FEDERAL UNIVERSITY OF PAMPA

Miguel da Silva Ecar

Intelligent Classification of SPI Practices and Evidences Based on NLP and Semantic Similarity

Alegrete
2021

Miguel da Silva Ecar

Intelligent Classification of SPI Practices and Evidences Based on NLP and Semantic Similarity

Master thesis presented as partial requirement for obtaining the Masters degree of Software Engineering at Federal University of Pampa.

Supervisor: Prof. PhD João Pablo Silva da Silva

Alegrete
2021

Ficha catalográfica elaborada automaticamente com os dados fornecidos
pelo(a) autor(a) através do Módulo de Biblioteca do
Sistema GURI (Gestão Unificada de Recursos Institucionais) .

E17i Ecar, Miguel da Silva
Intelligent Classification of SPI Practices and Evidences Based on NLP and Semantic Similarity / Miguel da Silva Ecar.
117 p.

Dissertação(Mestrado)-- Universidade Federal do Pampa, MESTRADO EM ENGENHARIA DE SOFTWARE, 2021.
"Orientação: João Pablo Silva da Silva".

1. Melhoria de Processo de Software. 2. Similaridade Semântica. 3. Ontologia. 4. Software Inteligente. I. Título.

MIGUEL DA SILVA ECAR

INTELLIGENT CLASSIFICATION OF SPI PRACTICES AND EVIDENCES BASED ON NLP AND SEMANTIC SIMILARITY

Dissertação apresentada ao Programa de Pós-graduação em Engenharia de Software da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Mestre em Engenharia de Software.

Dissertação defendida e aprovada em: 30 de setembro de 2021.

Banca examinadora:

_____

Prof. Dr. João Pablo Silva da Silva

Orientador

UNIPAMPA

_____

Prof. Dr. Marcelo Resende Thielo

UNIPAMPA

_____

Prof. Dr. Marcelo Soares Pimenta

UFRGS

Assinado eletronicamente por **JOAO PABLO SILVA DA SILVA**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 01/10/2021, às 11:23, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **MARCELO RESENDE THIELO**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 01/10/2021, às 12:09, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **MARCELO SOARES PIMENTA**, **Usuário Externo**, em 01/10/2021, às 14:49, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0629065** e o código CRC **F192DD47**.

I dedicate this project to God Almighty my creator, my strong pillar, my source of inspiration, wisdom, knowledge and understanding. He has been the source of my strength throughout this program and on His wings only have I soared.

ACKNOWLEDGEMENTS

First of all, I would like to thank God that give me healthy, strength and wisdom to overcome all difficulties.

I would like to express my special thanks of gratitude to my advisor, Dr. João Pablo Silva da Silva who gave me the golden opportunity to do this wonderful project, which also helped me in doing a lot of Research and I came to know about so many new things. I would also like to thank Tiago Soldá for his consistent support and guidance during the running of this project.

Special thank to my wife Luziele and my family, for putting up with me being sat in the office for hours on end, and for providing guidance and a sounding board when required. Thank to my daughter Samia, my little partner in working hours during the night.

Last but not least, I would like to thank all who directly or indirectly were part of my training, thank you very much.

I am making this project not only for marks but to also increase my knowledge.

Happiness is the illusory claim
to convert an instant of joy
into eternity.
(Clóvis de Barros Filho)

ABSTRACT

Software Process Improvement (SPI) consists of a set of changes in software development companies, which introduces new and improved methods, techniques, and tools. SPI initiatives generally are performed based on a reference model, such as CMMI, ISO 9001, ISO 15504, among others. One of the first steps when investing in SPI initiatives is the SPI Diagnostic. The SPI Diagnostic is generally performed manually, which demands high effort from consultants. Moreover, a high data volume is generated and must be analyzed, which is bound to subjective analysis. Since there has been a lack of automation tools to support this process, it turns SPI Diagnostic a challenging process. This work aims to propose an intelligent tool called Coptic for Practice-Evidence classification, using Natural Language Processing and Semantic Similarity. We also propose Base of Knowledge about Software Engineering Practices (Badge), which is a domain ontology that generalizes SPI resources that says "what should be done" and PStory, which is a template to write pieces of evidence. We evaluated Badge through a focus group session. We evaluated PStory through an exercise and questionnaire with industry professionals. We evaluated Coptic by a quasi-experiment with PStories evaluated by industry professionals. As an outcome, Coptic presented satisfactory results using the initial corpus. We conclude that Coptic presents a valuable result in terms of providing support to professionals in performing a SPI Diagnostic. Badge introduces a domain ontology that differs from the related proposals in literature and has value to SPI initiatives. We also concluded that PStory introduces a simple way to write pieces of evidence, and Coptic provides support to SPI Practices-Evidence matching process.

Key-words: Coptic, Badge, PStory, Software Process Improvement, SPI.

RESUMO

Melhoria de Processo de Software (MPS) consiste em um conjunto de mudanças nas empresas de desenvolvimento de software, que pode estar relacionado a criação ou melhoria de métodos, técnicas, processos e ferramentas. Iniciativas de MPS geralmente são realizados com base em um modelo de referência, como CMMI, ISO 9001, ISO 15504, entre outros. Um dos primeiros passos ao investir em iniciativas de SPI é o diagnóstico. Na maioria dos casos o diagnóstico é realizado manualmente, o que demanda maior esforço dos consultores. Além disso, um grande volume de dados é gerado e deve ser analisado, o que resulta em análises com certa subjetividade. Como não há ferramentas de automação para dar suporte a esse processo, o diagnóstico torna-se um processo desafiador. Este estudo tem como objetivo propor uma ferramenta inteligente denominada Coptic para classificação de evidências e práticas de MPS, utilizando Processamento de Língua Natural e Similaridade Semântica. Também propomos a Badge, que é uma ontologia de domínio que generaliza recursos de MPS do tipo que dizem "o que deve ser feito" e PStory, que é um modelo para escrita de evidências. A Ontologia Badge foi avaliada através de um grupo focal. Avaliamos o PStory por meio de um exercício e questionário com profissionais da indústria. Coptic foi avaliado através de um *quasi-experimento* com PStories avaliadas por profissionais da indústria. Como resultado, o Coptic apresentou resultados satisfatórios com o corpus inicial. Concluímos que Badge apresenta uma ontologia de domínio que difere das propostas relacionadas na literatura e tem valor para iniciativas de MPS. O PStory apresenta uma maneira simples de escrever evidências, e o Coptic fornece suporte para o processo de classificação de evidências e práticas de MPS.

Palavras-chave: Coptic, Badge, PStory, Melhoria de Processo de Software.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

14 – 14c
AI – Artificial Intelligence
AtomicSEB – Atomic SEB
AtomicSEP – Atomic SEP
Badge – Base of Knowledge about Software Engineering Practices
CLEI 2020 – XLVI Latin American Computer Conference - 2020
CM – Configuration Management
CMMI – Capability Maturity Model Integration
CompositeSEB – Composite SEB
CompositeSEP – Composite SEP
Coptic – Collector of Software Process Improvement Evidences
DL – Description Logics
DSL – Domain Specific Language
GQM – Goal Question Metric
IDEAL – Initiating Diagnosing Establishing Acting Learning
KnowledgeBase – Software Engineering Knowledge Base
MA – Measurement and Analysis
ML – Machine Learning
MPS.BR – Brazilian Software Process Improvement
NLP – Natural Language Processing
PDCA – Plan Do Check Act
PMBoK – Project Management Body of Knowledge
PMC – Project Monitoring and Control
PP – Project Planning
PPQA – Process and Product Quality Assurance
PStory – Practitioner Story
QIP – Quality Improvement Paradigm
Ranking – Ranking
REQM – Requirements Management
RUP – Rational Unified Process
SAM – Supplier Agreement Management
SE – Software Engineering
SEB – Software Engineering Boundary
SEP – Software Engineering Practice
SEPG – Software Engineering Process Group
SLR – Systematic Literature Review
SMEs – Small and Medium Enterprises
SPI – Software Process Improvement

SPI Diagnostic – Software Process Improvement Diagnostic
SPT – Syntactic Parse Tree
SWEBoK – Software Engineering Body of Knowledge
TQM – Total Quality Management
UML – Unified Modeling Language
US – User Story

TABLE OF CONTENTS

1 INTRODUCTION

Software Process is an essential tool that companies take to generate high-quality software products. The software process is the complete set of software engineering activities needed to transform user requirements into the software (HUMPHREY, 1988). Pressman (2010) defines software process as a framework for activities, actions, and tasks required to build software. A software process is a set of interrelated actions and activities carried out to obtain a specified set of products, results, or services (SEI, 2010), *i.e.*, a steps sequence performed with a specific purpose (BOURQUE; FAIRLEY, 2014).

The globalized marketplace makes the organization improve its products and services continuously. Thus, they should apply improvement practices to maintain theirs processes aligned to their business needs (SEI, 2010). Considering that software processes define how o build software products, they should evolve and improve to fit new organization targets and needs. SPI area is the way to guide this kind of situation.

SPI consists of a set of changes in software development companies, which introduces new and improved methods, techniques, and tools. SPI is an initiative to avoid the delivery of low-quality systems (HUMPHREY, 1989). Such changes also affect work organization, attitudes, and management practices at all levels (IVERSEN, 1998). According to Pressman (2005), SPI covers a set of activities that will lead to a better software process and, as a consequence, higher-quality software delivered. Considering what the definition in Pries-Heje, Johansen et al. (2010), SPI must involve people actively and affect their daily activities, making the business successful. One of the most basic processes for SPI is called Plan Do Check Act (PDCA). The PDCA cycle was proposed by Shewhart e Deming (1986), and it is commonly used as a problem-solving model in the context of quality management (DEMING, 2018).

High maturity level organizations have maintained a workgroup called Software Engineering Process Group (SEPG) that is responsible for carrying out the improvement actions. SEPG is the focal point for process improvement. Composed of line practitioners who have varied skills, the group is at the center of the collaborative effort of everyone in the organization who is involved with software engineering process improvement (FOWLER; RIFKIN, 1990). For the context of this work, we use SEPG to refer to practitioners or consultants that act in SPI initiatives.

Among several SPI responsibilities, the SPI Diagnostic is vital due to its importance in obtaining a clear perception about the current state of organization software processes. SPI Diagnostic consists in identifying the real problems and what model is more appropriate to the current situation (SILVA; BRANCHER, 2017). Moreira et al. (2013) says that SPI Diagnostic is the verification process of identifying strengths and weaknesses to conduct improvement actions. Based on this, we define SPI Diagnostic as the process to understand an organization and its processes, focusing on recommending resources for improvement. Generally SPI Diagnostic are conducted using *ad-hoc* strategies, based

on own resources, knowledge, experience, skills, and abilities (GARTISER et al., 2014). SPI Diagnostic strategies are usually created considering the improvement approach, such as, IDEAL (MCFEELEY, 1996; ANACLETO et al., 2004), or PDCA (QUINQUIOLO, 2002), or it is based on process-reference model, such as, CMMI (CHRISSIS; KONRAD; SHRUM, 2011), or Brazilian Software Process Improvement (MPS.BR) (SOFTEX, 2011).

1.1 Motivation

Let's start this section with a problem situation. A SEPG starts a SPI Diagnostic process by planning a series of interviews in the target organization. Due to logistic purposes, only one SEPG member should run the interview, but all SEPG members will analyze the recorded interview result. Some of the target organization members were not comfortable with the interview being audio recorded. Thus, the planning was to run a semi-structured interview and to take notes about the answers. In this kind of interview, the SEPG creates a protocol to guide the interviewers, but there is the flexibility to improvise based on answers or interview directions.

After the interview, the SEPG obtain a massive quantity of information. Thus, this information was separated and distributed to all SEPG members. After a preliminary analysis, performed individually, SEPG members performed a shared analysis in a meeting to discuss and to create a final report. In this meeting, SEPG members realized that they did not correctly obtain some information as the interview flows. For example, actions or tasks reported without saying the tool used to perform it; different practitioners reported same tasks as different practices; roles inside organizations processes reported with various forms that might arise by only different visions of the same thing or misunderstandings; among others particularities. During the meeting, the SEPG member, responsible for running the interview, was constantly under doubt and taking notes to clarify his understanding. It was the leading cause of stress and slowness in the shared analysis.

Considering this problem situation, SPI Diagnostic was a challenging process due to:

- fault of structure to perform the interview;
- high subjectivity, from the SEPG member, to take notes during the interview;
- high subjectivity from the other SEPG members to perform prior analysis based on the notes taken by another member; and
- the difficulty in analyzing a huge amount of information generated in the interview.

This work is motivated by challenges when performing a SPI Diagnostic. Firstly, SPI Diagnostic is generally conducted manually through questionnaires and interviews (MOREIRA et al., 2013). Based on this, the information gathering is a huge challenge, once it is the first key point in any SPI initiative. We also consider that SPI Diagnostic is

a subjective process that demands high knowledge level. Being performed by a SEPG, SPI Diagnostic demands a specific knowledge in software engineering and process reference models (LIMA; SANTOS, 2014). It implies a high effort in training and preparation, in order to conduct and implant SPI Diagnostic (PRIKLADNICKI; BECKER; YAMAGUTI, 2005), which reflects in a high implementation cost (MINELLA; THIRY; FERNANDES, 2015).

Related works, such as Dagnino e Cordes (2014), Gasca-Hurtado, Hincapié e Muñoz (2017), Moreira et al. (2013), Rodríguez et al. (2018), propose solutions to address the current status information gathering challenge. But, none systematize the way to specify or registry the gathered information. We consider this as an essential gap because there is a risk of a bias in the SPI Diagnostic result, without a clear orientation about how much information should be collected. As stated in the problem situation mentioned previously, this solution would help in the effort to analyze a vast data amount. Moreover, it minimizes the subjectivity of information analysis. Nonetheless, it does not address the subjectivity in the gathering.

Secondly, SPI Diagnostic is a complex task from a data analysis point of view, once SEPG must interpret all collected data (MOREIRA et al., 2013). One difficulty in performing this interpretation is handling a large data amount. The volume of knowledge, both theoretical and practical, is vast and sometimes much more detailed than needed (ZANNI-MERK; ALMIRON; RENAUD, 2011). Therefore, those difficulties impose a high cost to the company, which is a big problem, mainly for small and medium organizations (MOREIRA et al., 2013; HERRERA; RAMÍREZ, 2003).

There are proposals, such as, Moreira et al. (2013), Bozkaya, Gabriels e Werf (2009), which implements intelligent solution improve productivity. But, none of them provide support to handle the subjectivity intrinsic to the process. As mentioned in the problem situation, these solutions would not help the difficulties of structure and subjectivity to gather information during the interview. Moreover, it minimizes the effort to analyze a large amount of data, but it is not guaranteed.

Considering this, we ask the following questions:

1. How can one systematize the information gathering specification in SPI Diagnostic?
2. How can one minimize the data analysis subjectivity in SPI Diagnostic?

