to the project management body of knowledge (PMBOK Guide). Upper Darby, PA, 2004.

[Preesman 2009] Roger Pressman. Software Engineering: A Practitioner's Approach. 9th Ed. McGraw-Hill Higher Education, 2009.

[Qualiti 2008] Qualiti Software Processes, Minicurso de Introdução a testes de software, Brasil, 2008.

[SEI 1993] Key Practices of the Capability Maturity Model for Software, Version 1.1, Document No.CMU/SEI-93-TR-25,1993.

[Sommeville 2004] Sommerville, Ian. Software Engineering. 7th Edition, 2004.

[Souza et al. 2008] Souza, E. e Gusmão C. RBTPProcess - Modelo de Processo de Teste de Software baseado em Riscos. 13º Workshop de Teses e Dissertações em Engenharia de Software, 2008.

[Souza et al. 2009] Ellen Souza, Cristine Gusmão, Keldjan Alves, Júlio Venâncio, Renata Melo “Measurement and Control for risk- based test cases and activities”, IEEE Software, 2009.

[Souza et al. 2010] Ellen Souza, Cristine Gusmao, Júlio Venâncio , Risk-Based Testing: A Case Study, Seventh International Conference on Information Technology, 2010.

[Telelogic 2006] Telelogic. Get It Right the first Time: Writing Better Requirements, 2006.

[Trigo et al. 2007] TRIGO, T. R. ; GUSMÃO, C. M. G. Avaliando um modelo de identificação de Projetos de Software similares – Trabalho de Conclusão de curso, Universidade de Pernambuco, 2007.

[Vera et al. 2007] Vera, A. & Kuntz, L. Process-based organization design and hospital efficiency. Health Care Management Review, 32(1): p.55-65.,2007.

[Wiegers 2003] Wiegers, Karle. Software Requirements 2h Edition, 2003.

[Wilson et al. 97] Wilson, W.; Rosenberg, L.; Hyatt, L. Automated Analysis of Requirement Specifications. In: Proceedings of the Nineteenth International Conference on Software Engineering (ICSE-97), Boston, May 1997.

[Wikipédia 2012] Wikipédia, available at
<http://en.wikipedia.org/wiki/Business_process_management> Accessed: 22 December 2011.

APPENDIX A

Artifacts developed and used in this work.

A1- Template for Attributes classification spreadsheet

ID Requirement	Correctness	Stability	Difficulty	Ambiguity	Completeness
<Id requirement>	<Value>	<Value>	<Value>	<Value>	<Value>
Legend: (1) Very High (2)High (3)Average (4)Low (5) Very Low/Absent*					
* Used just for ambiguity					

A2- Template for risks requirements

ID	Name	Description	Attributes		Risks
<Id requirement>	Requirement Name	Requirement description	Correcteness	<value>	Risk 1
			Stability	<value>	Risk 2
			Difficuty	<value>	Risk n
			Ambiguity	<value>	
			Completeness	<value>	
Legend: (1) Very High (2)High (3)Average (4)Low (5) Very Low/Absent*					
* Used just for ambiguity					

A3- Risks Requirements Description

Correcteness		Attribute that defines if the requirements reflect the real intentions of the client, i.e., if the requirements satisfy the real necessities expressed by him/her. If the requirement does not represent the necessity of the user, this attribute can generate great risks regarding validity, enabling the delivery of a product that is different from client expectations.
Classification	Value	Classification Definition
Very High	1	The requirement presents all the client's real needs
High	2	The requirement presents the client's main needs
Average	3	The requirement presents some of the client's necessities
Low	4	The requirement presents few client's needs
Very Low	5	The requirement presents none of client's needs

Stability		Attribute that defines the degree of stability of the requirement.
Classification	Value	Classification Definition
Very High	1	Remote probability of change;
High	2	Low probability of change
Average	3	There is no indicator of change in the requirement.
Low	4	It is possible that the requirement changes.
Very Low	5	It is highly probable that the requirement changes.

Difficulty		Attribute that defines the level of effort to implement the requirement
Classification	Value	Classification Definition
Very High	1	Far above the average of effort to implement a requirement
High	2	Above average effort to implement the requirement
Average	3	Average effort to implement the requirement
Low	4	Below average effort to implement the requirement
Very Low	5	Far below the average effort to implement a requirement

Ambiguity		Attribute that defines how much the requirement admits more than one interpretation. The ambiguity is related to understanding the requirement. Requirements that demand interpretation or are not clear, presents indicators of clarity risks. The writing of the requirement can make dubious sense or different from the real intention.
Classification	Value	Classification Definition
Very High	1	Requirement is totally misunderstood.
High	2	Requirement admits many interpretations
Average	3	Requirement admits more than one interpretation
Low	4	Requirement asks for effort in the interpretation of the reading
Absent	5	Requirement is totally understood

Completeness		Attribute that defines the degree of completeness of the requirement. Incomplete requirements do not reflect the real intentions of the user and will have to be modified in a posterior moment. Complete requirements increase the quality of codification, thus helping the fulfillment of the deadline, correct execution under the budget and project's resources. Incomplete requirements make more difficult the test execution and the code verification.
Classification	Value	Classification Definition
Very High	1	Requirement has all necessary information
High	2	Requirement has just important information
Average	3	Requirement has just the essential information
Low	4	Requirement has little information
Very Low	5	Requirement has no relevant information

A4: New Requirements

ID	Name	Correctness	Stability	Difficulty	Ambiguity	Completeness
Req_21	Autenticar Usuário	1	1	4	5	2
V1.5_RF001 Prioridade: Essencial Perfil: Médico Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha). Saídas e pós-condições: Usuário é autenticado no sistema e o sistema apresenta a tela inicial (menu principal do aplicativo). Fluxo de eventos Fluxo Principal 1-O usuário preenche o formulário de autenticação com nome de usuário e senha; 2-O usuário pressiona o botão “login”. Fluxo Alternativo 1: Dados válidos 1-Com os dados validados pelo sistema, o usuário é autenticado no sistema e automaticamente será apresentado o menu principal do aplicativo móvel. Fluxo Alternativo 2: Dados inválidos 1-Caso os dados de entrada sejam inválidos, aparece uma mensagem de erro e a tela de autenticação é exibida novamente.						
ID	Name	Correctness	Stability	Difficulty	Ambiguity	Completeness
Req_22	Enviar evidencias	2	3	2	5	3
V1.5_RF002 Prioridade: Essencial Perfil: Médico Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Upload de evidências do dispositivo móvel para o servidor intermediário, banco de dados do dispositivo e servidor alterados. Fluxo de eventos Fluxo Principal 1-Usuário seleciona a opção “enviar evidências” no menu principal do aplicativo; 2-O dispositivo se conecta ao servidor intermediário e faz o upload dos diagnósticos realizados no expediente; 3-O dispositivo móvel apaga os diagnósticos enviados ao servidor; 4-Durante o processo de upload é exibida uma mensagem informando o progresso da operação.						
ID	Name	Correctness	Stability	Difficulty	Ambiguity	Completeness
Req_23	Efetuar Diagnostico	2	2	3	3	4
V1.5_RF003 Prioridade: Essencial Perfil: Médico Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Diagnóstico do paciente de acordo com as respostas fornecidas ao questionário. Fluxo de eventos Fluxo Principal 1-Usuário seleciona a opção “diagnóstico” no menu principal do aplicativo; 2-O agente de saúde preenche o questionário definido pela equipe médica; 3-Ao final do preenchimento, o usuário pressiona o botão “OK”; 4-O sistema exibe o diagnóstico. Fluxo Alternativo 1: O usuário não preenche completamente o questionário 1-Caso o usuário não responda a alguma(s) pergunta(s) o sistema irá exibir uma mensagem solicitando que o usuário preencha completamente o questionário.						

A5: Data base spreadsheet template

B2: Data Base Spreadsheet

Version	1.5	Real ID	RF004	ID	REQ_01	Name	Atualizar Arvore		
Correctness	2	Stability	2	Difficulty	2	Ambiguity	5	Completeness	2
Description									
Prioridade: Essencial Perfil: Médico Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: <i>Download</i> de script inteligente (árvore de decisão) do servidor intermediário para o dispositivo móvel, banco de dados do dispositivo e servidor alterados. Fluxo de eventos Fluxo Principal 1- Usuário seleciona a opção “atualizar árvore” no menu principal do aplicativo; 2- O dispositivo móvel se conecta ao servidor intermediário e faz o <i>download</i> da árvore de decisão atualizada; 3- O dispositivo móvel apaga a árvore local e adiciona a árvore atualizada. 4- Durante o processo de <i>download</i> é exibida uma barra informando o progresso da operação.									
Risks									