1.2 Goal

Our main goal is to develop a solution that helps SEPGs by providing an intelligent solution to analyze SPI Diagnostic information. In other words, we propose an intelligent software that guides the information gathering and classifies the collected information according to reference models. We decomposed it into the following specific goals:

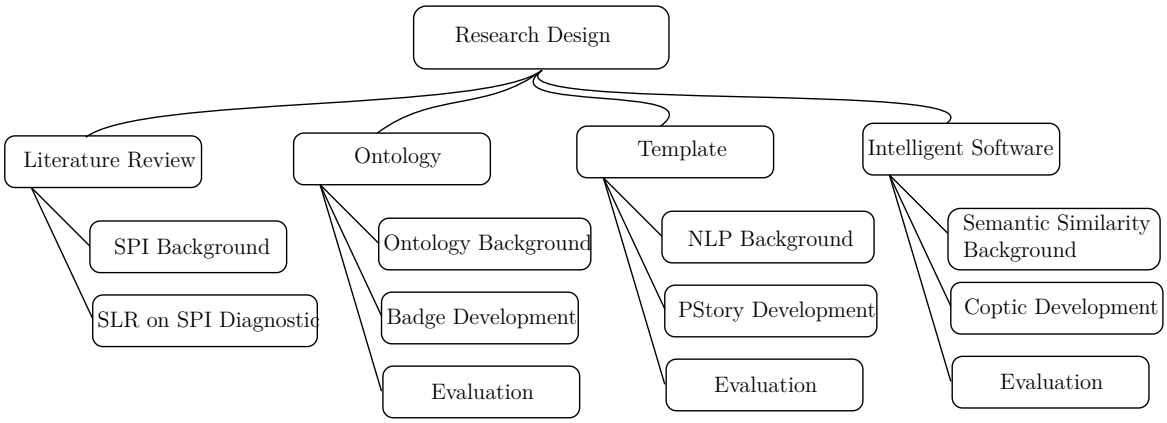
- investigating the state of the art of SPI Diagnostic proposals;

- proposing an ontology to generalize SPI resources;
- proposing a template to standardize the information gathering specification;
- developing intelligent software to provide support for evidence classification in SPI Diagnostic;
- performing evaluations of previously mentioned solutions.

1.3 Research Design

In Figure 2, we present the structure used in this research. For each goal, the first step was looking at and studying the related background. We start by conducting the Systematic Literature Review (SLR) aiming to find SPI Diagnostic solution proposals. Then, we start the development of an ontology to generalize SPI resources and a template to standardize the information gathering specification in parallel with the intelligent software development. We chose to develop an ontology due to it is an approach to structure knowledge and it is largely used in this context. Soon after, we also in parallel evaluated both proposals.

Figure 1 – Research design process



Source: Author.

1.4 Contributions

Considering the contributions of this work, we highlight 3 main contributions:

- proposing a way to generalize SPI resources, through a domain ontology;
- proposing a template, called PStory, to structure and formalize the specification of gathering diagnostic information; and
- a software that performs semantic analysis of PStories and performs the evidence-practice matching against process reference models.

Moreover, we would like to highlight some complementary results:

- the Systematic Literature Review about SPI Diagnostic solution proposals published in XLVI Latin American Computer Conference - 2020 (CLEI 2020) (ECAR et al., 2020a); and
- the proposed ontology to generalize SPI resources published also in CLEI 2020 (ECAR et al., 2020b).

1.5 Organization

This document is organized according to the following: We present Related Works in chapter 2, with the conduction and results of both SLR; We present the Domain Ontology in chapter 3, where we give the development of our Domain Ontology that generalize SPI resources; The PStory is presented in chapter 4; An intelligent software tool called Coptic is presented in chapter 6, the intelligent software that provides support to conduct the SPI Diagnostic; Conclusions are presented in chapter 7.

2 LITERATURE REVIEW

In this chapter, we present the Systematic Literature Review (SLR) according to the guidelines proposed by Kitchenham e Charters (2007), which aims to investigate solution proposals for SPI Diagnostic. It is organized as follows: Software Process Improvement (SPI) conceptual background is presented in section 2.1; other literature reviews are presented in section 2.2; SLR protocol is presented in section 2.3; SLR execution is presented in section 2.4; SLR result analysis is presented in section 2.5; threats to validity are presented in section 2.6; Lastly, we present chapter lessons in section 2.7.

2.1 SPI Background

According to Humphrey (1989), improving the development process may improve the software quality. Likewise, the SPI implementation helps to improve a different number of business goals, such as:

- enhancing the product quality;
- minimizing wasted time and effort;
- reducing the time to market;
- providing predictability (ISO, 2004).

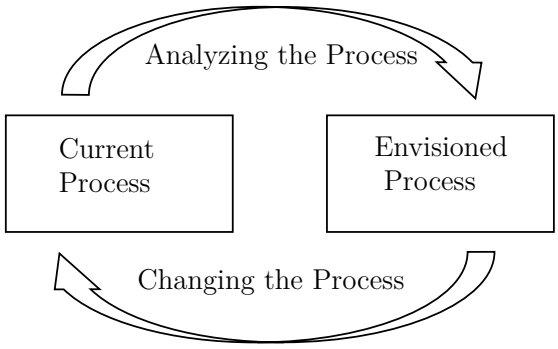
In this sense, there are two main approaches to conduct such improvement changes Münch et al. (2012). They are:

- **Model-based SPI approaches (top-down)**: compare the organization process with a reference model, for example (VASCONCELLOS et al., 2017):
  - CMMI(CHRISSIS; KONRAD; SHRUM, 2011);
  - ISO 15540 (ISO, 2004);
- **Continuous SPI approaches (bottom-up)**: address specific solutions for identified problems and evaluate specific effects from improvement programs (VASCONCELLOS et al., 2017), for example:
  - Quality Improvement Paradigm (QIP) (BASILI, 1989);
  - Total Quality Management (TQM) (MARTÍNEZ-LORENTE; DEWHURST; DALE, 1998);
  - Agile SPI (SALO; ABRAHAMSSON, 2005).

A generic model for a SPI initiative is composed of two phases, as shown in Figure 2. It is a cycle with two primary states. Looking at the activity "Current Process State", the process is analyzed and improved, resulting in an envisioned (or planned) process; Thus, this improved (and new) process assumes the value of the current process, and the cycles keep going on (STELZER; MELLIS, 1998).

One of the most basic processes for SPI is the Plan Do Check Act (PDCA) cycle. The PDCA cycle was proposed by Shewhart e Deming (1986), and it is commonly used as a problem-solving model in the context of quality management (DEMING, 2018). It is a model for conceptual process improvement (JOHNSON, 2002). As can be seen in

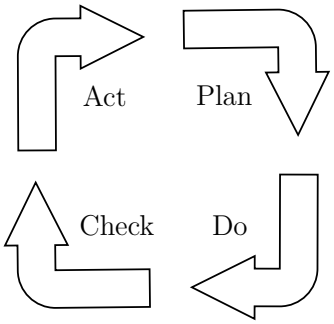
Figure 2 – A generic model of organizational change in software process improvement.



Source: Stelzer e Mellis (1998).

Figure 3, the "Plan" (P) is the starting point to quality improvement. Activities necessary to achieve the plan are implemented in "Do" (D). In "Check" (C), one checks the results to understand the causes of the problems. Actions are taken to improve the processes in "Action" (A) (DAHLGAARD; KRISTENSEN; KANJI, 1995).

Figure 3 – Plan Do Check Act (PDCA)



Source: Author.

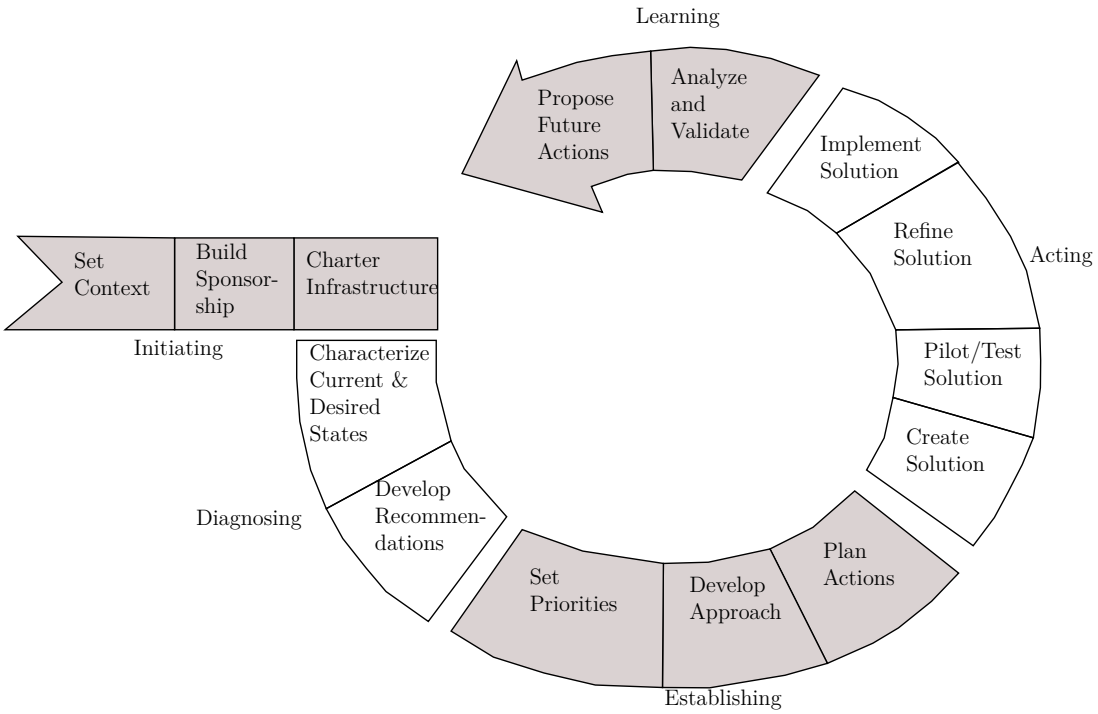
The IDEAL model (MCFEELEY, 1996) is an example of abstraction used to guide a SPI initiative. The model shown in Figure 4 depicts five phases of a SPI initiative, which provide a continuous loop through the steps necessary for SPI.

Initiating Phase: In this phase, one establishes the infrastructure, the roles and responsibilities for the infrastructure are initially defined, and initial resources are assigned. Moreover, one creates a SPI plan to guide the organization throughout the completion of the Initiating, Diagnosing, and Establishing phases (MCFEELEY, 1996).

Diagnosing Phase: This phase establishes the basis for the later phases. Here, the SPI action plan is initiated and guided by the organizational vision, strategic business plan, lessons learned from the past (i.e., from improvement efforts), key business issues faced by the organization, and long-range goals (MCFEELEY, 1996).

Establishing Phase: In this phase, one prioritizes the issues addressed by the organization with its improvement activities. Then the SPI action plan draft is completed.

Figure 4 – The IDEAL Model.



Source: adapted from McFeeley (1996).

During this phase, the measurable goals are developed from the general purposes defined in the Initiating phase (MCFEELEY, 1996).

Acting Phase: Here, solutions to address the areas for improvement discovered during the diagnosing phase are created, piloted, and deployed throughout the organization. One develops plans to execute pilots to test and evaluate the new or improved processes. The plans to accomplish the roll-out are developed and executed. After successful piloting, the processes are read for organization-wide adoption, deployment, and institutionalization (MCFEELEY, 1996).

Leveraging Phase: This phase contemplates several activities in parallel, solutions development, learned lessons, metrics on performance, and goal achievement. These artifacts are added to the process database to become an information source for personnel involved in the next pass-through model. Next, corrections or adjustments to the strategy, methods, or infrastructure can be made before starting (MCFEELEY, 1996).

2.1.1 SPI Diagnostic

The SPI Diagnostic is an activity that appears in a starting point of a SPI initiative and might be kept along with the cycles. SPI Diagnostic consists in identifying the real problems and what model is more appropriate to the current situation (SILVA; BRANCHER, 2017). According to Gasca-Hurtado, Hincapié e Muñoz (2017), SPI Diagnostic determines the processes maturity level of an organization and allows us to

recommend tools and techniques. Moreira et al. (2013) says that SPI Diagnostic is the verification process of identifying strengths and weaknesses to conduct improvement actions. SPI Diagnostic intends to discover the current status of an organization.

Based on this, we define SPI Diagnostic as the process to understand an organization and its processes, focusing on recommending improvement resources. Generally, SPI Diagnostic are conducted using *ad-hoc* strategies, based on own resources, knowledge, experience, skills, and abilities (GARTISER et al., 2014). In other words, each SEPG handles its way to conduct a SPI Diagnostic. In most cases, a high level of experience is determinant to conduct a successful SPI Diagnostic. One can expect a better and precise diagnostic from SEPG composed of senior professionals.

The SEPG generally uses questionnaires and interviews to retrieve information about the current status of an organization. SPI Diagnostic strategies are usually is made considering the improvement approach, such as, IDEAL (MCFEELEY, 1996; ANACLETO et al., 2004), or PDCA (QUINQUIOLO, 2002), or it is based on process-reference model, such as, CMMI (CHRISSIS; KONRAD; SHRUM, 2011), or MPS.BR (SOFTEX, 2011).

An example of a simple road-map to perform a SPI Diagnostic is:

- defining improvement goals;
- collect information in the highest level about the organization, such as mission statement, values, vision, core products, target public, organizational structure, among others;
- collect information about organization daily process, example, questionnaires, and interviews with organization human resources, directors board, employees;
- analyzing work products;
- analyzing collected resources against improvement goals demands;
- delivering a status report about analyzed data.

2.2 Others Literature Reviews

We looked for similar works in the main search bases, in an *ad-hoc* way, before performing this work. We found two related studies. Khan et al. (2017) present a tertiary study that aims to identify how many SLRs in SPI area had been published. Silva e Brancher (2017) mention SLR in their paper, published in 2017.

The first study, conducted by Khan et al. (2017), aims to find papers that are SLR in the field of SPI published between 2004 and 2015. They analyzed 7684 papers and selected 24. They grouped the result into 5 groups. 8 papers about factors, 6 papers about Small and Medium Enterprises (SMEs), 8 papers about process models, 1 paper about software quality and 1 paper about software testing.

Considering the result found by Khan et al. (2017), we analyzed the purpose and research questions of each study. Based on our analysis, none of them share the same goal

or answer our research questions. Hence we decided to perform a specific SLR to look for studies that answer our research questions.

The second study, conducted by Silva e Brancher (2017), performs the search for papers between 2011 and 2016. Their SLR also presents similar protocol and goals. The SLR resulted in 13 selected papers. Considering that the paper goal is not presenting the SLR, but their SPI Diagnostic solution, the authors do not provide a deep SLR result discussion and analysis as our contribution presents. They classified the selected papers into two groups:

- papers presenting indicators to identify organizational strengths and weaknesses, and;
- papers presenting adhesion degree analysis of the process under development with the reference model.

Silva e Brancher (2017) does not provide a deep result analysis and discussion; thus, we decided to conduct a new SLR and analyze the result in depth.

2.3 Review Protocol

The first step was to establish the study SLR scope, to do that, we applied the Goal Question Metric (GQM)[1]. Next we present the our main goal:

Analyze	SPI studies
for the purpose of	find the state of the art
with respect to	diagnostic solutions
from the point view of	researcher
in the context of	scientific papers.