Version	1.5	Real ID	RF005	ID	REQ_02	Name	Armazenar Evidências		
Correctness	2	Stability	2	Difficulty	2	Ambiguity	5	Completeness	2
Description									
Prioridade: Essencial Perfil: Médico Entradas e pré-condições: Usuário autenticado no sistema e na tela de resultado de diagnóstico. Saídas e pós-condições: Banco de dados do dispositivo móvel alterado. Fluxo de eventos Fluxo Principal 1- O sistema exibe o resultado do diagnóstico e pergunta ao usuário se ele concorda ou não com o resultado informado pela árvore de decisão. 2- O usuário entra com sua opinião acerca do resultado apresentado pela árvore de decisão e pressiona o botão “OK”. 3- O sistema armazena o resultado do diagnóstico, o questionário preenchido e a opinião do usuário sobre o resultado fornecido pela árvore no banco de dados do dispositivo móvel. Fluxo Alternativo 1: O usuário concorda com o resultado da árvore Caso o usuário concorde com o resultado do sistema ele deverá selecionar a opção “sim” e pressionar o botão “OK”. Fluxo Alternativo 2: O usuário não concorda com o resultado da árvore Caso o usuário não concorde com o resultado do sistema ele deverá escolher a opção “não” e justificar sua opinião; Após emitir sua opinião o usuário deve pressionar o botão “OK”. Fluxo Alternativo 3: O usuário não informa se concorda ou não com o resultado da árvore Caso o usuário não informe se concorda ou não com o resultado apresentado pela árvore e pressione o botão “OK”, o sistema irá exibir uma mensagem solicitando que o usuário emita sua opinião.									
Risks									

Version	1.5	Real ID	RF006	ID	REQ_03	Name	Gerenciar Evidências		
Correctness	3	Stability	4	Difficulty	2	Ambiguity	3	Completeness	5
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Não possui.</p> <p>Saídas e pós-condições: Informações de evidências inseridas/excluídas do servidor intermediário.</p> <p>Fluxo de eventos principal</p> <p>O servidor intermediário, ao receber requisições do dispositivo móvel, armazena ou exclui uma evidência.</p>									
Risks									

Version	1.5	Real ID	RF007	ID	REQ_04	Name	Gerenciar Arvores		
Correctness	3	Stability	3	Difficulty	1	Ambiguity	2	Completeness	5
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Não possui.</p> <p>Saídas e pós-condições: Informações de árvores inseridas/alteradas/excluídas do servidor intermediário.</p> <p>Fluxo de eventos principalO servidor intermediário trata as requisições recebidas com o intuito de inserir uma nova árvore, alterar uma já existente que possui ou não alguma inconsistência ou excluir uma árvore considerada precária.</p>									
Risks									

Version	1.5	Real ID	RF008	ID	REQ_05	Name	Autenticar Administrador		
Correctness	1	Stability	1	Difficulty	3	Ambiguity	5	Completeness	1
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha).</p> <p>Saídas e pós-condições: Acesso a servidor intermediário liberado com sucesso.</p> <p>Fluxo de eventos principal O administrador preenche o formulário de autenticação com nome de usuário e senha.</p> <p>Fluxo Alternativo 1: Dados válidos Com os dados validados pelo sistema, o usuário é autenticado e automaticamente será apresentado o menu principal do servidor intermediário.</p> <p>Fluxo Alternativo 2: Dados inválidos Caso os dados de entrada sejam inválidos, aparece uma mensagem de erro e a tela de autenticação é exibida novamente.</p>									
Risks									

Version	1.5	Real ID	RF009	ID	REQ_06	Name	Gerenciar Usuários		
Correctness	2	Stability	2	Difficulty	3	Ambiguity	5	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Usuário autenticado no sistema.</p> <p>Saídas e pós-condições: Informações de usuários do servidor intermediário incluídas/removidas/alteradas.</p> <p>Fluxo de eventos principal O administrador acessa tela de gerência de usuários, onde poderá incluir, modificar ou excluir os mesmos. Para o cadastro, os seguintes dados são necessários: nome, login, senha, CPF e tipo de usuário (administrador, médico ou agente de saúde).</p>									
Risks									

Version	1.3	Real ID	RF001_1	ID	REQ_07	Name	Autenticar Profissional de Saúde		
Correctness	1	Stability	1	Difficulty	3	Ambiguity	3	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Perfil: Médico e ACS</p> <p>Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha). Dispositivo conectado ao servidor.</p> <p>Saídas e pós-condições: Usuário autenticado no sistema e download dos dados dos pacientes que serão consultados, formulário e artefato de IA.</p> <p>Fluxo de eventos principal Usuário preenche o formulário de autenticação com nome de usuário e senha. Após, há dois fluxos: dados válidos ou inválidos.</p> <p>Dados válidos Com os dados validados pelo sistema, o usuário é autenticado no sistema e automaticamente será feito o download dos formulários de doenças e artefatos de IA – disponível no servidor – para o dispositivo móvel.</p> <p>Dados inválidos Caso os dados de entrada sejam inválidos, a tela de autenticação é exibida novamente.</p>									
Risks									

Version	1.3	Real ID	RF002_1	ID	REQ_08	Name	Encerrar Sistema		
Correctness	2	Stability	3	Difficulty	2	Ambiguity	3	Completeness	2
Description									
<p>Prioridade: Essencial</p> <p>Perfil: Médico e ACS</p> <p>Entradas e pré-condições: Usuário autenticado no sistema.</p> <p>Saídas e pós-condições: Upload de diagnósticos do dispositivo móvel para o servidor intermediário, banco de dados do dispositivo e servidor alterados, sistema desligado.</p> <p>Fluxo de eventos principal Usuário seleciona a opção “encerrar sistema”. O dispositivo se conecta ao servidor, faz o upload dos diagnósticos realizados no expediente, apaga os artefatos de IA e os registros do expediente, realiza <i>logout</i> do usuário e, por fim, desliga o sistema.</p>									
Risks									

Version	1.3	Real ID	RF003_1	ID	REQ_09	Name	Iniciar Diagnostico		
Correctness	3	Stability	3	Difficulty	2	Ambiguity	2	Completeness	3
Description									
Prioridade: Essencial									
Perfil: Médico									
Entradas e pré-condições: Usuário autenticado no sistema.									
Saídas e pós-condições: Banco de dados do dispositivo móvel alterado.									
Fluxo de eventos principal									
O agente de saúde preenche o questionário definido pela equipe médica. Ao final, sistema armazena o resultado do diagnóstico e o questionário preenchido. O sistema ao final do preenchimento do questionário e antes de armazenar a consulta, perguntará ao usuário se o resultado informado pela árvore é aceitável. Caso os dados fornecidos não sejam suficientes para gerar um diagnóstico o sistema deverá alertar o usuário.									
Risks									

Version	1.3	Real ID	RF003_2	ID	REQ_10	Name	Verificar Atualizações com Servidor Inteligente		
Correctness	3	Stability	3	Difficulty	1	Ambiguity	2	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Não possui.</p> <p>Saídas e pós-condições: Relação de formulário de doenças e artefatos de IA disponíveis</p> <p>Fluxo de eventos principal</p> <p>A cada intervalo de tempo a aplicação se comunica com o servidor inteligente e verifica a existência de formulários de doenças e artefatos de IA disponíveis.</p>									
Risks									

Version	1.3	Real ID	RF004_2	ID	REQ_11	Name	Atualização de formulários de Doenças e Artefatos de IA		
Correctness	3	Stability	4	Difficulty	1	Ambiguity	3	Completeness	4
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Relação de formulário de doenças e artefatos de IA disponíveis (RF004).</p> <p>Saídas e pós-condições: Atualização dos formulários de doenças e dos artefatos de IA.</p> <p>Fluxo de eventos principal O servidor Intermediário compara a versão entre os formulários de doenças e artefatos de IA disponíveis. Caso exista versão mais nova disponível o servidor Intermediário faz requisição da mesma. (E se existir um formulário cuja versão antiga já foi preenchida?)</p>									
Risks									

Version	1.3	Real ID	RF005_2	ID	REQ_12	Name	Transferir informações para o servidor Inteligente		
Correctness	2	Stability	3	Difficulty	2	Ambiguity	2	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Base de dados modificada e conexão entre servidores estabelecida.</p> <p>Saídas e pós-condições: Transferência de dados de pacientes e diagnósticos realizados efetuada com sucesso.</p> <p>Fluxo de eventos principal A cada intervalo de tempo a aplicação enviará todos os diagnósticos realizados para o servidor inteligente.</p>									
Risks									

Version	1.3	Real ID	RF001_3	ID	REQ_13	Name	Autenticar Administrador		
Correctness	3	Stability	1	Difficulty	4	Ambiguity	4	Completeness	2
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha).</p> <p>Saídas e pós-condições: Acesso a servidor intermediário liberado com sucesso.</p> <p>Fluxo de eventos principal</p> <p>O administrador preenche o formulário de autenticação com nome de usuário e senha. Após, há dois fluxos: dados válidos ou inválidos.</p>									
Risks									

Version	1.3	Real ID	RF002_3	ID	REQ_14	Name	Gerenciar Usuários		
Correctness	5	Stability	1	Difficulty	4	Ambiguity	4	Completeness	2
Description									
<p>Prioridade: Essencial E</p> <p>Entradas e pré-condições: Usuário autenticado no sistema.</p> <p>Saídas e pós-condições: Informações de médicos, ACSs e/ou usuários do servidor intermediário incluídas/removidas/alteradas. Para cadastro, os seguintes dados são necessários: Login, senha, CPF, Unidade e data nascimento.</p> <p>Fluxo de eventos principal</p> <p>O administrador acessa tela de gerência de usuários, onde poderá incluir novos usuários, modificar informações, ou então excluí-las.</p>									
Risks									