2.3.1 Research Questions

The goal of this study is to investigate the proposed solutions for SPI diagnostic. We consider a solution any proposal that deals with the difficulty to perform a SPI Diagnostic. Thus, a solution may be a software, process, framework, model, method, methodology, etc.

Based on the goal, we derived our research questions by applying GQM:

- **RQ01** - What problems motivated the studies about SPI Diagnostic?
- **RQ02** - What solutions were developed to solve the reported problems?

2.3.2 Search and Selection Process

We performed a snowballing technique as our primary strategy for retrieving papers[2]. According to Jalali e Wohlin (2012), snowballing can yield similar results to

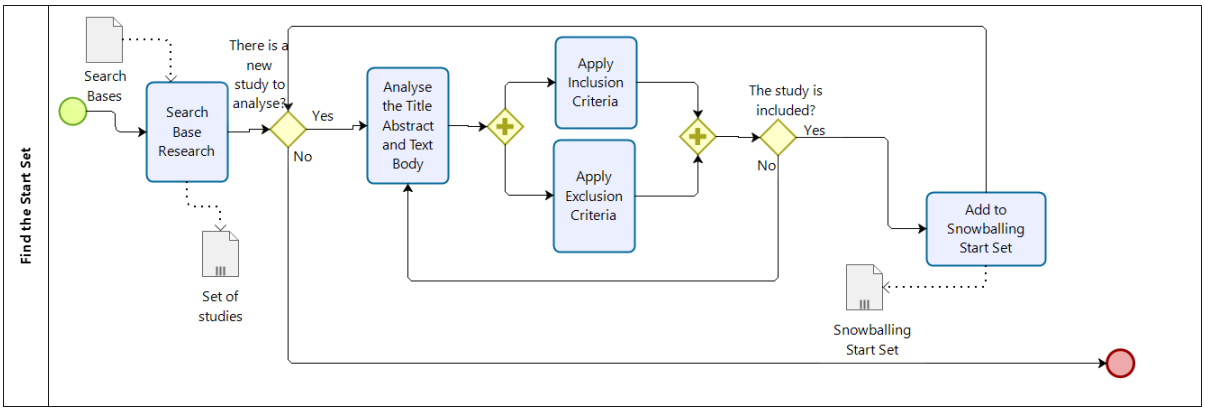
[1] GQM is a measurement approach that focuses on defined goals (SOLINGEN; BERGHOUT, 1999)

[2] The snowballing technique refers to using study references and citations to identify additional studies (WOHLIN, 2014)

database searches.

The first snowballing step is to define the start set, which must be formed of relevant studies for review goals (WEBSTER; WATSON, 2002). A good snowballing start set should come from different communities, covering several publishers, authors, and years (WOHLIN, 2014). The Figure 5 shows the process used to find a good starter set. Firstly, we performed research on search bases, then, from the set of studies retrieved, inclusion and exclusion criteria are applied. The result is the snowballing start set.

Figure 5 – Find the start set process.



Source: Author.

After finding the snowballing start set, we performed the snowballing. The Figure 6 shows the process used to conduct the snowballing. Based on the start set, we performed two iterative cycles, the backward and forward:

- **backward snowballing** consists in investigating all studies cited by a study from the start set;
- **forward snowballing** consists in investigating all studies that cite a study from the start set (WEBSTER; WATSON, 2002).

The backward and forward snowballing should be performed for each paper at once until there are no more candidates for inclusion. It is also essential to decide on either inclusion or exclusion before using a new study for snowballing.

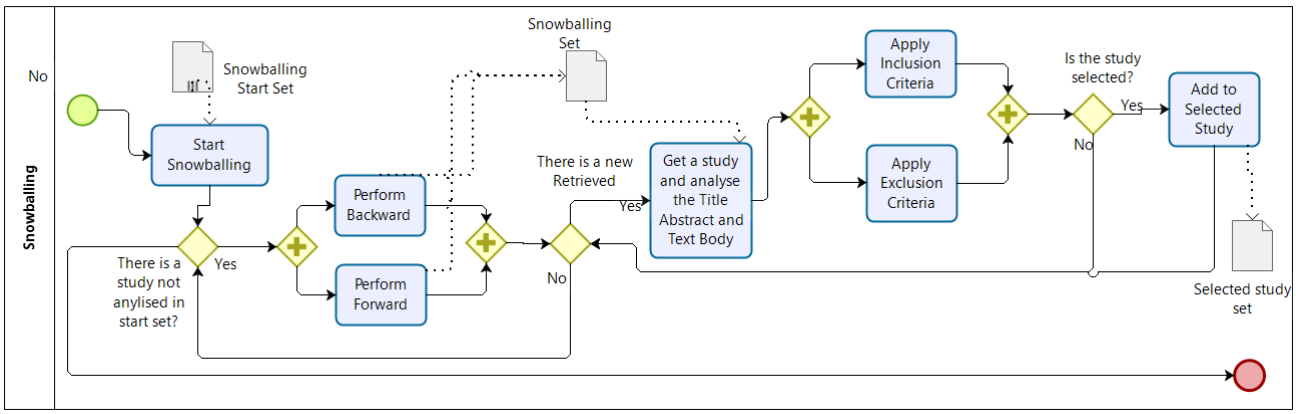
After that, we looked for terms in the title, keywords, or abstract in each paper. The terms to look for are: `diagnostic`, `diagnosis`, `diagnose`, `assessment`, `analysis`, `SPI`, `Software Process Improvement`. When we find at least one of these words, we read the full text and apply the inclusion and exclusion criteria. For each paper, we set at least one inclusion or exclusion criteria. The paper is accepted when we set at least one inclusion criteria and no exclusion criteria.

The inclusion criterion is:

- peer-reviewed papers which propose a solution for SPI Diagnostic.

The exclusion criteria are:

Figure 6 – Snowballing process.



Source: Author.

- papers which not found in indexed search bases;
- secondary or tertiary studies, such as systematic literature mappings, reviews, or surveys.

2.3.3 Data Extraction and Analysis Strategy

Following the GQM methodology, we defined the metrics needed to answer research questions. After reading each paper, firstly, we collected bibliometric data. Then, we extracted the following key text snippets:

- the motivation by looking at the introduction and conclusion sections;
- the solution proposal by looking in introduction and methodology or development sections;

After that, we applied the unitarization and clusterization steps from content analysis[3] to answer the research questions.

2.4 Review Execution

We ran the review execution by running the process presented in Figure 5 on March 18, 2019. The result presented in Table 1 was obtained through the following generic search string:

```
TITLE-ABS-KEY(((spi) OR (software AND process
AND improvement)) AND diagnostic).
```

We analyzed the title abstract and text body to apply the inclusion and exclusion criteria. We added the paper in the Snowballing start set if the paper is checked in any

[3] content analysis is a strategy to retrieve qualitative information(NEUENDORF, 2016)

Table 1 – Retrieved papers by search base.

Search Base	Result
Scopus	107
IEEE Xplore	40
Compendex	177
ACM DL	28
Springer Link	14
ScienceDirect	47
Total	**413**

Source: Author.

inclusion and non-exclusion criteria. After the classification step, we included four papers in the starter set.

We conducted snowballing running backward and forward between March 18-20, 2019, following the including and excluding criteria. We analyzed about 445 papers after five rounds. The Table 2 shows the result of each round. Round 0 represents the search for the starter set. As mentioned, the start set was composed of 4 papers. Then, we start the backward and forward iterations in rounds 1, 2, 3, 4, 5, until there are no more papers selected, as shown in the Selected column of Round 5.

Table 2 – Snowballing rounds.

Round	Retrieved	Selected
0	413	4
1	139	7
2	117	1
3	135	1
4	15	1
5	39	0
Total	**445**	**14**

Source: Author.

The selection process was peer-reviewed by two reviewers, where each reviewer applied the including and the excluding criteria for all papers after duplication detection. The paper was accepted when both reviewers selected the paper. When the reviewer included a paper and other reviewer excluded, they discussed to obtain consensus.

Through snowballing, we selected ten more papers beyond the four papers used as the starter set, resulting in a final set of 14 papers. We present the final list of selected papers in Table 3.

Table 3 – List of accepted papers.

ID	Paper	Author	Year
P01	Sarasvati: Diagnostic method for software process improvement	Silva e Brancher (2017)	2017
P02	Software process improvement assessment for multimodel environment tool to diagnose an organization	Gasca-Hurtado, Hincapié e Muñoz (2017)	2017
P03	GAIA Risks: A risk management framework	Gaffo e Barros (2012)	2012
P04	Tool to assess the maturity level of the risk management of a software development process	Gaffo et al. (2013)	2013
P05	A Model to Measure Organizational Readiness for Software Process Improvement	Dagnino e Cordes (2014)	2014
P06	A semantic layered architecture for analysis and diagnosis of SME	Gartiser et al. (2014)	2014
P07	Diagnóstico de processos em organizações intensivas em software usando um sistema especialista	Minella, Thiry e Fernandes (2015)	2015
P08	Autodiagnóstico de Processo de Software Baseado em Sistema Especialista	Moreira et al. (2013)	2013
P09	Diagnóstico al iniciar la mejora de proceso de software	Trujillo-Casañola et al. (2014b)	2014b
P10	Modelo Si.MPS.CU para valorar las organizaciones al iniciar la mejora de proceso de software	Trujillo-Casañola et al. (2014a)	2014a
P11	KAIRÓS: Intelligent System for Scenarios Recommendation at the Beginning of Software Process Improvement	Rodríguez et al. (2018)	2018
P12	Commitment to Software Process Improvement - Development of Diagnostic Tool to Facilitate Improvement	Abrahamsson (1999)	1999
P13	Process Diagnostics: a Method Based on Process Mining	Bozkaya, Gabriels e Werf (2009)	2009
P14	A Multi-agents System for Analysis and Diagnosis of SMEs	Zanni-Merk, Almiron e Renaud (2011)	2011

Source: Author.

2.5 Result Analysis

We analyze the results based on the SLR research questions.

2.5.1 What problems motivated the studies about SPI Diagnostic?

With this questioning intends to analyze what problems motivate the SPI Diagnostic solution proposal. We extracted from the papers the text snippet and synthesized it in the commonly mentioned problems.

As one of the mentioned problems, we highlight the lack of support to visualize the processes deficiency, highlighted in P05. Another problem is handling the vast amount of data needed to analyze and diagnose a company, highlighted in P06 and P14. Another problem is the elevated cost in the evaluation activity to diagnose a company, highlighted in P08. Another motivation is the difficulty of measuring the commitment from the involved people in a SPI initiative.

Another problem highlighted in P13 is related to deal with a short period to implant a SPI initiative. Another motivation highlighted in P02 is the difficulty of integrating a multi-model, such as combining agile and traditional approaches. P11 highlights the problem to recommend improvement scenarios using success factors and good practices to support decision-making.

The unmentioned papers do not highlight a specific problem. They deal with the need to increase productivity in the SPI initiative as a whole, focusing on the SPI Diagnostic process.

2.5.2 What solutions were developed to solve the reported problems?

As shown in Table 4, P01, P02, P03, P09, P10, and P12 are solutions based on specific questionnaires. P02, P04, P06, P07, P08, P11, and P14 are software tools solutions.

Moreover, we highlight P06, P07, P08, P11, and P14 which implements some Artificial Intelligence (AI) technique to add intelligence into the software. P05 presents a model called EASE, which aims to quantify the readiness of an organization to commit to a SPI initiative. P13 is a solution that uses data mining to process a massive amount of data to obtain the primary needed information.

As a solution for SPI diagnostic, we considered anything aiming at supporting the diagnostic task. In this sense, a solution can be a model, method, framework, software, methodology, technique, process, etc. In the following, we summarize what is the proposed solution in each selected paper.

Silva e Brancher (2017) presents a method called Sarasvati. This method creates a company map based on information obtained by a questionnaire represented in a radar chart.

Aiming at integrating good SPI practices when using more than one reference model or framework, Gasca-Hurtado, Hincapié e Muñoz (2017) presents software that facilitates the company diagnostic.

Table 4 – Solution proposals.

ID	Author	Questionnaire	Tool	Data Mining
P01	Silva e Brancher (2017)	X		
P02	Gasca-Hurtado, Hincapié e Muñoz (2017)	X	X	
P03	Gaffo e Barros (2012)	X		
P04	Gaffo et al. (2013)		X	
P05	Dagnino e Cordes (2014)			
P06	Gartiser et al. (2014)		X	
P07	Minella, Thiry e Fernandes (2015)		X	
P08	Moreira et al. (2013)		X	
P09	Trujillo-Casañola et al. (2014b)	X		
P10	Trujillo-Casañola et al. (2014a)	X		
P11	Rodríguez et al. (2018)		X	
P12	Abrahamsson (1999)	X		
P13	Bozkaya, Gabriels e Werf (2009)			X
P14	Zanni-Merk, Almiron e Renaud (2011)		X	

Source: Author.

Gaffo e Barros (2012) presents a risk management framework called GAIA Risks, which comprises five maturity levels, a deployment process, a diagnostic questionnaire, and process metrics.

Gaffo et al. (2013) presents a software based on GAIA Risks with the following features:

1. Automating data collection and automated calculation and chart projection.
2. Collecting data from other areas, such as human resources and information and communication technology (TIC).
3. Providing the diagnostic evaluation results online.

Dagnino e Cordes (2014) presents a model called EASE, which aims to quantify the risk and measure the mitigation process along the course of the SPI effort. It also quantifies the readiness of an organization to commit to a SPI initiative. The EASE tool was explicitly developed for assessing the readiness of an organization to engage in a SPI activity seriously. Anyway, it can be used in any other organizational change situation as well.

Gartiser et al. (2014) presents a software based on a layered semantic architecture for a knowledge-based system. The presented work has two main modules: a diagnostic module and a recommendation module according to the specified goals. The purpose of the developed software is to assist the SEPG in the process of thinking and reasoning, in their task of diagnosing and supporting SMEs.

Minella, Thiry e Fernandes (2015) presents a software based on an expert system to support the organization diagnostic. The system captures the development life cycle, mapping how the organization works. In this sense, the system indicates strengths and

weaknesses (MINELLA; THIRY; FERNANDES, 2015).

Moreira et al. (2013) presents software built on an expert system for self-diagnosis. It is a web tool that realizes the self-diagnosis to be performed by any development team member or tertiary process improvement company. This self-diagnosis approach was designed based on a process that includes characterization and organizational factors questionnaires. This approach allows one to evaluate the organization at any time, according to development team availability (MOREIRA et al., 2013).

Trujillo-Casañola et al. (2014b) presents a model to assess organizations starting a SPI initiative, based on indicators and metrics. This model considers experts' experience to help them mitigate the negative impact of the critical success factors. This approach helps to identify strengths and weaknesses to undertake change and facilitates the risk analysis (TRUJILLO-CASAÑOLA et al., 2014b).

Rodríguez et al. (2018) present KAIRÓS, intelligent software for scenarios recommendation at the beginning of SPI. This recommendation can be used as a source for organization diagnostic. By integrating artificial intelligence techniques, KAIRÓS automates the processing of Critical Success Factors and Good Practices combined.

2.6 Threats to Validity

We report the threats to validity based on the Petersen, Vakkalanka e Kuzniarz (2015) classification.

Theoretical validity: Risk is related to data extraction and bias analysis. Conceptual background capability may affect the researcher's decisions and compromise the results. To mitigate this, we performed peer data extraction and peer-reviewing, where each researcher performed its data extraction and compared with each other.

Interpretive validity: We highlight a risk related to our searching process. The ability to find relevant studies directly impacts the SLR results. Considering this, we applied a snowballing technique, which had its effectiveness demonstrated by Jalali e Wohlin (2012). Another risk is related to the quality of the start set used to initiate the snowballing. We performed systematic research in several search bases to select a good start set to mitigate this. Nevertheless, there is the possibility we have had unrecognized some relevant studies in our searching process.

Repeatability: Another risk to be considered is related to the selection process. When applying inclusion and exclusion criteria, one may not evaluate the studies in-depth and, therefore, may produce false positives and false negatives. To mitigate the risk related to the screening strategy, we performed a peer review process.

2.7 Chapter Lessons

In this chapter, we introduced the conceptual background about SPI area and provided a SLR about solution proposals to SPI Diagnostic. It attends the first specific goal of investigating state of the art. Considering this, we highlight the main findings:

- there is a lack of support to visualize the processes deficiencies;
- there is difficult to handle the huge amount of data needed to analyze and to diagnose a company;
- the elevated cost in the evaluation activity to diagnose a company;
- main solutions propose questionnaires and specific analysis from these questionnaires collected data;
- few solutions propose supporting software tools.

We also highlight the main gaps found in related works:

- proposals are generally linked to specific process reference models;
- there are no proposals to structure the information gathering specification to standardize the collected data.

Based on the gaps mentioned previously, our work focus on proposing a template to standardize the collected information specification in the gathering diagnostic process, in chapter 4. We also focused in to propose a domain ontology to allow the creation of several process reference models in chapter 3. Moreover, we are linking these two solutions in software that performs an intelligent analysis of collected evidence in chapter 6.

3 BASE OF KNOWLEDGE ABOUT SE PRACTICES

In this chapter, we present the Base of Knowledge about Software Engineering Practices (Badge) Ontology that is an ontology that aims to generalize SPI resources. The Badge has been developed according to Ontology Development 101 (NOY; MCGUINNESS et al., 2001). This chapter is based on the paper accepted to be published in CLEI 2020. It is organized as follows: section 3.1; the state of the art is presented in section 3.2; Badge ontology is presented in section 3.3; evaluation is presented in section 3.4; Lastly, we present chapter lessons in section 3.5.

3.1 Ontology Background

The term ontology comes from a branch of philosophy that deals with the nature of Being. Artificial intelligence researchers introduced it in computer science, who constructed computer models with some automated reasoning. Ontologies began to be treated as an integral part of knowledge-based systems (GUIZZARDI, 2005), being defined as an explicit specification of a conceptualization, in other words, a simplified view of what needs to be represented for some reason (GRUBER, 1993).

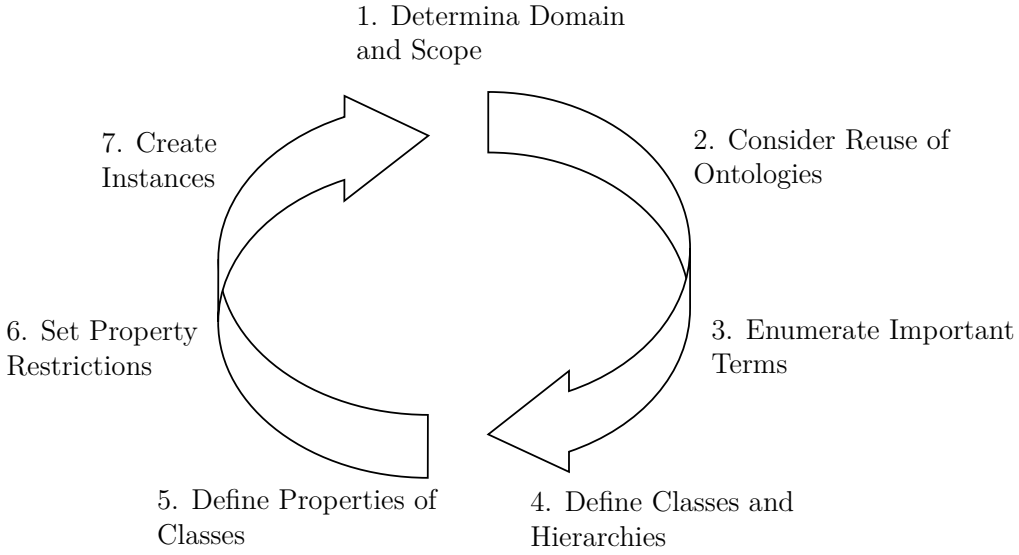
Later on, the ontology definition was defined as a formal and explicit specification of a shared conceptualization (BORST, 1997). In the context of computer and information science, an ontology defines a set of representational primitives in a particular knowledge area. Representational primitives are fundamentally classes, attributes, and relationships, including their meanings and restrictions that provide logically consistent applications.

There are several ways to model a domain, but usually, ontology development is performed through an iterative process. Additionally, the current application influences the modeling. Among the existing methodologies, we stand out Ontology Development 101 (see Figure 7), which establishes clear and objective procedures to build ontologies in a simple way (NOY; MCGUINNESS et al., 2001). The procedures are described below:

- *Determine Domain and Scope* defines the universe of concepts to be mapped, the purpose that motivates the model, and the questions the ontology should answer.
- *Consider Reuse of Ontologies* proposes to search and reuse ontologies that satisfy the domain and scope before effectively developing a new one.
- *Enumerate Important Terms* leads a complete survey of terms related to the domain of discourse, using the knowledge of experts and existing documentation as support.
- *Define Classes and Hierarchies* proposes the refinement and organization of terms raised in a taxonomy, which establishes the set of concepts needed to map the domain.
- *Define Properties of Classes* analyzes and refines each class with attributes that characterize the concept, ensuring that the model can meet the scope.

- *Set Property Restrictions* proposes the definition of constraints for the properties, such as domain, range, and cardinality.
- *Create Instances* performs the creation of individuals for the ontology, enabling it to complete the verification and validation of a model.

Figure 7 – An ontology engineering method.



Source: Author.

3.2 Related Ontologies

We performed an *ad hoc* search to look for related ontologies. The Table 5 shows ontology proposals for SPI resources harmonization.

Table 5 – Related Ontologies

ID	Author	Models	
P01	Ferchichi, Bigand e Lefebvre (2008)	An Ontology for Quality Standards Integration in Software Collaborative Projects	CMMI + ISO 9001
P02	Pardo et al. (2012)	An ontology for the harmonization of multiple standards and models	COBIT 4.1 + Basel II + VAL IT + RISK IT + ISO 27002 + ITIL
P03	Pardo-Calvache et al. (2014)	A reference ontology for harmonizing process-reference models	CMMI-ACQ and ISO 9001

Source: Author.

The ontology proposed by Ferchichi, Bigand e Lefebvre (2008) aims to integrate several processes using a common ontology from various perspectives. They integrated

two quality models, ISO 9001:2000 and CMMI. The goal is to generate a multi values ontology allowing double certification from these two models.

Both works presented in Pardo et al. (2012) and Pardo-Calvache et al. (2014) propose two ontologies the aim to harmonize quality models. The work presented in Pardo et al. (2012) proposes an ontology called H2mO that aims to support the harmonization of multiple models. The work presented in Pardo-Calvache et al. (2014) proposes a sub-ontology called PrMO that complements H2mO. It establishes and clarifying the key process elements to support the harmonization of multiple models through homogenization of their process structures.

3.3 Badge Development

Eventually, SPI initiatives might evolve more than one reference model. For example, an organization aims to implement CMMI and MPS.BR at the same time. In this case, it is convenient to harmonize the chosen models to reach the expected goals. Reference models harmonization should generalize the common aspects and specialize the distinct elements. It should be performed respecting the particularities and aspects of the selected models and also respecting the characteristics of SPI initiatives.

Firstly, according to Ontology Development 101 (NOY; MCGUINNESS et al., 2001), we defined the domain and scope for Badge. The Badge represents the knowledge need to generalize the main SPI resources that say "what should be done" in a software process. Badge can answer the following questions:

- What software engineering practices exist?
- How can these practices be clustered?
- How can these practices be ranked?

Secondly, we considered the reuse of models presented in related work. It was not possible to perform direct reuse (e.g., importing the OWL specification). However, we performed conceptual reuse of the followed ideas:

- from Ferchichi, Bigand e Lefebvre (2008), we reused the following concepts:
  - **QualityStandard**
  - **Practice**
  - **MaturityLevel**
- from Pardo et al. (2012), we reused the following concepts:
  - **QualityModel**,
  - **ProcessCategory**
  - **ProcessGroup**

Thirdly, we extracted the principal terms from the main SPI resources. For this, we analyzed CMMI, MPS.BR, Software Engineering Body of Knowledge (SWEBoK), Project Management Body of Knowledge (PMBoK), and agile manifesto. We performed this step-through brainstorming with specialists and Software Engineering (SE) researchers,

composing ideas, and filtering the most relevant terms for SPI context. The terms are shown in Table 6

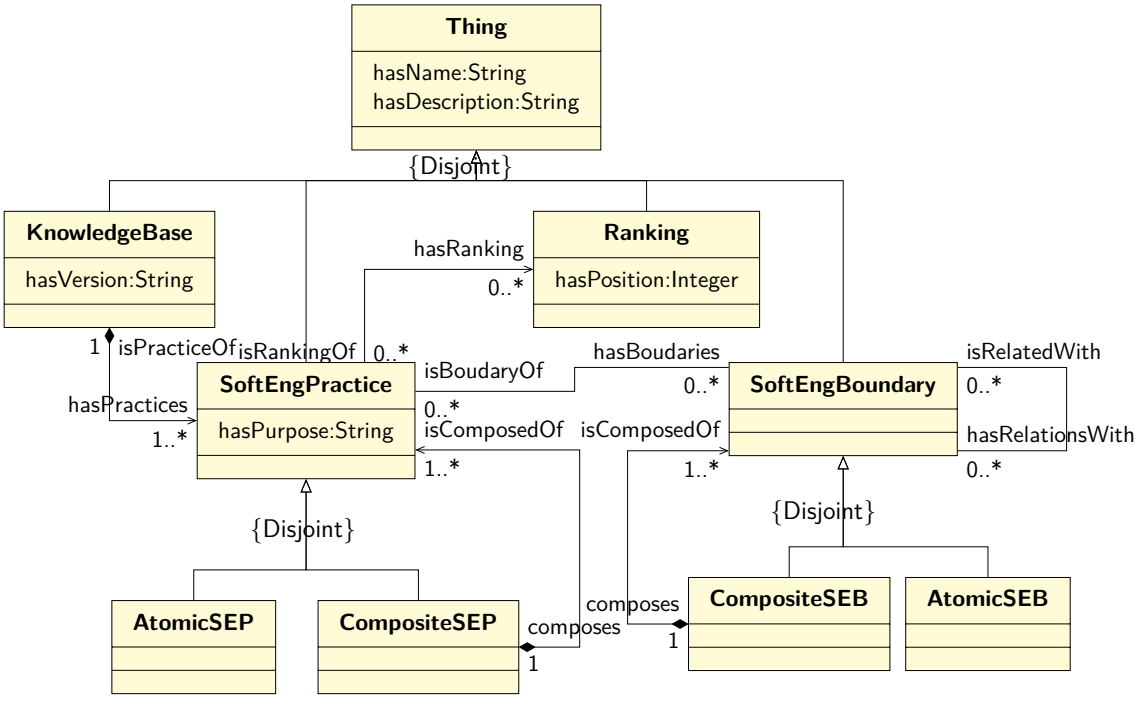
Table 6 – SPI resources terms

Term	Resource
CMMI	Process Area, Category, Generic/Specific Goal, Generic/Specific Practice, Maturity Level
MPS.BR	Process Attribute, Process, Expected Result, Maturity Level
SWEBoK	Knowledge Area, Topic, Sub-Topic
PMBoK	Process Area, Process
agile manifesto	Value, Principle

Source: Author.

According to Ontology Development 101 (NOY; MCGUINNESS et al., 2001), in the following steps, one should define the class hierarchy, classes properties, and property restrictions. We present in Figure 8 the result of these steps execution in a conceptual model specified in Unified Modeling Language (UML).

Figure 8 – Badge Ontology



Source: Author.

Software Engineering Knowledge Base (KnowledgeBase): The group of concepts demanded in SPI initiatives. It is the top-level concept in ontology. For this concept, we reused the ideas of QualityStandard from Ferchichi, Bigand e Lefebvre (2008) and QualityModel from Pardo et al. (2012). The Table 7 shows the Description Logics (DL) specification for KnowledgeBase.

Table 7 – Badge - KnowledgeBase DL specification

$KnowledgeBase \sqsubseteq Thing$
$KnowledgeBase \sqsubseteq \geq 1\ hasPractices\ SEP$

Source: Author.

Software Engineering Boundary (SEB): Is a software engineering discipline. A supergroup that wraps common concepts. A particular area or field that includes related aspects. Anything that groups software engineering concepts. **Atomic SEB (AtomicSEB):** A Boundary that there have no "sub Boundaries". A leaf in the hierarchical structure. **Composite SEB (CompositeSEB):** A Boundary is decomposed into other Boundaries. A Boundary that contains "sub Boundaries", Atomic or Composite. For this concept, we reused the ideas of ProcessCategory and ProcessGroup from Pardo et al. (2012). The Table 8 shows the DL specification for SEB, AtomicSEB, and CompositeSEB.

Table 8 – Badge - SEB DL specification

$SEB \sqsubseteq Thing$
$SEB \sqsubseteq CompositeSEB$
$SEB \sqsubseteq \geq 0\ isBoundaryOf\ SEP$
$SEB \sqsubseteq \geq 0\ hasRelationsWith\ SEB$
$SEB \sqsubseteq \geq 0\ isRelatedWith\ SEB$
$CompositeSEB \sqsubseteq SEB$
$CompositeSEB \sqsubseteq \geq 1\ isComposedOf\ SEB$
$AtomicSEB \sqsubseteq SEB$
$AtomicSEB \sqsubseteq \geq 0\ SEP$
$AtomicSEB \sqsubseteq \geq 0\ AtomicSEB$

Source: Author.

Software Engineering Practice (SEP): A Practice that attends to one or more Practices. A Procedure that can be decomposed into several actions or tasks. **Atomic SEP (AtomicSEP):** A Practice that there have no "sub Practices". A leaf in the hierarchical structure. **Composite SEP (CompositeSEP):** A Practice that is decomposed into other Practices. A Practice that contains "sub Practices", Atomic or Composite. For this concept, we reused the ideas of Practice from Ferchichi, Bigand e Lefebvre (2008). The Table 9 shows the DL specification for SEP, AtomicSEP, and CompositeSEP.

Ranking (Ranking): A stage that classifies the adherence level of related Boundaries and Practices. Levels may classify the processes in scaled stages. For this concept, we reused the ideas of MaturityLevel from Ferchichi, Bigand e Lefebvre (2008). The Table 10 shows the DL specification for Ranking.