Version	1.3	Real ID	RF003_3	ID	REQ_15	Name	Gerenciar Visitas		
Correctness	4	Stability	5	Difficulty	4	Ambiguity	3	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Usuário autenticado no sistema.</p> <p>Saídas e pós-condições: Agendamento de visitas a pacientes são incluídos/removidos/modificados.</p> <p>Fluxo de eventos principal O administrador acessa tela de visitas. Cada visita contém uma data e está associada a um profissional de saúde e a um paciente. Sistema poderá informar o status de cada visita (desejável).</p>									
Risks									

Version	1.3	Real ID	RF004_3	ID	REQ_16	Name	Gerenciar Pacientes		
Correctness	2	Stability	2	Difficulty	3	Ambiguity	4	Completeness	2
Description									
Prioridade: Essencial Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Informações de pacientes incluídas/removidas/alteradas. Fluxo de eventos principal O administrador acessa tela de gerência de pacientes, onde poderá incluir novos pacientes, modificar informações sobre os agentes existentes ou então excluí-las.									
Risks									

Version	1.3	Real ID	RF005_3	ID	REQ_17	Name	Consultar pacientes		
Correctness	1	Stability	1	Difficulty	2	Ambiguity	3	Completeness	4
Description									
Prioridade: Desejável Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Informações de pacientes visualizadas. Fluxo de eventos principal O administrador acessa tela de pacientes selecionados e agrupados segundo algum critério, onde poderá visualizar os pacientes existentes na base.									
Risks									

Version	1.3	Real ID	RF005_4	ID	REQ_18	Name	Produzir Artefatos de IA		
Correctness	5	Stability	5	Difficulty	2	Ambiguity	2	Completeness	5
Description									
Prioridade: Essencial Entradas e pré-condições: Não possui. Saídas e pós-condições: Artefato produzido com sucesso. Fluxo de eventos principal A cada intervalo de tempo definido o servidor Inteligente produz uma nova versão para cada artefato de IA suportado pelo sistema.									
Risks									

Version	1.3	Real ID	RF006_4	ID	REQ_19	Name	Cadastrar Formulário de doenças		
Correctness	2	Stability	2	Difficulty	5	Ambiguity	5	Completeness	4
Description									
<p>Prioridade: Desejável</p> <p>Entradas e pré-condições: Não possui.</p> <p>Saídas e pós-condições: Novo formulário criado com sucesso.</p> <p>Fluxo de eventos principal</p> <p>O operador preenche os dados do novo formulário de doenças. A partir destes dados um novo formulário é gerado.</p>									
Risks									

Version	1.3	Real ID	RF007_4	ID	REQ_20	Name	Registrar Operações		
Correctness	3	Stability	2	Difficulty	3	Ambiguity	5	Completeness	3
Description									
<p>Prioridade: Desejável</p> <p>Entradas e pré-condições: Não possui.</p> <p>Saídas e pós-condições: Registro das operações realizado com sucesso.</p> <p>Fluxo de eventos principal</p> <p>O servidor Inteligente registra todas as operações realizadas, incluindo as que não obtiveram êxito.</p>									
Risks									

APPENDIX B

Appendix B presents results of evaluations of this work.

B1: Attributes classification spreadsheet filled

ID Requirement	Correctness	Stability	Difficulty	Ambiguity	Completeness
REQ_01	2	2	2	5	2
REQ_02	2	2	2	5	2
REQ_03	3	4	2	3	5
REQ_04	3	3	1	2	5
REQ_05	1	1	3	5	1
REQ_06	2	2	3	5	3
REQ_07	1	1	3	3	3
REQ_08	2	3	2	3	2
REQ_09	3	3	2	2	3
REQ_10	3	3	1	2	3
REQ_11	3	4	1	3	4
REQ_12	2	3	2	2	3
REQ_13	3	1	4	4	2
REQ_14	5	1	4	4	2
REQ_15	4	5	4	3	3
REQ_16	2	2	3	4	2
REQ_17	1	1	2	3	4
REQ_18	5	5	2	2	5
REQ_19	2	2	5	5	4
REQ_20	3	2	3	5	3
Legend: (1) Very High (2)High (3)Average (4)Low (5) Very Low/Absent*					
* Used just for ambiguity					

B2: Data Base Spreadsheet

Version	1.5	Real ID	RF004	ID	REQ_01	Name	Atualizar Arvore		
Correctness	2	Stability	2	Difficulty	2	Ambiguity	5	Completeness	2
Description									
Prioridade: Essencial Perfil: Médico Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: <i>Download</i> de script inteligente (árvore de decisão) do servidor intermediário para o dispositivo móvel, banco de dados do dispositivo e servidor alterados. Fluxo de eventos Fluxo Principal 1- Usuário seleciona a opção “atualizar árvore” no menu principal do aplicativo; 2- O dispositivo móvel se conecta ao servidor intermediário e faz o <i>download</i> da árvore de decisão atualizada; 3- O dispositivo móvel apaga a árvore local e adiciona a árvore atualizada. 4- Durante o processo de <i>download</i> é exibida uma barra informando o progresso da operação.									
Risks									
Falta de Motivação									
Falta de experiencia na implementação									
Pouco tempo disponivel para implementação									

Version	1.5	Real ID	RF005	ID	REQ_02	Name	Armazenar Evidências		
Correctness	2	Stability	2	Difficulty	2	Ambiguity	5	Completeness	2
Description									
Prioridade: Essencial Perfil: Médico Entradas e pré-condições: Usuário autenticado no sistema e na tela de resultado de diagnóstico. Saídas e pós-condições: Banco de dados do dispositivo móvel alterado. Fluxo de eventos Fluxo Principal 1- O sistema exibe o resultado do diagnóstico e pergunta ao usuário se ele concorda ou não com o resultado informado pela árvore de decisão. 2- O usuário entra com sua opinião acerca do resultado apresentado pela árvore de decisão e pressiona o botão “OK”. 3- O sistema armazena o resultado do diagnóstico, o questionário preenchido e a opinião do usuário sobre o resultado fornecido pela árvore no banco de dados do dispositivo móvel. Fluxo Alternativo 1: O usuário concorda com o resultado da árvore Caso o usuário concorde com o resultado do sistema ele deverá selecionar a opção “sim” e pressionar o botão “OK”. Fluxo Alternativo 2: O usuário não concorda com o resultado da árvore Caso o usuário não concorde com o resultado do sistema ele deverá escolher a opção “não” e justificar sua opinião; Após emitir sua opinião o usuário deve pressionar o botão “OK”. Fluxo Alternativo 3: O usuário não informa se concorda ou não com o resultado da árvore Caso o usuário não informe se concorda ou não com o resultado apresentado pela árvore e pressione o botão “OK”, o sistema irá exibir uma mensagem solicitando que o usuário emita sua opinião.									
Risks									
Falta de informação no armazenamento dos dados									
Não informa como será inserido no sistema a opinião do usuário através de formulário ?									
Não apresenta como deve ser interface externa ?									
Falta de informação referente a segurança da transação ?									
Impactos na implementação, devido a falta de detalhes ?									
Não define o tipo de arquivo enviado									

Version	1.5	Real ID	RF006	ID	REQ_03	Name	Gerenciar Evidências			
Correctness	3	Stability	4	Difficulty	2	Ambiguity	3	Completeness	5	
Description										
Prioridade: Essencial Entradas e pré-condições: Não possui. Saídas e pós-condições: Informações de evidências inseridas/excluídas do servidor intermediário. Fluxo de eventos principal O servidor intermediário, ao receber requisições do dispositivo móvel, armazena ou exclui uma evidência.										
Risks										
Falta de informações essenciais : O requisito não contém informações relevantes e essenciais para implementação deste requisito, não informa sobre os artefatos de IA, nem tampouco define as pré condições do sistema										
Requisito não informa Quem é suportado pelo sistema.										
Instabilidade da interface externa										
Instabilidade na definição dos atributos/ campos										

Version	1.5	Real ID	RF007	ID	REQ_04	Name	Gerenciar Arvores			
Correctness	3	Stability	3	Difficulty	1	Ambiguity	2	Completeness	5	
Description										
Prioridade: Essencial Entradas e pré-condições: Não possui. Saídas e pós-condições: Informações de árvores inseridas/alteradas/excluídas do servidor intermediário. Fluxo de eventos principal O servidor intermediário trata as requisições recebidas com o intuito de inserir uma nova árvore, alterar uma já existente que possui ou não alguma inconsistência ou excluir uma árvore considerada precária.										
Risks										
Não está definida a forma de atualização de uma árvore (upload ou será disponibilizada uma interface para o usuário).										
Não define o que é considerado uma árvore precária. O que fazer se já existe uma árvore e tentar inserir uma nova?										
Falta de experiência da equipe										