Table 9 – Badge - SEP DL specification

$SEP \sqsubseteq Thing$
$SEP \sqsubseteq = 1\ isPracticeOf\ KnowledgeBase$
$SEP \sqsubseteq \geq 0\ hasRanking\ Ranking$
$SEP \sqsubseteq AtomicSEP$
$SEP \sqsubseteq CompositeSEP$
$SEP \sqsubseteq \geq 0\ hasBoundaries\ SEB$
$CompositeSEP \sqsubseteq SEP$
$CompositeSEP \sqsubseteq \geq 1\ isComposedOf\ SEP$
$AtomicSEP \sqsubseteq SEP$

Source: Author.

Table 10 – Badge - Ranking DL specification

$Ranking \sqsubseteq Thing$
$Ranking \sqsubseteq \geq 0\ isRankingOf\ SEP$

Source: Author.

3.4 Badge Evaluation

We applied the Focus Group Technique[1] to empirically evaluate Badge, according to the guidelines proposed by Kontio, Lehtola e Bragge (2004).

3.4.1 Planning

We started our planning by defining the focus group scope by writing it as a GQM goal, as follows:

Analyze	the Badge Ontology
for the purpose of	analyze the generalization capability
with respect to	SPI resources that say “what should be done”
from the point view of	researchers
in the context of	subjects instantiating SPI resources into Badge.

Once we established the focus groups scope, we defined the subject profile as following:

- all subjects should have:
  - a degree in a computer-related area;
  - knowledge in SPI;
- at least on the subject should have:
  - practice experience in SPI initiative.

[1] Focus groups can provide valuable, complementary empirical experience quickly and at low cost in the software engineering context (BLOOR, 2001; KONTIO; LEHTOLA; BRAGGE, 2004; KONTIO; BRAGGE; LEHTOLA, 2008).

Following, we defined the roles and responsibilities. The Moderator is responsible for instigating and limiting the discussions of the subject. The Supporter is responsible for providing and distribute resources, help and limiting the Moderator.

Then, we established the following roadmap:

1. **opening**, in which the moderator introduced general information about the session;
2. **background**, in which the moderator provided a conceptual foundation about Badge and SPI;
3. **discussion**, in which the subjects evaluated the models and provided feedback;
    a) an analysis between Badge and CMMI;
    b) an analysis between Badge and SWEBoK;
    c) an analysis between Badge and Agile Manifesto;
    d) an analysis between Badge and Scrum;
4. **closure**, in which the moderator collected the final opinions and thanked the subjects.

In each discussion, the subjects should analyze and discuss the presented models' generality capability. In other words, the subjects should discuss if it is possible to use Badge to generalize the models' concepts. They should argue if there is in Badge a concept that generalizes the specific model concept. From other points of view, the subjects should analyze if it is possible to inherit or extend a specific model concept from Badge.

The fourth discussion was used as the control group to avoid biased analysis. The Scrum model should not be able to be generalized by Badge. In this way, if the subjects agree that Scrum extends Badge, we can infer that the subject's interpretation was skewed.

All the tasks were planned to be performed as simply as possible. For this reason, we prepared all instrumentation based on sheets and pencils. For each discussion, we prepared pieces of the sheet of the conceptual model and the glossary of terms for Badge and the conceptual model, and the glossary of terms for each specific model. We also prepared a piece of sheet with the two conceptual models, Badge and the specific model, to enable the subjects to trace the inheritance lines and collect the final answer.

Finally, we planned to provide a complete document for each model presented in the data show projection for further consulting. Moreover, we defined that the subjects could consult any resource on their mobile device or laptop.

3.4.2 Execution

The focus group session was held on November 27, 2019, beginning at 09:15 am and ending at 11:48 am. We used a reserved room containing audio recording equipment, a projector, a whiteboard, and pens. We also provided printed documents describing a Badge Ontology description, conceptual background, forms, and all subjects were free to use the internet to look for related content.

We invited four subjects to run this focus group session. The subjects have heterogeneous profiles to allow discussions from different perspectives. For this reason, we invited practitioners and academics. **Subject 1 (S1)** is a practitioner with more than twenty years in the industry. He has experience as programmer, software project manager, SPI consultant working on CMMI and MPS.BR implantation and Agile Transformation. **Subject 2 (S2)** is a Software Engineer with more than five years as a practitioner in a software development company. He has a Master's degree in Computer Science and Data Science field. **Subject 3 (S3)** is a Software Engineer and Master's degree student with three years of experience as a programmer. **Subject 4 (S4)** is a Software Architect with more than ten years of experience in the industry as a Technical Leader.

3.4.2.1 Opening Report

The moderator started the session by welcoming and thanking the participants and giving them instructions about how to perform the focus group. The subjects were instructed to work by themselves as a discussion group, noting that the moderator could intervene to keep the work flowing, but the idea was that they worked with no interference. The moderator also highlighted that every member was free to ask questions and express their opinions during the evaluation. At this moment, the subjects signed the consent form.

3.4.2.2 Background Report

After the opening step, the moderator presented a brief background about the focus group theme and how the tasks should be performed. The moderator also provided a glossary of the key terms to be used during the focus group session. After that, we started the discussions steps.

3.4.2.3 First Discussion Report

The moderator starts the first discussion by asking if Badge generalizes the CMMI model.

S1 started saying that as he has significant experience with CMMI, so he was comfortable explaining the model to those who were not familiar with it. S1 explained the CMMI model details and the group started to argue about the generalization of the concepts in Badge.

S2 said that he needs to start scratching the sheet to visualize the generalization possibility. Based on this, the group elected S2 as the responsible for drawing the answer.

The supporter suggested opening the CMMI document on the data show to help the subjects to understand some concepts.

S1 said that the hierarchical organization of CMMI and Badge are similar and that the first concept that he can identify is that class **CMMI** is an extension of **KnowledgeBase**. S2 and S4 agreed with S1.

S1 said that in his understanding **ProcessArea** is a kind of **SEB**. S2 and S3 agreed and discussed about **ProcessArea** is a kind of **CompositeSEB** or a kind of **AtomicSEB**. S2 argued that **ProcessArea** could be a kind of **CompositeSEP**. At this point S1 started to provide some details about CMMI and defend that **ProcessArea** is closer to **CompositeSEB**. All subjects agreed with S1 and started discussing the next concept.

S3 said that in his point of view **SpecificGoal** is a kind of **CompositeSEP**, **SpecificPractice** is kind of **AtomicSEP**, **GenericGoal** is kind of **CompositeSEB** and **GenericPractice** is kind of **AtomicSEB**. None of the other subjects disagreed with him.

S3 also said that **Level** is clearly kind of **Ranking**. This concept was also a consensus.

Then the subjects started to discuss how **Category** fits in the context. S1 said that **Category** is kind of **CompositeSEB**. S4 argued that the concept of **Category** is not clear for him. S1 used the CMMI document do explain what is **Category** in CMMI. S4 agreed that it is kind of **CompositeSEB**. S2 disagreed of the relation among **Category**, **ProcesArea** and **CompositeSEB**. At that point, the supporter interfered and called attention that it is modeled in Badge using the Composite design pattern to promote the creation of a hierarchical structure. After that explanation S2 understand the relation among those concepts and agreed that **Category** is a kind of **CompositeSEB**.

At Last S2 paid attention again to concept of **Level** and the relation with **GenericPractice** and **ProcessArea**. He argued that it differs in Badge. Considering these doubts, S1 explained how it works in CMMI and that the relationship does not miss this meaning because the concepts keep being related by transitivity.

Finally, they cleared the final answer and finished the first discussion. The final discussion result is presented in Table 11.

3.4.2.4 Second Discussion Report

The moderator starts the second discussion by asking if Badge generalizes the SWEBoK model.

S2 started saying that it seemed too simple and asked the meaning of each concept in the model. The moderator opened the document with the analytical structure of SWEBoK. The subjects started to discuss the structure.

S1 said that **SWEBoK** is clearly kind of **KnowledgeBase**. All subjects agreed.

S2 started saying that **KnowledgeArea** may be **CompositeSEB** but he is not sure. S3 and S4 agreed with S2, but they had doubts about the relationship with **Topic**.

Table 11 – First discussion result.

CMMI Concept	Badge Generalization
CMMI	KnowledgeBase
Process Area	AtomicSEB
Category	CompositeSEB
Level	Ranking
Generic Goal	CompositeSEB
Specific Goal	CompositeSEP
Generic Practice	AtomicSEB
Specific Practice	AtomicSEP

Source: Author.

S1 said that **Topic** maybe is a kind of **CompositeSEP**. S2 said that he is not sure that it is correct. S1 and S4 highlight that the important is the meaning of the concept, not the concept name.

S1 said that in his view, **Topic** is closer to kind of **CompositeSEB** than **CompositeSEP**.

S2 argued that in his view, it is a multiple extension from **CompositeSEB** and **CompositeSEP**, due to it is a limitation and it says how to implement. S1 highlighted that for any model, the practice does not say how to do it. It only says what to do; how to do it is defined by yourself. S1 also completed that this difference should be clear for everyone.

At this point the doubt was if **Topic** is kind of **CompositeSEB** or **CompositeSEP**. The subjects discussed this for a while and concluded that **Topic** is a kind of **CompositeSEB**.

The final discussion result is presented in Table 12

Table 12 – Second discussion result.

SWEBoK Concept	Badge Generalization
SWEBoK	KnowledgeBase
Knowledge Area	CompositeSEB
Topic	SEB
Atomic Topic	AtomicSEB
Composite Topic	CompositeSEB

Source: Author.

3.4.2.5 Third Discussion Report

The moderator starts the second discussion by asking if Badge generalizes the Agile Manifesto.

S1 started saying that in his point of view, there is no compatibility. S4 said that it is not composed of practices necessarily.

S2 and S4 said that **Principle** does not seem a limitation. S3 said that **Principle** concept is similar to **Practice**. S4 argued with S3. At this point, S1 started arguing that it might have to push it, to say that **Principle** is **Practice**. The moderator highlighted that they should not force to extend a concept. The extension should be fluid.

S3 argued that in his point of view **Principle** is a **AtomicSEP** and that it is not pushed. S4 and S1 agreed that it is ok to say that **Principle** is a kind of **AtomicSEP** but **Value** has no relation.

S4 said that in his view, it is possible to say that **Value** is a limitation, so, a kind of **SEB**. S3 agreed that based on the definition, it makes sense.

S2 argued that it is not precisely a limitation or grouping, so it is unnatural. Based on this, all subjects agreed that make this relation is pushed. Thus, there is no concept on the ontology to represent the concept of **Value**.

The final discussion result is presented in Table 13

Table 13 – Third discussion result.

Agile Manifesto Concept	Badge Generalization
Agile Manifest	KnowledgeBase
Principle	CompositeSEP
Value	-

Source: Author.

3.4.2.6 Fourth Discussion Report

The moderator starts the second discussion by asking if Badge generalizes the Scrum framework.

S2 said that **Scrum** might be a **KnowledgeBase**, but the other concepts must be analyzed.

S1 said that Scrum says how to do and not what to do; thus, it is incompatible. S3 and S4 tried to fit the concept of **Artifact**, **Role** and **Event** in the Badge concepts, in order to find the extension relation.

S2 said that **Event** and **Artifact** are concepts difficult to find a concept in Badge S1 highlights again that it is not natural to make the relation between Badge and Scrum.

S2 and S3 discussed that Scrum said how things should be done; it is difficult to make a relation. They also state that it is pushed to make a relation between **Artifact** and **Role** with Badge concepts.

The subjects discussed trying to make a relation between Scrum concepts and Badge concepts but with no reach a point.

At this point, the Moderator intervened and asked if, based on this discussion, they think that maybe Badge does not generalize Scrum. All subjects agreed that no, Badge does not represent the knowledge in Scrum concepts. S1 said that besides the experiment goal is to find the relation, they conclude that there is no relationship because it is unnatural to relate these concepts. S4 said that when trying to read the concepts and to establish a link, it seems unnatural.

3.4.2.7 Closure Report

After the discussion steps, the moderator thanked all and provided feedback about the collected data. The subjects also asked questions about the purpose of the specific discussions and expected results. The moderator provided explanations about the subjects' doubts; then, the moderator finished the session.

3.4.3 Threats to Validity

A threat to the focus group conclusion validity is that the researchers may influence the result by searching for specific feedback. We mitigated this by carrying out a peer review process to analyze the data source and extract the session's findings. Another threat is that some outside elements may disturb the subjects during session executions. We mitigated this by performing the focus group in a reserved room and asking the subjects do not use smartphones during the session execution.

A threat to the focus group's internal validity is that the history may affect the session results because they were performed differently. We minimized this by negotiating with the subjects the best day to run the session so that no event or situation took away their concentration. Another threat is that the subjects may react negatively or positively as session time passes. We mitigated this by ensuring a maximum duration for each session and monitoring the time spent in each discussion iteration. Another threat is that the evaluation instrument may be poorly designed. We minimized this by carrying out a peer review process to verify all material used in sessions.

A threat to the focus group construct validity is that the subjects may not know how the generalization task should be performed. We mitigated this by explaining and exemplifying how the task should be done at the session background step. Another threat is that the subjects may try to guess the expected results by researchers. We minimized this by masking the focus group goal to the subjects and inserting a controlled group task. Another threat is that the researchers may bias the results based on their expectations from each focus group session. We mitigated this by carrying out a peer-review process to ensure evidence in the data source that supports the findings.

A threat to the focus group external validity is that the researchers may select the wrong subjects to participate in the sessions. We minimized this by inviting to compose the set of subjects researchers with experience in conceptual modeling, SPI and

Software Engineering. Another threat is that the researchers may provide the subjects with models specified in some notations unknown. We mitigated this by using UML as modeling language.

3.5 Chapter Lessons

In this chapter, we introduced Badge that is a domain ontology that aims to generalize SPI resources that says "what should be done". Badge was evaluated through a focus group session. It attends to the second specific goal of proposing an ontology to generalize SPI resources. Considering this, we highlight the main findings:

- the subjects agreed that Badge is able to generalize CMMI;
- the subjects agreed that Badge is able to generalize SWEBoK;
- the subjects agreed that Badge is able to generalize Agile Manifesto;
- the subjects agreed, as expected, that Badge is not able to generalize Scrum;
- the subjects concluded Badge is able to generalize models that says "what should be done".

We also highlight the main gaps found in related ontologies:

- they are created to fit specific models, such as PMBoK, CMMI, ISO, among others;
- they do not focus on generalizing models that says "what should be done".

From the evaluation results, we understand that Badge ontology was well understood by the subjects, and the generalization tasks went performed with attention. The difference between a model that says "what should be done" from a model that says "how it should be done" is an important concept to be applied when using Badge as knowledge representation.

4 PSTORY - PRACTITIONER STORY

In this chapter, we present the Practitioner Story (PStory) that is a template in which it is possible to express evidence of Software Engineering practices. It is organized as follows: Conceptual background is presented in section 4.1; Practitioner Story (PStory) template is presented in section 4.2; evaluation is presented in section 4.3; Lastly, we present chapter lessons in section 4.4.

4.1 Background

In this section, we present conceptual background about context-free grammar.

4.1.1 Context-Free Grammar

Language is a set of finite sentences constructed out of a limited set of elements (CHOMSKY, 1957). A language is created based on grammar. Thus, a grammar of a language is a device for producing the language sentences (CHOMSKY, 1957). Moreover, the grammar will generate all the grammatical sentences of a language, and none of the ungrammatical ones (CHOMSKY, 1975), which means that grammar will not create sentences that do not belong to which language.

According to Chomsky (1956), the formal grammars are classified as (also called Chomsky hierarchy):

- recursively enumerable;
- context-sensitive;
- context-free; and
- regular.

A context-free grammar is a collection of rules to structure context-free phrases. Grammar is considered context-free whether the production rules can be applied independently of the non-terminal context (CHOMSKY, 1956). Each such rule names a constituent type and specifies a possible expansion thereof (BUNDY; WALLEN, 1984).

A context-free grammar is defined by 4-tuple $G = (N, E, P, S)$, where N is the set of non-terminal symbols, E is the set of terminals, P is the finite list of production rules, and S is a non-terminal called initial symbol (RODRIGUES; LOPES, 2007). Based on context-free grammar, it is possible to determine if a sentence belongs to a context-free language or not (RODRIGUES; LOPES, 2007).

A simple example is grammar to create palindromes. Given, $G = (S, a, b, P, S)$ with productions:

$S \rightarrow aSa$
$S \rightarrow bSb$
$S \rightarrow \varepsilon$

4.2 PStory Specification

Practitioner Story (PStory) is a structured text format in which it is possible to express evidence of Software Engineering practices. PStory is freely inspired in the US[1] idea. Based on this idea, we create PStory to be the description of a software engineering practice realization from the practitioner's point of view. Thus, it describes a role, action, tool or mechanism, and result or purpose of a daily practice realization.

The text pattern is:

`As [a | an | the] <role>, I <action>, using <tool/mechanism>, in order to <result/purpose>.`

Where:

- `Role:` is the practitioner role, for example, developer, manager, tester analyst, etc.
- `Action:` is a daily action performed by a practitioner.
- `Tool/Mechanism:` is the way that this action is performed, it can be software, a physical tool, or a conceptual tool. For example, written text, conversation, meeting, etc.
- `Result/Purpose:` is the expected result, goal, or purpose of such action.

Following, we present some examples of PStories:

- As a developer, I move a card from TODO to Doing, using kanban board, in order to manage its work.
- As the Product Manager, I create tasks, using MS Project, in order to manage the project.
- As an analyst, I talk to clients and write user stories, using notepad, in order to obtain requirements.

The corresponding context-free grammar is presented in Figure 9. The terminal symbols are As a, As an, As the, I, using, in order to, ., ,, plus any words representing a role, an action, a tool or mechanism, or a result or purpose. The non-terminal symbols are ART, ME, USING, GOAL, ROLE, ACTION, TOOL, RESULT, DOT, COMMA. The initial symbol is PSTORY.

4.3 PStory Evaluation

In this subsection, we present the PStory evaluation. We performed a quasi-experiment evaluation focusing on usability and functionality. The evaluation was made by introducing the PStory by examples and exercises, then answering a questionnaire with objective questions.

[1] User Story (US) is characterized as a short and high-level description of required functionality written in customer language (BOURQUE; FAIRLEY, 2014).

Figure 9 – PStory Context-free Grammar

PSTORY → *ART ROLE ACTION USING TOOL GOAL RESULT DOT*

ART → `As the` OR `As a` OR `As an`
ME → *COMMA* `I`
USING → *COMMA* `using`
GOAL → *COMMA* `in order to` *DOT*

COMMA → `,`
DOT → `.`

ROLE → `<list of roles>`
ACTION → `<list of actions>`
TOOL → `<list of tools/mechanism>`
RESULT → `<list of results/purposes>`

4.3.1 Protocol

We created a protocol using GQM. Following, we present the Goal:

Analyze	the PStory
for the purpose of	evaluate
with respect to	functionality and usability
from the point view of	researcher
in the context of	professionals who have experience in SPI initiatives.

Following, we present the protocol Questions:

- Does the PStory can express the necessary information for diagnostic?
- Does the PStory helps to collect the relevant information for diagnostic?
- Does the PStory is plausible to be used in real life?

Next, we present the Metrics:

- likert agreement degree.

The evaluation was divided into two steps. In the first step, we executed an exercise with the subjects. The subjects were presented to software explicitly created to run this evaluation to understand and create PStories based on interview transcripts. This software has a text box with syntactical validation to facilitate the use and provide a better experience. In the second step, the subjects answer a form with profile questions plus ten questions to be answered on Likert scale about the experience to use PStory. The Table 14 shows the profile questions and answer options. The Table 15 shows the ten questions.

Table 14 – Profile questions

ID	Question
P01	How long do you act in software development?
P02	What is your most relevant academic classification?
P03	How long do you act in software process improvement initiatives?

Source: Author.

Table 15 – PStory evaluation questions

ID	Question
Q01	Each PStory template element contains diagnostic relevant information.
Q02	The PStory template contains information not relevant for carrying out the diagnosis.
Q03	Regarding the PStory template structure, I think there is information missing.
Q04	The PStory structure helps me to conduct the interview.
Q05	The PStory structure helps me to notice gaps in responses during the interview.
Q06	The PStory structure helps me identify the evidence that needs to be collected for diagnostic.
Q07	The PStory structure allows me to filter which information is useful for diagnostic during the interview.
Q08	I would use PStory to capture information for diagnosis.
Q09	I would recommend PStory to fellow consultants.
Q10	After an adaptation period it is easy to write a PStory.

Source: Author.

4.3.2 Execution and Result Analysis

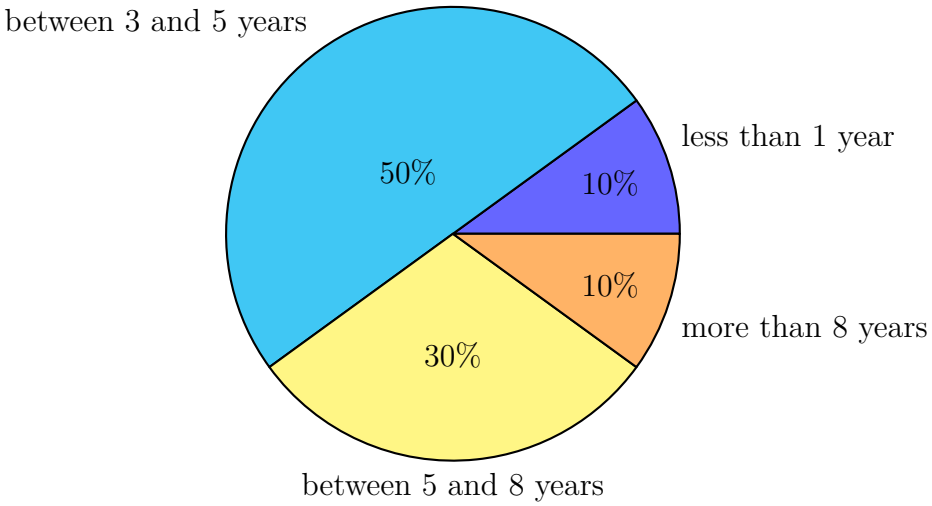
We executed the evaluation between June 15 and 30, 2021, with ten professionals with different skills and experience in SPI. As mentioned previously, we developed software to perform the evaluation. The software screens are presented in Appendix A.

As said previously, the questionnaire is divided into two parts. The first part is a profile questionnaire. The Figure 10 shows the answers to the first profile question. As can be seen, most of the subjects have between 3 and 8 years of software development. The Figure 11 shows the answers to the second profile question. As can be seen, all the subjects have graduated. Three subjects have a master's degree, and two have specialization. The Figure 12 shows the answers to the third profile question. As can be seen, the majority of the subjects have between one and five years of experience with SPI.

The second part contains ten questions to be answered on the Likert scale. The Figure 13 shows the box plot chart with all questionnaire answers.

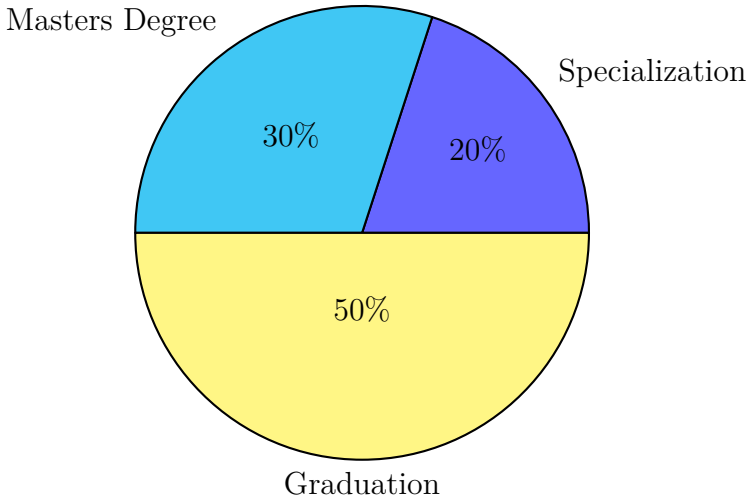
The answers analysis is performed considering the GQM questions. Considering

Figure 10 – How long do you act in software development?



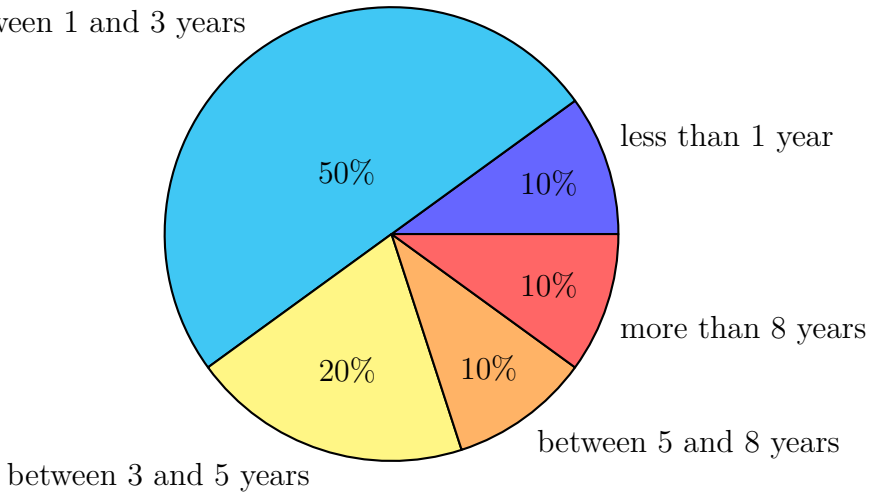
Source: Author.

Figure 11 – What is your most relevant academic classification?



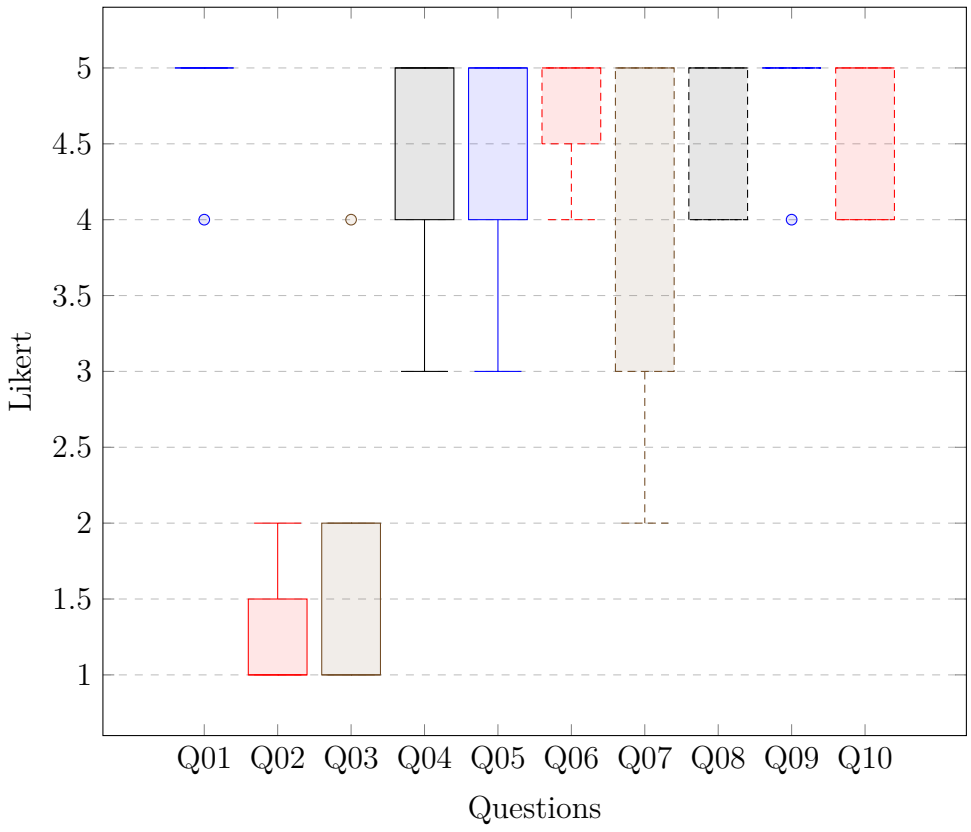
Source: Author.

Figure 12 – How long do you act in software process improvement initiatives?



Source: Author.

Figure 13 – PStory evaluation result.



Source: Author.

the first GQM question, we analyze the answers for Q01, Q02, and Q03. Based on this, the subjects strongly agree that each PStory element has importance for diagnostic. Moreover, they disagree there is no relevant information in PStory and also disagree that there is missing information.

Considering the second GQM question, we analyze the answers for Q04, Q05, Q06, and Q07. Based on this, the subjects agree that PStory structure helps to conduct the interview. They also agree that it helps to identify gaps in the interview questions and answers. Furthermore, they agree that PStory helps to identify what evidence needs to be collected, and it also helps to filter what information is helpful for diagnostic.

Considering the third GQM question, we analyze the answers for Q08, Q09, and Q10. Based on this, the subjects strongly agree that they would use PStory in real life, and they would recommend it for fellow professionals. They also agree that, after an adaptation time, PStory is easy to use.

4.4 Chapter Lessons

In this chapter, we introduced Practitioner Story (PStory) that is a template created to simplify and standardize the information gathering specification in SPI Diagnostic. PStory was evaluated through a quasi-experiment focusing on usability and functionality

It attends to the third specific goal of proposing a template to standardize the information gathering specification. Considering this, we highlight the main findings:

- we formalize PStory as a context-free language to make it easy to use and validate its correctness inside the software;
- we evaluated PStory by a quasi-experiment where professionals could exercise the PStory use and evaluate it.

5 HOW ARE ALL PROPOSES CONNECTED?

In this chapter, we present an example of how our solutions are connected to each other. It is organized as follows: The context and how to use Badge, PStory, and Coptic are presented in section 5.1; Lastly, we present chapter lessons in section 5.2.