Version	1.5	Real ID	RF008	ID	REQ_05	Name	Autenticar Administrador		
Correctness	1	Stability	1	Difficulty	3	Ambiguity	5	Completeness	1
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha).</p> <p>Saídas e pós-condições: Acesso a servidor intermediário liberado com sucesso.</p> <p>Fluxo de eventos principal</p> <p>O administrador preenche o formulário de autenticação com nome de usuário e senha.</p> <p>Fluxo Alternativo 1: Dados válidos</p> <p>Com os dados validados pelo sistema, o usuário é autenticado e automaticamente será apresentado o menu principal do servidor intermediário.</p> <p>Fluxo Alternativo 2: Dados inválidos</p> <p>Caso os dados de entrada sejam inválidos, aparece uma mensagem de erro e a tela de autenticação é exibida novamente.</p>									
Risks									
O requisito não informa a quantidade de caracteres que cada atributo deve possuir									
A conexão com o banco de dados pode ser interrompida e os dados, não serem atualizados no sistema.									
As informações passadas podem ser incoerente com o tipo de dados que o banco de dados irá armazenar									
O sistema pode validar o usuario só pelo login									
Falta de informação referente a tamanho dos campos									
Falta de informação referente tipo de dados									
Falta de informação referente a formatação									

Version	1.5	Real ID	RF009	ID	REQ_06	Name	Gerenciar Usuários		
Correctness	2	Stability	2	Difficulty	3	Ambiguity	5	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Usuário autenticado no sistema.</p> <p>Saídas e pós-condições: Informações de usuários do servidor intermediário incluídas/removidas/alteradas.</p> <p>Fluxo de eventos principal</p> <p>O administrador acessa tela de gerência de usuários, onde poderá incluir, modificar ou excluir os mesmos. Para o cadastro, os seguintes dados são necessários: nome, login, senha, CPF e tipo de usuário (administrador, médico ou agente de saúde).</p>									
Risks									
Não informa se serão necessárias validações nos dados.									
O requisito não informa a quantidade de caracteres que cada atributo deve possuir									
Falta de informação referente ao tamanho dos campos									
O sistema pode receber textos sem coerências ou um conjunto de caracteres sem conteúdo, fazendo um armazenamento de informações desnecessarias.									
O sistema pode receber textos sem coerências ou um conjunto de caracteres sem conteúdo, fazendo um armazenamento de informações desnecessarias.									

Version	1.3	Real ID	RF001_1	ID	REQ_07	Name	Autenticar Profissional de Saúde		
Correctness	1	Stability	1	Difficulty	3	Ambiguity	3	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Perfil: Médico e ACS</p> <p>Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha). Dispositivo conectado ao servidor.</p> <p>Saídas e pós-condições: Usuário autenticado no sistema e download dos dados dos pacientes que serão consultados, formulário e artefato de IA.</p> <p>Fluxo de eventos principal Usuário preenche o formulário de autenticação com nome de usuário e senha. Após, há dois fluxos: dados válidos ou inválidos.</p> <p>Dados válidos Com os dados validados pelo sistema, o usuário é autenticado no sistema e automaticamente será feito o download dos formulários de doenças e artefatos de IA – disponível no servidor – para o dispositivo móvel.</p> <p>Dados inválidos Caso os dados de entrada sejam inválidos, a tela de autenticação é exibida novamente.</p>									
Risks									
Falta Informações essenciais no requisito									
Não está claro em que direção o download será realizado (para o servidor ou para o dispositivo móvel).									
O requisito não contém informações dos itens do questionário.									
As informações contidas no questionário pode ser ilegítimas fazendo com que o questionário seja inválido									
Informações inválidas pode ser passadas, provocando o não aproveitamento das informações.									

Version	1.3	Real ID	RF002_1	ID	REQ_08	Name	Encerrar Sistema		
Correctness	2	Stability	3	Difficulty	2	Ambiguity	3	Completeness	2
Description									
<p>Prioridade: Essencial</p> <p>Perfil: Médico e ACS</p> <p>Entradas e pré-condições: Usuário autenticado no sistema.</p> <p>Saídas e pós-condições: Upload de diagnósticos do dispositivo móvel para o servidor intermediário, banco de dados do dispositivo e servidor alterados, sistema desligado.</p> <p>Fluxo de eventos principal Usuário seleciona a opção “encerrar sistema”. O dispositivo se conecta ao servidor, faz o upload dos diagnósticos realizados no expediente, apaga os artefatos de IA e os registros do expediente, realiza logout do usuário e, por fim, desliga o sistema.</p>									
Risks									
Não está definido o que fazer caso não seja possível conexão com o servidor.									
Não está definido o termo expediente. Deverá ser baixado para o servidor o serviço do dia inteiro ou apenas os registros realizados na sessão.									
Sincronização por indisponibilidade do meio de comunicação.									
A equipe não tem experiência com este tipo de demanda e pode haver problemas no tratamento dos inúmeros problemas que podem ocorrer na comunicação entre os dispositivos.									
Considerando que o grau de dificuldade é alto existe grande risco de problemas durante a implementação.									

Version	1.3	Real ID	RF003_1	ID	REQ_09	Name	Iniciar Diagnostico		
Correctness	3	Stability	3	Difficulty	2	Ambiguity	2	Completeness	3
Description									
Prioridade: Essencial									
Perfil: Médico									
Entradas e pré-condições: Usuário autenticado no sistema.									
Saídas e pós-condições: Banco de dados do dispositivo móvel alterado.									
Fluxo de eventos principal									
O agente de saúde preenche o questionário definido pela equipe médica. Ao final, sistema armazena o resultado do diagnóstico e o questionário preenchido. O sistema ao final do preenchimento do questionário e antes de armazenar a consulta, perguntará ao usuário se o resultado informado pela árvore é aceitável. Caso os dados fornecidos não sejam suficientes para gerar um diagnóstico o sistema deverá alertar o usuário.									
Risks									
Não está definido como os dados serão persistidos.									
Falta definir os dados com clareza									
Falta de clareza em quem vai executar o requisito									
Pode haver problemas de desempenho associados a percorrer a árvore em dispositivos móveis									
Pode existir problemas na persistência dos dados em dispositivos móveis.									
Falta de experiência na equipe.									

Version	1.3	Real ID	RF003_2	ID	REQ_10	Name	Verificar Atualizações com Servidor Inteligente		
Correctness	3	Stability	3	Difficulty	1	Ambiguity	2	Completeness	3
Description									
Prioridade: Essencial									
Entradas e pré-condições: Não possui.									
Saídas e pós-condições: Relação de formulário de doenças e artefatos de IA disponíveis									
Fluxo de eventos principal									
A cada intervalo de tempo a aplicação se comunica com o servidor inteligente e verifica a existência de formulários de doenças e artefatos de IA disponíveis.									
Risks									
Não especifica o intervalo de tempo									
Falta de experiência na implementação									
Falta de clareza no requisito									

Version	1.3	Real ID	RF004_2	ID	REQ_11	Name	Atualização de formulários de Doenças e Artefatos de IA		
Correctness	3	Stability	4	Difficulty	1	Ambiguity	3	Completeness	4
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Relação de formulário de doenças e artefatos de IA disponíveis (RF004).</p> <p>Saídas e pós-condições: Atualização dos formulários de doenças e dos artefatos de IA.</p> <p>Fluxo de eventos principal O servidor Intermediário compara a versão entre os formulários de doenças e artefatos de IA disponíveis. Caso exista versão mais nova disponível o servidor Intermediário faz requisição da mesma. (E se existir um formulário cuja versão antiga já foi preenchida?)</p>									
Risks									
Não especifica o merge entre o formulario antigo e o novo									
Não especifica o formulario antigo nem o novo									
Falta de experiência na implementação									
Tempo de implementação curto									

Version	1.3	Real ID	RF005_2	ID	REQ_12	Name	Transferir informações para o servidor Inteligente		
Correctness	2	Stability	3	Difficulty	2	Ambiguity	2	Completeness	3
Description									
<p>Prioridade: Essencial</p> <p>Entradas e pré-condições: Base de dados modificada e conexão entre servidores estabelecida.</p> <p>Saídas e pós-condições: Transferência de dados de pacientes e diagnósticos realizados efetuada com sucesso.</p> <p>Fluxo de eventos principal A cada intervalo de tempo a aplicação enviará todos os diagnósticos realizados para o servidor inteligente.</p>									
Risks									
Não especifica o intervalo de tempo									
Não Especifica que todos os diagnosticos são enviados para o servidor									
Falta de experiência na implementação									

Version	1.3	Real ID	RF001_3	ID	REQ_13	Name	Autenticar Administrador		
Correctness	3	Stability	1	Difficulty	4	Ambiguity	4	Completeness	2
Description									
Prioridade: Essencial Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha). Saídas e pós-condições: Acesso a servidor intermediário liberado com sucesso. Fluxo de eventos principal O administrador preenche o formulário de autenticação com nome de usuário e senha. Após, há dois fluxos: dados válidos ou inválidos.									
Risks									
Não identifica-se mudanças nas interfaces externas									
Não define o formulário, mas apresenta as informações importantes à tarefa									
Falta definir o formato da transferencia denifindo os SLAs (Service Level Agreement) de segurança.									