5.1 Context

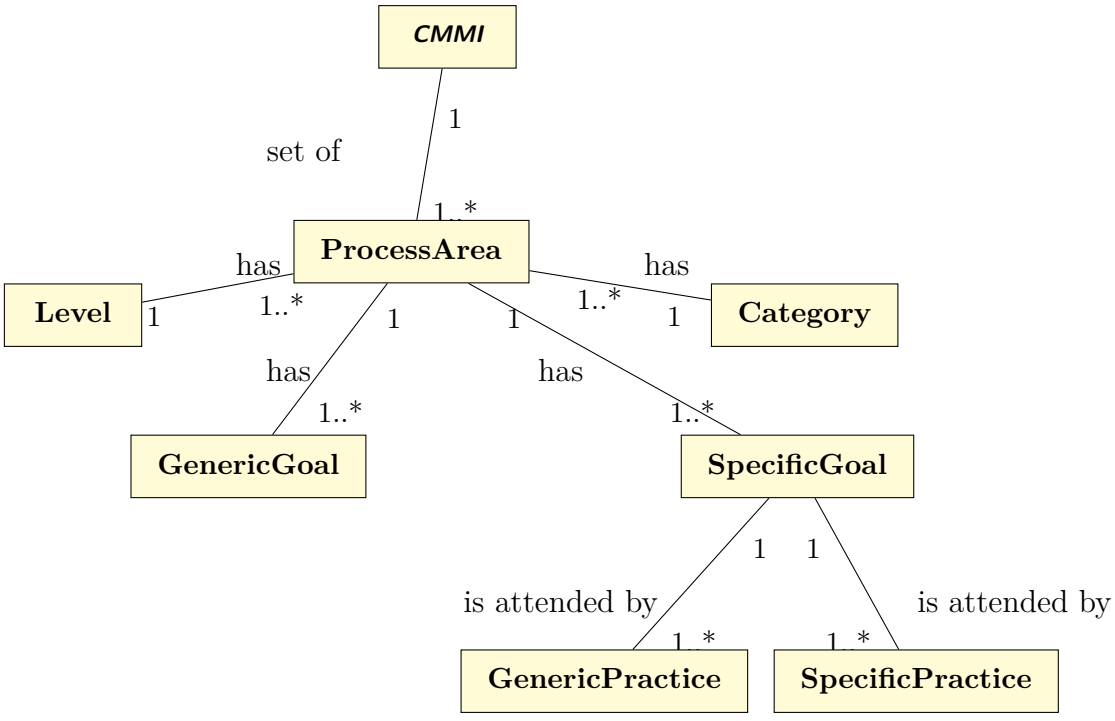
Getting back to the problem situation mentioned previously (see section 1.1), the SEPG starts by creating the reference model on Coptic based on Badge, in this example, we will consider that this improvement initiative is to CMMI ML 2 upgrade.

5.1.1 Creating Model Based on Badge

Based on this, let's consider that the reference model is already created in Coptic and the corpora are populated. The Appendix C shows the example of a minimum corpus for CMMI ML 2 corpora based on Rational Unified Process (RUP)

The reference model is created on Coptic. The Figure 14 shows the CMMI instance on Badge ontology.

Figure 14 – CMMI instance



Source: Author.

5.1.2 Writing PStories

The rest of the diagnostic process is still the same, the SEPG member runs interviews on the target organization. Using a web client for Coptic, the consultant takes notes by writing the answers in PStory format. The Table 16 shows examples of PStories written during the interview.

Table 16 – PStories written during the interview.

PStory
As a Project Manager, I evaluate the risks, using Risk Matrix, in order to delimit the scope of the project.
As a Project Manager, I review the requirements, using Jira, in order to maintain the fidelity of the understanding of the functionalities.
As the Project Manager, I plan phases and iterations, using Jira, in order to establish project estimates.
As a PO, I manage requirements changes, using Jira, in order to establish the customer journeys correctly.
As the Project Manager, I plan the phases and iterations, using Jira, in order to establish project estimates.
As the Project Manager, I develop project planning, using Confluence, in order to establish a project plan.
As a Project Manager, I identify and evaluate the risks, using confluence, in order to delimit the scope of the project.

Source: Author.

5.1.3 Getting Coptic Analysis Result

In the previous example, after the interview, the SEPG obtain a massive quantity of information, and this information is distributed to all SEPG members to perform an individual analysis and a later meeting to discuss the findings. Using our solution, while the PStories are written during the interview, Coptic is able to perform a preliminary analysis based on corpora and returns to the client software a similarity level of attendance of the PStory and CMMI practices. The Table 17 shows an example of an answer that Coptic is able to provide during an interview.

In this example, Coptic is retrieving the best-matched practice based on the semantic similarity from each PStory. Moreover, Coptic is able to perform different answers, for example, all matched practices from each PStory, all PStories that match a single practice, among others.

After ending the interview series, all SEPG members have access to the collected information and the preliminary analysis performed by Coptic. Based on this analysis, the group can have a small discussion about any divergence in the Coptic analysis and the consultant's interpretation. In this scenario, the analysis is less subjective and less

Table 17 – PStories analyzed by Coptic

PStory	Practice	Similarity
As a Project Manager, I apply an evaluative checklist, using Confluence, in order to assess the parameters of the process.	PPQA SP 1.1	49.37%
As a Project Manager, I evaluate the risks, using Risk Matrix, in order to delimit the scope of the project.	PP SP 1.1	8.51%
As the Project Manager, I conduct project reviews, using Notion, in order to establish records.	PP SP 2.6	5.53%
As a PO, I manage requirements changes, using Jira, in order to establish the customer journeys correctly.	REQM SP 1.3	48.95%
As the Project Manager, I plan the phases and iterations, using Jira, in order to establish project estimates.	MA SP 1.4	20.98%
As the Project Manager, I develop project planning, using Confluence, in order to establish project plan.	PMC SP 2.1	20.42%
As the Team Leader, I create reports with the team deliverability from the last five sprints, using Jira, in order to assess the effectiveness of the current process.	PP SP 3.2	6.11%

Source: Author.

dependent on exclusive knowledge and consultant experience. Previous problems, such as a huge amount of data, incomplete collected information, and subjective analysis are minimized by the semi-automated process performed by Coptic.

5.2 Chapter Lessons

In this chapter, we presented how our solutions are connected. We are proposing three solutions whose goal is to perform an intelligent classification of SPI practices and evidence, thus, each solution has its purpose and goal.

Considering this, we highlight that:

- Badge is an ontology able to create several knowledge bases of software engineering;
- PStory is a template used to write evidence of software engineering practices attendance;
- Coptic is an intelligent tool able to perform intelligent analysis and classification of practices and evidence written using PStory.

6 COPTIC - COLLECTOR OF SOFTWARE PROCESS IMPROVEMENT EVIDENCES

In this chapter, we introduce a software tool called Coptic. Coptic is an intelligent Practice-Evidence classification tool based on Natural Language Processing and Semantic Similarity. It has been developed based on the incremental and iterative life cycle. The chapter is organized as follows: conceptual background is presented in section 6.1; related tools are presented in section 6.2; Coptic is presented in section 6.3; evaluation is presented in section 6.4; Lastly, we present chapter lessons in section 6.5.

6.1 Background

In this section, we present conceptual background about intelligent software engineering and the synergy between software engineering and AI. Moreover, we present conceptual background about Natural Language Processing (NLP) and Machine Learning (ML).

6.1.1 Natural Language Processing

Natural Language Processing (NLP) is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do valuable things (CHOWDHURY, 2003). Based on this research area, it is possible to extract concepts from natural language text and manipulate them. It is also possible to identify semantics, syntax, grammatical class, among others.

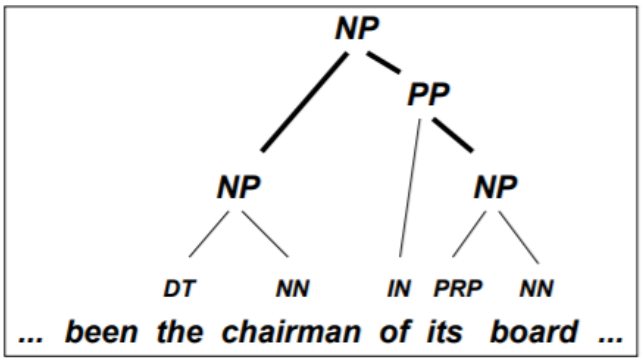
NLP uses Syntactic Parse Tree (SPT) to extract the syntactic elements inside a sentence (KAMBHATLA, 2004). SPT is a structure that represents a context-free language. The Figure 15 shows the SPT of a natural language sentence, where the elements in the tree represent:

- DT - Determiner;
- NP - Noun Phrase;
- PP - Prepositional Phrase;
- NN - Noun, singular or mass;
- IN - Preposition or subordinating conjunction;
- PRP - Personal pronoun (KAMBHATLA, 2004).

6.1.2 Semantic Similarity

Although synonyms to be known as a friendly semantic relation between words, defining synonymy is more complicated than it might at first seem (MILLER; CHARLES, 1991). According to Cambridge (2020), a synonym is a word or phrase that has the same or nearly the same meaning as another word or phrase in the same language. While, according to Merriam-Webster (2020), a synonym is defined as one of two or more words

Figure 15 – Syntactic Parse Tree.



Source: Kambhatla (2004)

or expressions of the same language that have the same or nearly the same meaning in some or all senses.

The problem is precisely to define what is "same meaning". Following a formulation usually attributed to Leibniz, two words are said to be synonyms if one can be used in a statement in place of the other without changing the meaning of the statement (MILLER; CHARLES, 1991). Considering this, one can define that two or more words or phrases have the same meaning by finding the correspondent Semantic Similarity.

Semantic Similarity may be obtained by combining a lexical taxonomy structure with corpus statistical information (JIANG; CONRATH, 1997). In other words, semantic similarity aims to calculate the relatedness between a pair of words phrases in terms of semantic meaning.

The method proposed by Jiang e Conrath (1997) uses a corpus-based method in conjunction with lexical taxonomies to calculate semantic similarity between words or concepts. In the method proposed by Kolb (2009), the corpus is tokenized, and commonplace function words are eliminated, then they use a context window for counting co-occurrences.

DISCO[1] is an open-source Java library dedicated to the semantic similarity computation between words (KOLB, 2008; KOLB, 2009). The tool is distributed under the Apache License, version 2.0. Several measures are implemented. Interestingly, numerous languages are also supported: Arabic, Czech, Dutch, English, French, German, Italian, Russian, and Spanish. Unfortunately, the last version available to date is version 3.0 (released in 2018).

6.2 Related Tools

We propose to develop an intelligent tool to support SPI Diagnostic. Thus, we highlight as related tools the subset of studies from chapter 2 that proposes an intelligent

[1] It can be found at http://www.linguatools.de/disco

software for SPI Diagnostic. The Table 18 shows the list of intelligent software tools.

Table 18 – Related Tools

ID	Author	
P01	Minella, Thiry e Fernandes (2015)	Diagnóstico de processos em organizações intensivas em software usando um sistema especialista
P02	Moreira et al. (2013)	Autodiagnóstico de Processo de Software Baseado em Sistema Especialista
P03	Rodríguez et al. (2018)	KAIRÓS: Intelligent System for Scenarios Recommendation at the Beginning of Software Process Improvement

Source: Author.

Minella, Thiry e Fernandes (2015) present a software based on an expert system to support the organization diagnostic. The system captures the development life cycle, mapping how the organization works. Moreira et al. (2013) presents a web-based software tool built on an expert system for self-diagnosis. Rodríguez et al. (2018) present KAIRÓS, intelligent software for scenarios recommendation at the beginning of SPI. KAIRÓS automates the processing of Critical Success Factors and Good Practices combined.

Our proposed tool focus on the information specification of the gathering step on SPI Diagnostic. Moreover, it also provides intelligent support in the analysis step, helping to match evidence and practices. This intelligent support is based on NLP and semantic similarity. It differs from previously presented works, once, Minella, Thiry e Fernandes (2015) propose a tool to support the diagnostic process handling. It is also different from Moreira et al. (2013) that propose a tool for self-diagnosis. Both proposals are based on expert systems.

6.3 Coptic Development

Coptic is intended to helps SEPGs to match Software Engineering Practices and evidence during the diagnostic phase in SPI initiatives. In Coptic, the user can write PStories and receives; as a result, the percentage of similarity between the provided PStory and PStories from corpora for each reference model practice. Initially, we defined Coptic scope, architecture, and technologies. Coptic has a complete number of features to support practice and evidence matching, as can be seen in Table 19.

The software architecture is composed of RESTful web services. As shown in Figure 16, the software has two web services that communicate with each other. The core service is responsible for process data and communicates with storage and semantic calculator service. The semantic calculator service receives the PStory to be measured

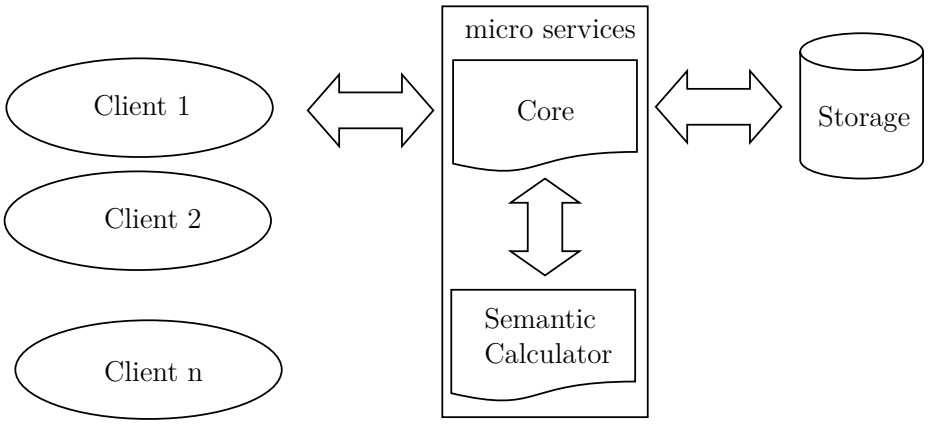
Table 19 – Coptic user stories.

Code	User Story
US01	As a SEPG member, I want to maintain Software Engineering Knowledge Base.
US02	As a SEPG member, I want to maintain a Reference PStory corpora.
US03	As a SEPG member, I want to estimate the similarity among PStory and Software Engineering Practice.

Source: Author.

and the source corpora, thus it responds with the semantic similarity percentage. Coptic may be used with any client through HTTP requests.

Figure 16 – Software Architecture



Source: Author.

Coptic is a web-based tool developed in Java, using Spring Boot REST. The storage layer is based on the PostgreSQL database. The PStory grammar and parser have been developed by using Xtext framework (BEHRENS et al., 2008), which is an Eclipse-based framework for Domain Specific Language (DSL) building. The Figure 17 shows the Xtext version on the PStory grammar. The Semantic Similarity service is developed using DISCO.

The Semantic similarity is performed based on a reference PStory corpora. The user maintains this corpus, and it can be improved as new reference PStories are discovered. The preliminary PStory corpora is based on RUP(KROLL; KRUCHTEN, 2003). We used RUP as a reference once it is one of the most popular and complete processes for software development. We created the corpora based on the activities proposed by Monteiro et al. (2013). However, as mentioned previously, each time that a new PStory is discovered and the user decides that it is a good reference, it can be added in the PStory corpora in order to make it more comprehensive.