Version	1.3	Real ID	RF002_3	ID	REQ_14	Name	Gerenciar Usuários		
Correctness	5	Stability	1	Difficulty	4	Ambiguity	4	Completeness	2
Description									
Prioridade: Essencial E Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Informações de médicos, ACSs e/ou usuários do servidor intermediário incluídas/removidas/alteradas. Para cadastro, os seguintes dados são necessários: Login, senha, CPF, Unidade e data nascimento. Fluxo de eventos principal O administrador acessa tela de gerência de usuários, onde poderá incluir novos usuários, modificar informações, ou então excluí-las.									
Risks									
Não identifica-se mudanças nas interfaces externas									
atividades da gerência não estão relacionadas a requisitos explicitos anteriormente									

Version	1.3	Real ID	RF003_3	ID	REQ_15	Name	Gerenciar Visitas		
Correctness	4	Stability	5	Difficulty	4	Ambiguity	3	Completeness	3
Description									
Prioridade: Essencial Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Agendamento de visitas a pacientes são incluídos/removidos/modificados. Fluxo de eventos principal O administrador acessa tela de visitas. Cada visita contém uma data e está associada a um profissional de saúde e a um paciente. Sistema poderá informar o status de cada visita (desejável).									
Risks									
Não define como são os status e a diferenças entre si									
Não estão claros quais status serão gerenciados									

Version	1.3	Real ID	RF004_3	ID	REQ_16	Name	Gerenciar Pacientes		
Correctness	2	Stability	2	Difficulty	3	Ambiguity	4	Completeness	2
Description									
Prioridade: Essencial Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Informações de pacientes incluídas/removidas/alteradas. Fluxo de eventos principal O administrador acessa tela de gerência de pacientes, onde poderá incluir novos pacientes, modificar informações sobre os agentes existentes ou então excluí-las.									
Risks									
Falta detalhe de informações									
Falta informação sobre na interface externa									

Version	1.3	Real ID	RF005_3	ID	REQ_17	Name	Consultar pacientes		
Correctness	1	Stability	1	Difficulty	2	Ambiguity	3	Completeness	4
Description									
Prioridade: Desejável Entradas e pré-condições: Usuário autenticado no sistema. Saídas e pós-condições: Informações de pacientes visualizadas. Fluxo de eventos principal O administrador acessa tela de pacientes selecionados e agrupados segundo algum critério, onde poderá visualizar os pacientes existentes na base.									
Risks									
Falta Definir quais os critérios de implementação									
Falta informação sobre na interface externa									
Falta de experiência na implementação									
Falta de informação referente a segurança da transação									

Version	1.3	Real ID	RF005_4	ID	REQ_18	Name	Produzir Artefatos de IA		
Correctness	5	Stability	5	Difficulty	2	Ambiguity	2	Completeness	5
Description									
Prioridade: Essencial Entradas e pré-condições: Não possui. Saídas e pós-condições: Artefato produzido com sucesso. Fluxo de eventos principal A cada intervalo de tempo definido o servidor Inteligente produz uma nova versão para cada artefato de IA suportado pelo sistema.									
Risks									
* O requisito não contém informações relevantes e essenciais para implementação deste requisito, não informa sobre os artefatos de IA, nem tampouco define as pré condições do sistema									
Requisito pouco claro, devido a ambiguidade. Quem é suportado pelo sistema?									
Falta de experiência da equipe na implementação									

Version	1.3	Real ID	RF006_4	ID	REQ_19	Name	Cadastrar Formulário de doenças		
Correctness	2	Stability	2	Difficulty	5	Ambiguity	5	Completeness	4
Description									
Prioridade: Desejável Entradas e pré-condições: Não possui. Saídas e pós-condições: Novo formulário criado com sucesso. Fluxo de eventos principal O operador preenche os dados do novo formulário de doenças. A partir destes dados um novo formulário é gerado.									
Risks									
Faltando informações essenciais, podendo resultar em variações.									
Requisito incompleto, faltando informações essenciais, quais os dados e itens do formulário?									
Falta informações essenciais									

Version	1.3	Real ID	RF007_4	ID	REQ_20	Name	Registrar Operações		
Correctness	3	Stability	2	Difficulty	3	Ambiguity	5	Completeness	3
Description									
Prioridade: Desejável Entradas e pré-condições: Não possui. Saídas e pós-condições: Registro das operações realizado com sucesso. Fluxo de eventos principal O servidor Inteligente registra todas as operações realizadas, incluindo as que não obtiveram êxito.									
Risks									
O requisito não contém as informações de que dados devem conter no log.									
Falta de Experiência na equipe na implementação									

B3: Spreadsheet with data executed

	A	B	C	D	E	F	G	H	I	J	K	L
1	Requisito no banco de Dados	A1	A2	A3	A4	A5	SLA1	Sla2	sia3	sia4	sl5	Sequencial
2												
3	REQ005	1	1	3	5	3	4	4	3	4	4	3,8
4	REQ006	2	2	3	5	3	3	3	3	4	4	3,4
5	REQ002	2	2	3	5	3	3	3	3	4	4	3,4
6	REQ007	1	1	3	3	3	4	4	3	2	4	3,4
7	REQ019	2	2	5	5	4	3	3	3	4	3	3,2
8	REQ020	3	2	3	5	3	2	3	3	4	4	3,2
9	REQ013	3	1	4	4	2	2	4	4	3	3	3,2
10	REQ016	2	2	3	4	2	3	3	3	3	3	3,0
11	REQ017	2	2	3	4	4	4	4	2	2	3	3,0
12	REQ001	2	2	2	5	2	3	3	2	4	3	3,0
13	REQ014	5	1	4	4	2	0	4	4	3	3	2,8
14	REQ012	2	3	2	2	3	3	2	2	1	4	2,4
15	REQ008	2	3	2	3	2	3	2	2	2	3	2,4
16	REQ009	3	3	2	2	3	2	2	2	1	4	2,2
17	REQ015	4	5	4	3	3	1	0	4	2	4	2,2
18	REQ010	3	3	1	2	3	2	2	1	1	4	2,0
19	REQ003	3	4	2	3	5	2	1	2	2	2	1,8
20	REQ011	3	4	1	3	4	2	1	1	2	3	1,8
21	REQ004	3	3	1	2	5	2	2	1	1	2	1,6
22	REQ018	5	5	2	2	5	0	0	2	1	2	1,0
23												
24												
25												
26												
27												

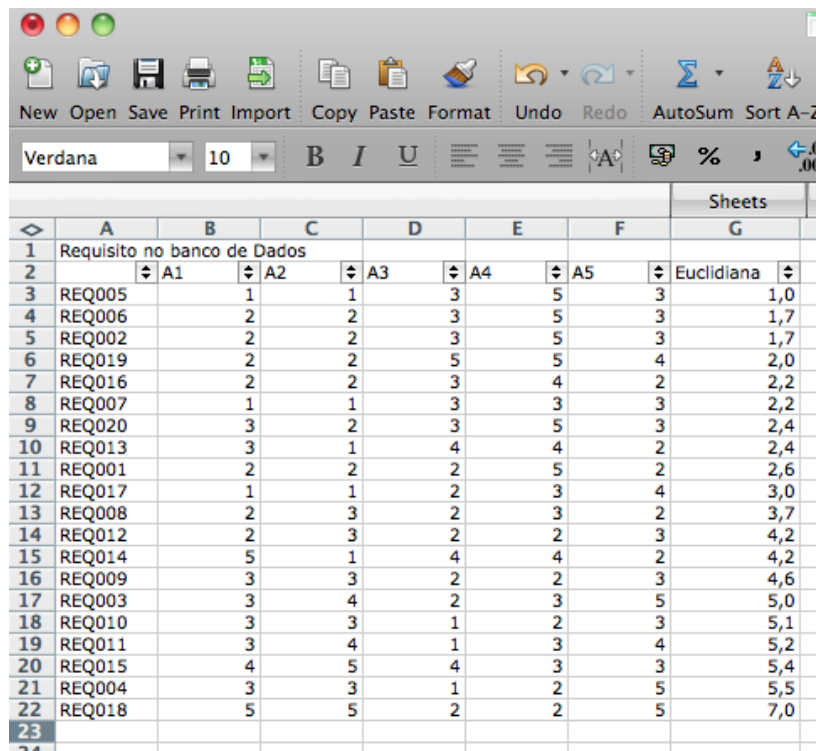
Requirement_021 – Sequential algorithm

	A	B	C	D	E	F	G	H	I	J	K	L
1	Requisito no banco de Dados	A1	A2	A3	A4	A5	SLA1	Sla2	sia3	sia4	sl5	Sequencial
2												
3	REQ006	2	2	3	5	3	4	3	4	4	4	3,6
4	REQ002	2	2	3	5	3	4	3	3	4	4	3,6
5	REQ001	2	2	2	3	2	4	3	4	4	3	3,6
6	REQ020	3	2	3	5	3	3	3	3	4	4	3,4
7	REQ012	2	3	2	2	3	4	4	4	1	4	3,4
8	REQ008	2	3	2	3	2	4	4	2	2	3	3,4
9	REQ005	1	1	3	5	3	3	2	3	4	4	3,2
10	REQ016	2	2	3	4	2	4	3	3	3	3	3,2
11	REQ009	3	3	2	2	3	3	4	4	1	4	3,2
12	REQ019	2	2	5	5	4	4	3	1	4	3	3,0
13	REQ010	3	3	1	2	3	3	4	3	1	4	3,0
14	REQ007	1	1	3	3	3	3	2	3	2	4	2,8
15	REQ017	1	1	2	3	4	3	2	4	2	3	2,8
16	REQ003	3	4	2	3	5	3	3	4	2	2	2,8
17	REQ011	3	4	1	3	4	3	3	3	2	3	2,8
18	REQ013	3	1	4	4	2	3	2	2	3	3	2,6
19	REQ004	3	3	1	2	5	3	4	3	1	2	2,6
20	REQ015	4	5	4	3	3	2	2	2	2	4	2,4
21	REQ014	5	1	4	4	2	1	2	2	3	3	2,2
22	REQ018	5	5	2	2	5	1	2	4	1	2	2,0
23												
24												
25												

Requirement_022 – Sequential algorithm

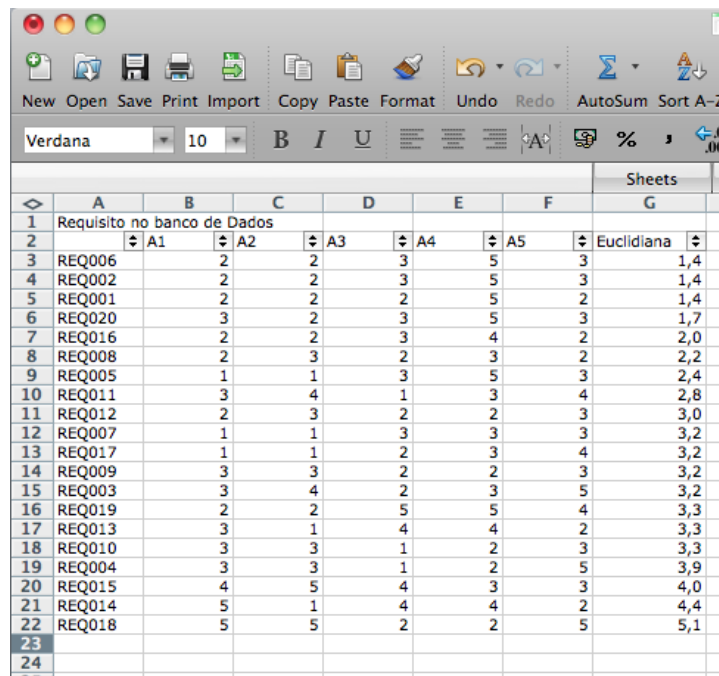
	A	B	C	D	E	F	G	H	I	J	K	L
1	Requisito no banco de Dados	A1	A2	A3	A4	A5	SLA1	Sla2	sia3	sia4	sl5	Sequencial
2												
3	REQ006	2	2	3	5	3	4	4	4	2	3	3,4
4	REQ002	2	2	3	5	3	4	4	4	2	3	3,4
5	REQ016	2	2	3	4	2	4	4	4	3	2	3,4
6	REQ007	1	1	3	3	3	3	3	4	4	3	3,4
7	REQ017	1	1	2	3	4	3	3	3	4	4	3,4
8	REQ020	3	2	3	5	3	3	4	4	2	3	3,2
9	REQ012	2	3	2	2	3	4	3	3	3	3	3,2
10	REQ008	2	3	2	3	2	4	3	3	4	2	3,2
11	REQ019	2	2	5	5	4	4	2	2	4	2	3,2
12	REQ001	2	2	2	5	2	4	4	3	2	2	3,0
13	REQ005	1	1	3	5	3	3	3	4	2	3	3,0
14	REQ009	3	3	2	2	3	3	3	3	3	3	3,0
15	REQ003	3	4	2	3	5	3	2	3	4	3	3,0
16	REQ011	3	4	1	3	4	3	2	4	4	3	3,0
17	REQ010	3	3	1	2	3	3	2	3	3	3	2,8
18	REQ013	3	1	4	4	2	3	3	3	3	2	2,8
19	REQ004	3	3	1	2	5	3	3	2	3	3	2,8
20	REQ015	4	5	4	3	3	2	1	3	4	3	2,6
21	REQ014	5	1	4	4	2	1	3	3	3	2	2,4
22	REQ018	5	5	2	2	5	1	1	3	3	3	2,2
23												
24												
25												

Requirement_023 – Sequential algorithm



	A	B	C	D	E	F	G
1	Requisito no banco de Dados						
2		A1	A2	A3	A4	A5	Euclidiana
3	REQ005	1	1	3	5	3	1,0
4	REQ006	2	2	3	5	3	1,7
5	REQ002	2	2	3	5	3	1,7
6	REQ019	2	2	5	5	4	2,0
7	REQ016	2	2	3	4	2	2,2
8	REQ007	1	1	3	3	3	2,2
9	REQ020	3	2	3	5	3	2,4
10	REQ013	3	1	4	4	2	2,4
11	REQ001	2	2	2	5	2	2,6
12	REQ017	1	1	2	3	4	3,0
13	REQ008	2	3	2	3	2	3,7
14	REQ012	2	3	2	2	3	4,2
15	REQ014	5	1	4	4	2	4,2
16	REQ009	3	3	2	2	3	4,6
17	REQ003	3	4	2	3	5	5,0
18	REQ010	3	3	1	2	3	5,1
19	REQ011	3	4	1	3	4	5,2
20	REQ015	4	5	4	3	3	5,4
21	REQ004	3	3	1	2	5	5,5
22	REQ018	5	5	2	2	5	7,0

Requirement_021 – Euclidean distance algorithm



	A	B	C	D	E	F	G
1	Requisito no banco de Dados						
2		A1	A2	A3	A4	A5	Euclidiana
3	REQ006	2	2	3	5	3	1,4
4	REQ002	2	2	3	5	3	1,4
5	REQ001	2	2	2	5	2	1,4
6	REQ020	3	2	3	5	3	1,7
7	REQ016	2	2	3	4	2	2,0
8	REQ008	2	3	2	3	2	2,2
9	REQ005	1	1	3	5	3	2,4
10	REQ011	3	4	1	3	4	2,8
11	REQ012	2	3	2	2	3	3,0
12	REQ007	1	1	3	3	3	3,2
13	REQ017	1	1	2	3	4	3,2
14	REQ009	3	3	2	2	3	3,2
15	REQ003	3	4	2	3	5	3,2
16	REQ019	2	2	5	5	4	3,3
17	REQ013	3	1	4	4	2	3,3
18	REQ010	3	3	1	2	3	3,3
19	REQ004	3	3	1	2	5	3,9
20	REQ015	4	5	4	3	3	4,0
21	REQ014	5	1	4	4	2	4,4
22	REQ018	5	5	2	2	5	5,1

Requirement_022 – Euclidean distance algorithm

Verdana 10 B I U							AutoSum Sort A-Z
							0.00
	A	B	C	D	E	F	Sheets
1	Requisito no banco de Dados						
2	A1	A2	A3	A4	A5	Euclidiana	
3	REQ007	1	1	3	3	3	1,7
4	REQ017	1	1	2	3	4	1,7
5	REQ012	2	3	2	2	3	2,0
6	REQ006	2	2	3	5	3	2,2
7	REQ002	2	2	3	5	3	2,2
8	REQ016	2	2	3	4	2	2,2
9	REQ009	3	3	2	2	3	2,2
10	REQ020	3	2	3	5	3	2,4
11	REQ008	2	3	2	3	2	2,4
12	REQ005	1	1	3	5	3	2,6
13	REQ003	3	4	2	3	5	2,6
14	REQ019	2	2	5	5	4	2,8
15	REQ013	3	1	4	4	2	2,8
16	REQ010	3	3	1	2	3	2,8
17	REQ004	3	3	1	2	5	2,8
18	REQ001	2	2	2	5	2	3,0
19	REQ011	3	4	1	3	4	3,0
20	REQ015	4	5	4	3	3	3,9
21	REQ014	5	1	4	4	2	4,0
22	REQ018	5	5	2	2	5	4,6
23							

Requirement_023 – Euclidean distance algorithm

ANNEX A

Annex A presents the requirements(versions 1.3 and 1.5 respectively) used in evaluations of this work.

Version 1.3

Functional requirements

1. Ator: Profissional de Saúde

[RF001] Autenticar Profissional de Saúde

Prioridade: Essencial

Perfil: Médico e ACS

Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha). Dispositivo conectado ao servidor.

Saídas e pós-condições: Usuário autenticado no sistema e download dos dados dos pacientes que serão consultados, formulário e artefato de IA.

Fluxo de eventos principal

Usuário preenche o formulário de autenticação com nome de usuário e senha. Após, há dois fluxos: dados válidos ou inválidos.

Dados válidos

Com os dados validados pelo sistema, o usuário é autenticado no sistema e automaticamente será feito o download dos formulários de doenças e artefatos de IA—disponível no servidor – para o dispositivo móvel.

Dados inválidos

Caso os dados de entrada sejam inválidos, a tela de autenticação é exibida novamente.

[RF002] Encerrar Sistema

Prioridade: Essencial

Perfil: Médico e ACS

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: *Upload* de diagnósticos do dispositivo móvel para o servidor intermediário, banco de dados do dispositivo e servidor alterados, sistema desligado.

Fluxo de eventos principal

Usuário seleciona a opção “encerrar sistema”. O dispositivo se conecta ao servidor, faz o *upload* dos diagnósticos realizados no expediente, apaga os artefatos de IA e os registros do expediente, realiza *logout* do usuário e, por fim, desliga o sistema.

[RF003] Iniciar Diagnóstico

Prioridade: Essencial

Perfil: Médico

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Banco de dados do dispositivo móvel alterado.

Fluxo de eventos principal

O agente de saúde preenche o questionário definido pela equipe médica. Ao final, sistema armazena o resultado do diagnóstico e o questionário preenchido. O sistema ao final do preenchimento do questionário e antes de armazenar a consulta, perguntará ao usuário se

o resultado informado pela árvore é aceitável. Caso os dados fornecidos não sejam suficientes para gerar um diagnóstico o sistema deverá alertar o usuário.

[RF004] Reportar Suspeita

Prioridade: Essencial

Perfil: ACS

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Banco de dados do dispositivo móvel alterado.

Fluxo de eventos principal

Usuário preenche formulário de reporte de suspeita para o paciente selecionado, onde deverão ser disponibilizadas informações da doença, sintomas e imagens, quando necessário.

[RF005] Solicitar Agendamento de Consulta na Unidade

Prioridade: Desejável

Perfil: ACS e Médico

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Agendamento é enviado para o servidor intermediário.

Fluxo de eventos principal

Para o paciente selecionado o usuário solicita agendamento de consulta, onde as seguintes informações são requeridas: i) Motivo do agendamento e ii) Sugestão de data para consulta.

Obs.: Esta funcionalidade requer que o servidor intermediário gerencie as consultas médicas nas USFs.

[RF006] Cadastrar Paciente

Prioridade: Essencial

Perfil: ACS e Médico

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Banco de dados do dispositivo móvel alterado.

Fluxo de eventos principal

Usuário preenche formulário de cadastro com as informações necessárias sobre o paciente.

[RF007] Consultar Diagnóstico

Prioridade: Essencial

Perfil: Médico

Entradas e pré-condições: Usuário autenticado, paciente cadastrado e questionário de diagnóstico completo.

Saídas e pós-condições: Diagnóstico mostrado na tela.

Fluxo de eventos principal

O usuário pode visualizar os resultados de diagnósticos de um determinado paciente realizados anteriormente.

[RF008] Consultar Catálogo de Doenças

Prioridade: Desejável

Perfil: Médico e ACS

Entradas e pré-condições: Agente de saúde autenticado no sistema.

Saídas e pós-condições: Informações sobre doenças são mostradas na tela.

Fluxo de eventos principal

O agente de saúde seleciona uma doença disponível no catálogo de doenças e as informações sobre ela são mostradas na tela.

2. Ator: Servidor Intermediário

[RF001] Autenticar Dispositivo Móvel

Prioridade: Essencial

Entradas e pré-condições: Não possui.

Saídas e pós-condições: Atualização dos formulários de doenças e artefatos de IA disponíveis para o dispositivo móvel e liberação acesso aos dados do servidor intermediário via dispositivo móvel.

Fluxo de eventos principal

O servidor intermediário, ao receber uma requisição de autenticação, verifica se o *login* e a senha informados existem na sua base. A liberação ou não do acesso dependerá da existência ou não das informações advindas do dispositivo móvel. Com o usuário autenticado o servidor Intermediário envia para o dispositivo móvel, caso existam, as novas versões do formulário e artefatos de IA.

[RF002] Armazenar Dados de Pacientes

Prioridade: Essencial

Entradas e pré-condições: Autenticação do usuário de dispositivo móvel.

Saídas e pós-condições: Armazenamento de dados de pacientes realizada com sucesso.

Fluxo de eventos principal

É efetuado o cadastro automático de paciente com base nas informações lançadas pelo usuário do dispositivo móvel.

[RF003] Verificar atualizações com Servidor Inteligente

Prioridade: Essencial

Entradas e pré-condições: Não possui.

Saídas e pós-condições: Relação de formulário de doenças e artefatos de IA disponíveis

Fluxo de eventos principal

A cada intervalo de tempo a aplicação se comunica com o servidor inteligente e verifica a existência de formulários de doenças e artefatos de IA disponíveis (RF002 – Servidor Inteligente).

[RF004] Atualização de Formulários de Doenças e Artefatos de IA

Prioridade: Essencial

Entradas e pré-condições: Relação de formulário de doenças e artefatos de IA disponíveis (RF004).

Saídas e pós-condições: Atualização dos formulários de doenças e dos artefatos de IA.

Fluxo de eventos principal

O servidor Intermediário compara a versão entre os formulários de doenças e artefatos de IA disponíveis. Caso exista versão mais nova disponível o servidor Intermediário faz requisição da mesma. (E se existir um formulário cuja versão antiga já foi preenchida?)

[RF005] Transferir Informações para o Servidor Inteligente

Prioridade: Essencial

Entradas e pré-condições: Base de dados modificada e conexão entre servidores estabelecida.

Saídas e pós-condições: Transferência de dados de pacientes e diagnósticos realizados efetuada com sucesso.

Fluxo de eventos principal

A cada intervalo de tempo a aplicação enviará todos os diagnósticos realizados para o servidor inteligente.

3. Ator: Administrador

[RF001] Autenticar Administrador

Prioridade: Essencial

Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha).

Saídas e pós-condições: Acesso a servidor intermediário liberado com sucesso.

Fluxo de eventos principal

O administrador preenche o formulário de autenticação com nome de usuário e senha.

Após, há dois fluxos: dados válidos ou inválidos.

[RF002] Gerenciar Usuários

Prioridade: Essencial

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Informações de médicos, ACSs e/ou usuários do servidor intermediário incluídas/removidas/alteradas. Para cadastro, os seguintes dados são necessários: Login, senha, CPF, Unidade e data nascimento.

Fluxo de eventos principal

O administrador acessa tela de gerência de usuários, onde poderá incluir novos usuários, modificar informações, ou então excluí-las.

[RF003] Gerenciar Visitas

Prioridade: Essencial

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Agendamento de visitas a pacientes são incluídos/removidos/modificados.

Fluxo de eventos principal

O administrador acessa tela de visitas. Cada visita contém uma data e está associada a um profissional de saúde e a um paciente. Sistema poderá informar o status de cada visita (desejável).

[RF004] Gerenciar Pacientes

Prioridade: Essencial

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Informações de pacientes incluídas/removidas/alteradas.

Fluxo de eventos principal

O administrador acessa tela de gerência de pacientes, onde poderá incluir novos pacientes, modificar informações sobre os agentes existentes ou então excluí-las.

[RF005] Consultar Pacientes

Prioridade: Desejável

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Informações de pacientes visualizadas.

Fluxo de eventos principal

O administrador acessa tela de pacientes selecionados e agrupados segundo algum critério, onde poderá visualizar os pacientes existentes na base.

[RF006] Consultar Diagnósticos

Prioridade: Desejável

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Informações de diagnósticos visualizadas.

Fluxo de eventos principal

O administrador acessa tela de diagnósticos selecionados e agrupados segundo algum critério, onde poderá visualizar os diagnósticos realizados.

[RF007] Visualizar Relatórios

Prioridade: Desejável

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Relatórios visualizados com sucesso.

Fluxo de eventos principal

O administrador acessa opção de visualização de relatórios relacionados com os diagnósticos realizados, pacientes e agentes de saúde.

4. Ator: Servidor Inteligente

[RF001] Autenticar Servidor Intermediário

Prioridade: Essencial

Entradas e pré-condições: Dados de autenticação do usuário (nome do servidor e senha).

Saídas e pós-condições: Conexão entre os Servidores Intermediário e de IA estabelecida.

Fluxo de eventos principal

O servidor de IA, ao receber uma requisição de autenticação, verifica a identificação do servidor intermediário e a senha informados existem na sua base. O estabelecimento ou não da conexão dependerá da existência ou não das informações advindas do servidor intermediário.

[RF002] Listar formulários de doenças e artefatos de IA

Prioridade: Importante

Entradas e pré-condições: Conexão estabelecida (RF001).

Saídas e pós-condições: Relação de formulários de doenças e artefatos de IA mais atuais.

Fluxo de eventos principal

O servidor Inteligente informa ao servidor Intermediário a relação dos formulários de doenças e artefatos de IA disponíveis e suas respectivas versões.

[RF003] Transferir formulários de doenças e artefatos de IA

Prioridade: Essencial

Entradas e pré-condições: Conexão estabelecida (RF001).

Saídas e pós-condições: Formulário de doença ou artefato de IA solicitado.

Fluxo de eventos principal

O servidor Inteligente envia ao servidor Intermediário o documento contendo o formulário de doença ou artefatos de IA solicitado.

[RF004] Receber informações do Servidor Intermediário

Prioridade: Essencial

Entradas e pré-condições: Conexão estabelecida (RF001).

Saídas e pós-condições: Dados recebidos com sucesso.

Fluxo de eventos principal

O servidor Inteligente recebe os dados enviados pelo Servidor Intermediário e persiste no banco de dados central.

[RF005] Produzir artefatos de IA

Prioridade: Essencial

Entradas e pré-condições: Não possui.

Saídas e pós-condições: Artefato produzido com sucesso.

Fluxo de eventos principal

A cada intervalo de tempo definido o servidor Inteligente produz uma nova versão para cada artefato de IA suportado pelo sistema.

[RF006] Cadastrar formulário de doenças

Prioridade: Desejável

Entradas e pré-condições: Não possui.

Saídas e pós-condições: Novo formulário criado com sucesso.

Fluxo de eventos principal

O operador preenche os dados do novo formulário de doenças. A partir destes dados um novo formulário é gerado.

[RF007] Registrar operações

Prioridade: Desejável

Entradas e pré-condições: Não possui.

Saídas e pós-condições: Registro das operações realizado com sucesso.

Fluxo de eventos principal

O servidor Inteligente registra todas as operações realizadas, incluindo as que não obtiveram êxito.

Version 1.5

Functionals Requirements

1. Ator: Médico

[RF001] Autenticar Usuário

Prioridade: Essencial

Perfil: Médico

Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha).

Saídas e pós-condições: Usuário é autenticado no sistema e o sistema apresenta a tela inicial (menu principal do aplicativo).

Fluxo de eventos

Fluxo Principal

O usuário preenche o formulário de autenticação com nome de usuário e senha;

O usuário pressiona o botão “login”.

Fluxo Alternativo 1: Dados válidos

Com os dados validados pelo sistema, o usuário é autenticado no sistema e automaticamente será apresentado o menu principal do aplicativo móvel.

Fluxo Alternativo 2: Dados inválidos

Caso os dados de entrada sejam inválidos, aparece uma mensagem de erro e a tela de autenticação é exibida novamente.

[RF002] Enviar evidências

Prioridade: Essencial

Perfil: Médico

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Upload de evidências do dispositivo móvel para o servidor intermediário, banco de dados do dispositivo e servidor alterados.

Fluxo de eventos

Fluxo Principal

Usuário seleciona a opção “enviar evidências” no menu principal do aplicativo;

O dispositivo se conecta ao servidor intermediário e faz o *upload* dos diagnósticos realizados no expediente;
O dispositivo móvel apaga os diagnósticos enviados ao servidor;
Durante o processo de *upload* é exibida uma mensagem informando o progresso da operação.

[RF003] Efetuar Diagnóstico

Prioridade: Essencial

Perfil: Médico

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Diagnóstico do paciente de acordo com as respostas fornecidas ao questionário.

Fluxo de eventos

Fluxo Principal

Usuário seleciona a opção “diagnóstico” no menu principal do aplicativo;
O agente de saúde preenche o questionário definido pela equipe médica;
Ao final do preenchimento, o usuário pressiona o botão “OK”;
O sistema exibe o diagnóstico.

Fluxo Alternativo 1: O usuário não preenche completamente o questionário

Caso o usuário não responda a alguma(s) pergunta(s) o sistema irá exibir uma mensagem solicitando que o usuário preencha completamente o questionário.

[RF004] Atualizar árvore

Prioridade: Essencial

Perfil: Médico

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: *Download* de script inteligente (árvore de decisão) do servidor intermediário para o dispositivo móvel, banco de dados do dispositivo e servidor alterados.

Fluxo de eventos

Fluxo Principal

Usuário seleciona a opção “atualizar árvore” no menu principal do aplicativo;
O dispositivo móvel se conecta ao servidor intermediário e faz o *download* da árvore de decisão atualizada;
O dispositivo móvel apaga a árvore local e adiciona a árvore atualizada.
Durante o processo de *download* é exibida uma barra informando o progresso da operação.

[RF005] Armazenar evidências

Prioridade: Essencial

Perfil: Médico

Entradas e pré-condições: Usuário autenticado no sistema e na tela de resultado de diagnóstico.

Saídas e pós-condições: Banco de dados do dispositivo móvel alterado.

Fluxo de eventos

Fluxo Principal

O sistema exibe o resultado do diagnóstico e pergunta ao usuário se ele concorda ou não com o resultado informado pela árvore de decisão.

O usuário entra com sua opinião acerca do resultado apresentado pela árvore de decisão e pressiona o botão “OK”.

O sistema armazena o resultado do diagnóstico, o questionário preenchido e a opinião do usuário sobre o resultado fornecido pela árvore no banco de dados do dispositivo móvel.

Fluxo Alternativo 1: O usuário concorda com o resultado da árvore

Caso o usuário concorde com o resultado do sistema ele deverá selecionar a opção “sim” e pressionar o botão “OK”.

Fluxo Alternativo 2: O usuário não concorda com o resultado da árvore

Caso o usuário não concorde com o resultado do sistema ele deverá escolher a opção “não” e justificar sua opinião;

Após emitir sua opinião o usuário deve pressionar o botão “OK”.

Fluxo Alternativo 3: O usuário não informa se concorda ou não com o resultado da árvore

Caso o usuário não informe se concorda ou não com o resultado apresentado pela árvore e pressione o botão “OK”, o sistema irá exibir uma mensagem solicitando que o usuário emita sua opinião.

2. Ator: Servidor Intermediário

[RF006] Gerenciar Evidências

Prioridade: Essencial

Entradas e pré-condições: Não possui.

Saídas e pós-condições: Informações de evidências inseridas/excluídas do servidor intermediário.

Fluxo de eventos principal

O servidor intermediário, ao receber requisições do dispositivo móvel, armazena ou exclui uma evidência.

[RF007] Gerenciar Árvores

Prioridade: Essencial

Entradas e pré-condições: Não possui.

Saídas e pós-condições: Informações de árvores inseridas/alteradas/excluídas do servidor intermediário.

Fluxo de eventos principal

O servidor intermediário trata as requisições recebidas com o intuito de inserir uma nova árvore, alterar uma já existente que possui ou não alguma inconsistência ou excluir uma árvore considerada precária.

[RF008] Autenticar Administrador

Prioridade: Essencial

Entradas e pré-condições: Dados de autenticação do usuário (nome de usuário e senha).

Saídas e pós-condições: Acesso a servidor intermediário liberado com sucesso.

Fluxo de eventos principal

O administrador preenche o formulário de autenticação com nome de usuário e senha.

Fluxo Alternativo 1: Dados válidos

Com os dados validados pelo sistema, o usuário é autenticado e automaticamente será apresentado o menu principal do servidor intermediário.

Fluxo Alternativo 2: Dados inválidos

Caso os dados de entrada sejam inválidos, aparece uma mensagem de erro e a tela de autenticação é exibida novamente.

3. Ator: Administrador

[RF009] Gerenciar Usuários

Prioridade: Essencial

Entradas e pré-condições: Usuário autenticado no sistema.

Saídas e pós-condições: Informações de usuários do servidor intermediário incluídas/removidas/alteradas.

Fluxo de eventos principal

O administrador acessa tela de gerência de usuários, onde poderá incluir, modificar ou excluir os mesmos. Para o cadastro, os seguintes dados são necessários: nome, login, senha, CPF e tipo de usuário (administrador, médico ou agente de saúde).

ANNEX B

Annex B presents Taxonomy-Base Risk Identification Questionnaire for requirements [CARR93].

Product Engineering

1. Requirements

a. Stability

[Are requirements changing even as the product is being produced?]

[1] Are the requirements stable?

(No) (1.a) What is the effect on the system?

- Quality
- Functionality
- Schedule
- Integration
- Design
- Testing

[2] Are the external interfaces changing?

b. Completeness

[Are requirements missing or incompletely specified?]

[3] Are there any TBDs in the specifications?

[4] Are there requirements you know should be in the specification but aren't?

(Yes) (4.a) Will you be able to get these requirements into the system?

[5] Does the customer have unwritten requirements/expectations?

(Yes) (5.a) Is there a way to capture these requirements?

[6] Are the external interfaces completely defined?

c. Clarity

[Are requirements unclear or in need of interpretation?]

[7] Are you able to understand the requirements as written?

(No) (7.a) Are the ambiguities being resolved satisfactorily?

(Yes) (7.b) There are no ambiguities or problems of interpretation?

d. Validity

[Will the requirements lead to the product the customer has in mind?]

[8] Are there any requirements that may not specify what the customer really wants?

(Yes) (8.a) How are you resolving this?

[9] Do you and the customer understand the same thing by the requirements?

(Yes) (9.a) Is there a process by which to determine this?

[10] How do you validate the requirements?

- Prototyping
- Analysis
- Simulations

e. Feasibility

[Are requirements infeasible from an analytical point of view?]

[11] Are there any requirements that are technically difficult to implement?

(Yes) (11.a) What are they?

(Yes) (11.b) Why are they difficult to implement?

(No) (11.c) Were feasibility studies done for these requirements?

(Yes) (11.c.1) How confident are you of the assumptions made in the studies?

f. Precedent

[Do requirements specify something never done before, or that your company has not done before?]

[12] Are there any state-of-the-art requirements?

- Technologies
- Methods
- Languages
- Hardware

(No) (12.a) Are any of these new to you?

(Yes) (12.b) Does the program have sufficient knowledge in these areas?

(No) (12.b.1) Is there a plan for acquiring knowledge in these areas?

g. Scale

[Do requirements specify a product larger, more complex, or requiring a larger organization than in the experience of the company?]

[13] Is the system size and complexity a concern?

(No) (13.a) Have you done something of this size and complexity before?

[14] Does the size require a larger organization than usual for your company?