Following, we present class diagrams for each US presented previously. Some

Figure 17 – PStory Xtext Grammar

```
grammar com.ecarsm.mk.PStory
    with org.eclipse.xtext.common.Terminals

generate pStory "http://www.ecarsm.com/mk/PStory"

Model:
    (stories += (PStory))*
;

PStory:
        ('As a' | 'As an' | 'As the') role=Free
        (', I') action=Free
        (', using') tool=Free
        (', in order to') result=Free
        ('.')
;

Free hidden(WS):
    (ID|INT)+
;
```

Source: Author.

methods and attributes were omitted due to better visualization purposes. Considering the first US, the Figure 18 shows the class diagram with the entities that compose the Badge ontology to create knowledge bases. Considering the second US, the Figure 19 shows the main entities responsible to maintain a PStory reference corpora. Considering the third US, the Figure 20 shows the main classes to calculate and estimate the similarity among PStory and Practices.

As mentioned previously, Coptic is essentially web-service software, which means that there is no specific client to access it, any application which can make HTTP requests can access Coptic.

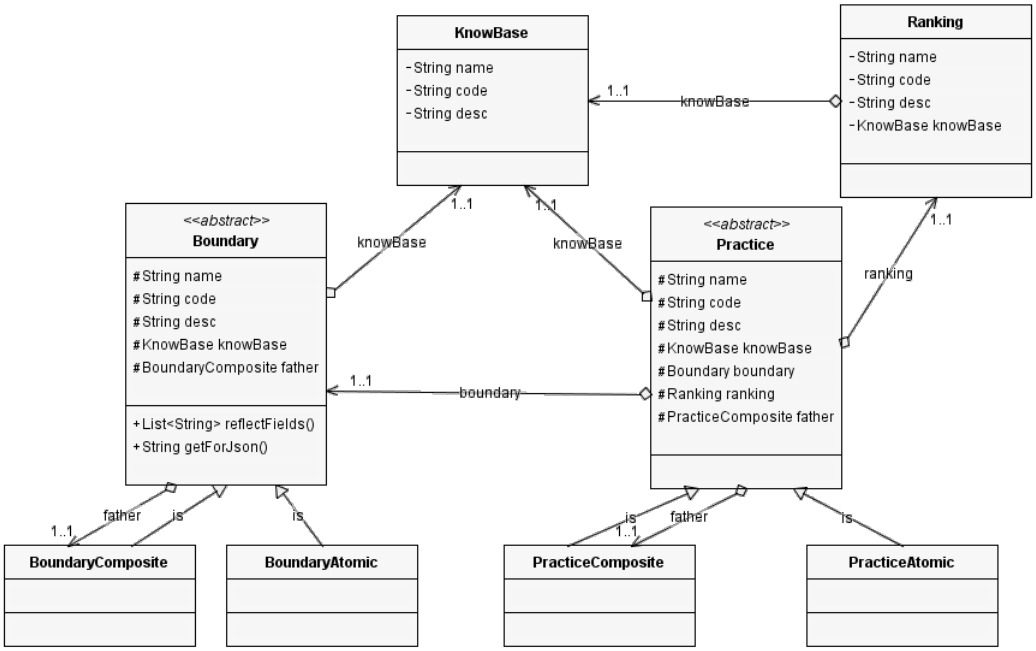
We developed a specific request to implement the requirements mentioned in Table 19. The Table 20 shows the requests for US01. The Table 21 shows the requests for US02. The Table 22 shows the requests for US03.

6.4 Coptic Evaluation

In this subsection, we present the Coptic evaluation. We performed the evaluation, using CMMI 1.3 Maturity Level 2 practices without Supplier Agreement Management (SAM).

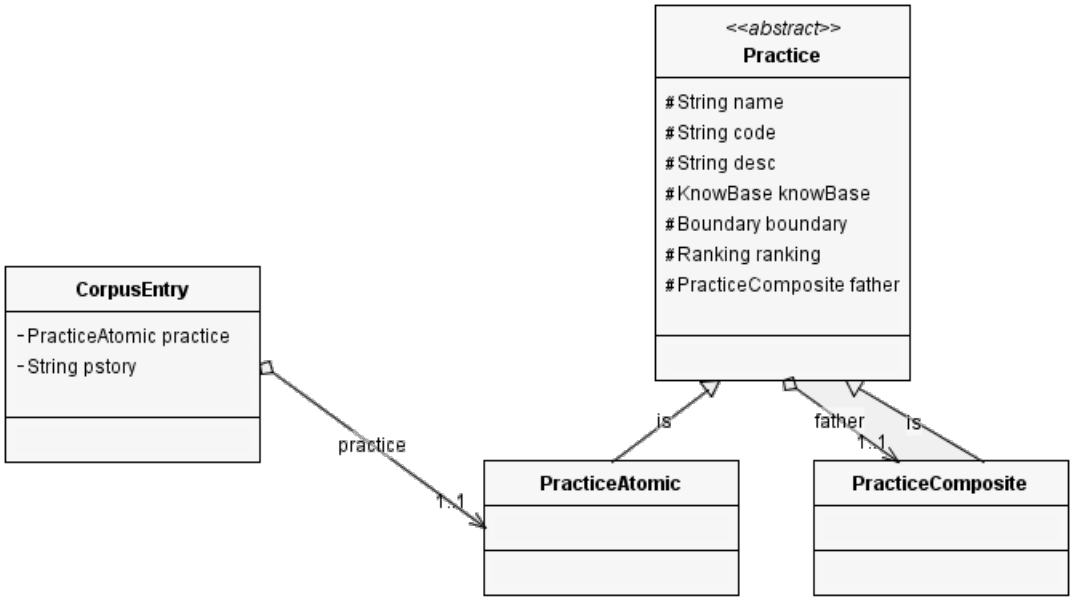
The evaluation was performed in two steps. The first step was the practice-evidence

Figure 18 – First US class Diagram.



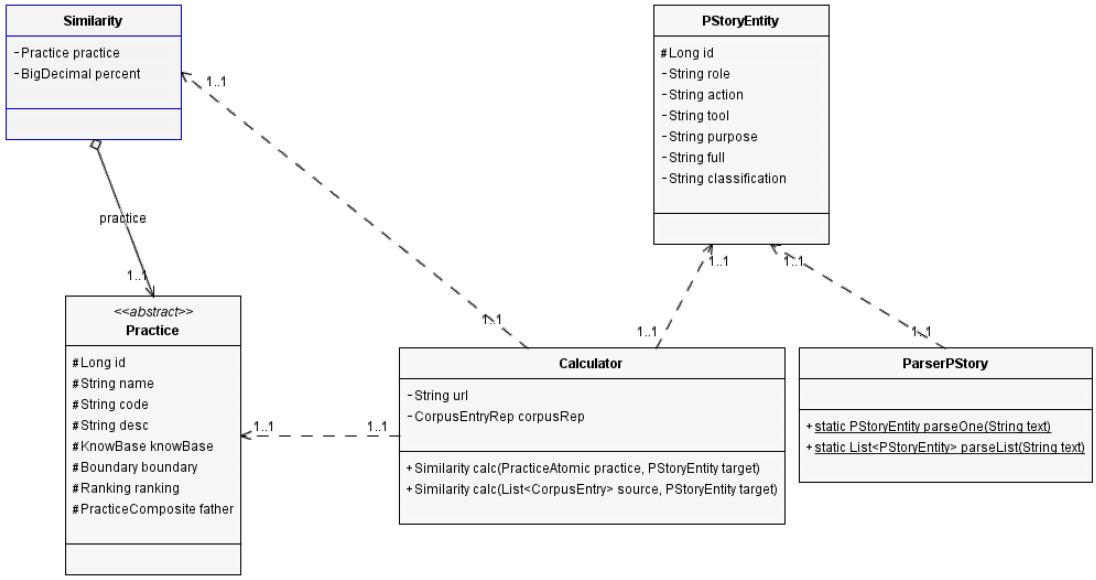
Source: Author.

Figure 19 – Second US class Diagram.



Source: Author.

Figure 20 – Third US class Diagram.



Source: Author.

Table 20 – US01 requests (see Table 19).

HTTP	Path	Description
POST	/commorus/knowbase	Create new knowledge base
GET	/commorus/knowbase/{knowbase-id}	Get knowledge base with all children in tree format
PUT	/commorus/knowbase/{knowbase-id}	Update any knowledge base element
DELETE	/commorus/knowbase/{knowbase-id}	Remove knowledge base by id

Source: Author.

Table 21 – US02 requests (see Table 19).

HTTP	Path	Description
POST	/corpus	Create new corpora entry
GET	/corpus/tree/{knowbase-id}	Get the knowledge base tree with each corpora entry added
PUT	/corpus/{entry-id}/tuning	corpora entry fine tuning

Source: Author.

Table 22 – US03 requests (see Table 19).

HTTP	Path	Description
POST	/diagnostic	Returns the practice-evidence matching rate based on knowledge base and ranking sent via body request
POST	/diagnostic/by/practice	Returns the practice-evidence matching rate based on specific practice code sent via body request

Source: Author.

matching performed by professionals. The second step was the practice-evidence matching performed by the Coptic then result in comparison and analysis.

Based on that, for each of the 48 practices, we discovered one positive and one negative PStory. The human practice-evidence matching was performed by two professionals with more than 15 years of experience who have participated in SEPGs in CMMI and MPS.BR initiatives.

6.4.1 Evaluation Protocol and Execution

In order to perform the Coptic evaluation, we created a protocol using GQM. Following, we present the Goal:

Analyze	the use of Coptic Tool
for the purpose of	evaluate
with respect to	effectiveness
from the point view of	researcher
in the context of	professionals who have experience with CMMI.

Following, we present the protocol Questions:

- Does the Coptic result match reality?
- Does the Coptic performs the analysis faster than manual analysis.

Next, we present the Metrics:

- the practice-evidence matching performed by professionals.
- the practice-evidence matching performed by Coptic.
- Comparison quotient of practice-evidence between PStory and practice made by the professionals and Coptic.

We performed the evaluation between July 10, 2021, and August 15, 2021. In the first step, we created a form with all practices mentioned previously and three answer options, two PStories and a “None of them.” option. The professionals should select the PStories that attend the practice fully or partially or select “None of them.” in case no PStories attend the practice.

In case of divergent answers between the professionals, we used the Delphi tech-

nique (TUROFF; LINSTONE, 2002) to find a consensus. After four rounds of Delphi, we ended with two lists, a list of negative PStories for each practice and other lists with positive PStories, both agreed by two professionals.

In the second step, we used Coptic to find the semantic similarity among each PStory and the practices.

6.4.2 Result analysis and Discussion

As mentioned previously, we have found two lists. The first one is a list of positive PStories, which means PStories that attend each practice of CMMI ML 2 without SAM. The second list is a list of negative PStories, which means PStories that do not attend the practices.

This attendance is stated by the agreement of two professionals. The Appendix B shows the positive and negative lists of the Coptic evaluation result with each PStory.

Considering this evaluation, we assumed that Coptic positive results mean that the PStory attend the practice.

Based on this, for the positive list, Coptic presented 93.75% of coherence with professional analysis.

The Figure 21 shows the result of CM area considering the positive PStories. The chart shows that SP1.1, SP2.1, and SP3.1 present positive but low similarity, while SP1.2, SP1.3, SP2.2, and SP3.2 present positive and medium similarities.

The Figure 22 shows the result of MA area considering the positive PStories. By the chart, we can analyze that SP1.1, SP1.2, and SP1.4 present positive and high similarity, SP1.3, SP2.2, and SP2.3 present positive and medium similarity, while SP2.1 and SP2.4 present positive but slightly low similarity.

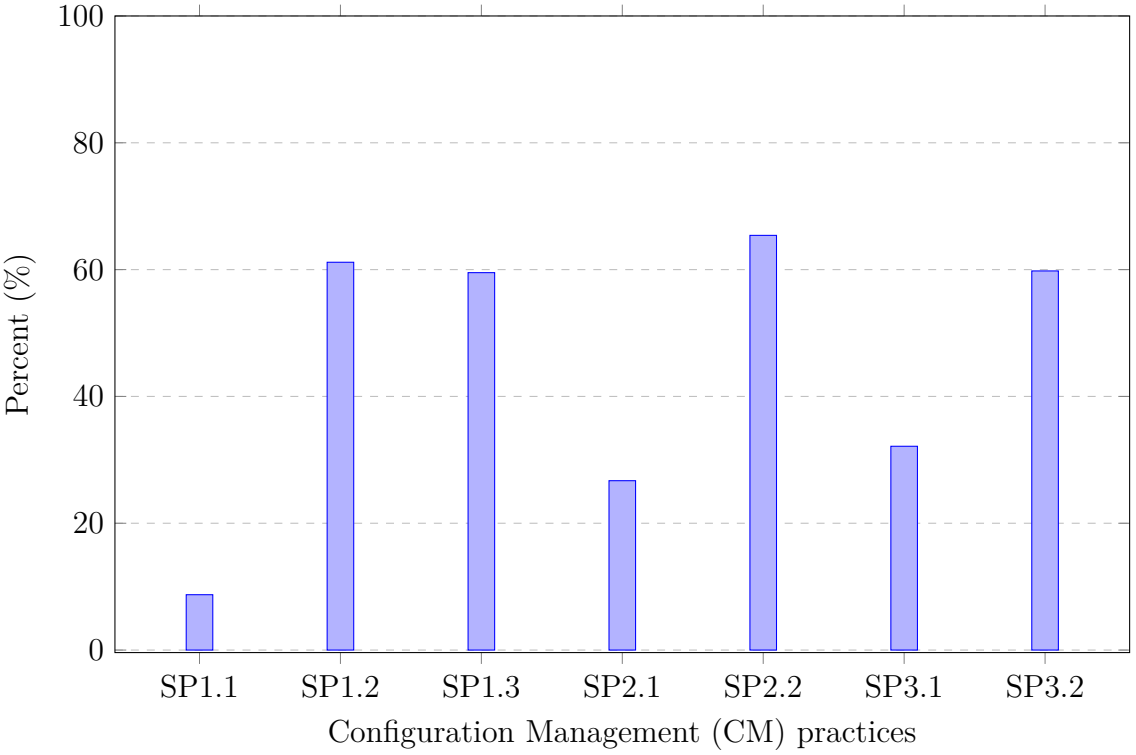
The Figure 23 shows the result of PPQA area considering the positive PStories. The chart shows that SP1.1 and SP2.1 present positive and medium similarity, SP2.2 presents positive and low similarity, while SP1.2 presents negative and slightly low similarity.

The Figure 24 shows the result of PMC area considering the positive PStories. By the chart, we can analyze that the SP1.2 presents positive and high similarity, SP1.2, SP1.5, SP1.6, and SP1.7 present positive and medium similarity, while SP1.3, SP1.4, SP2.2, and SP2.3 present positive and low similarity, and SP2.1 shows negative and medium similarity.

The Figure 25 shows the result of PP area considering the positive PStories. By the chart, we can analyze that the SP2.3 presents positive and high similarities, SP1.1, SP2.2, SP2.4, SP2.5, SP2.7, SP3.2, and SP3.3 present positive and medium similarity, while SP1.3, SP1.4, SP2.1, SP2.6, and SP3.1 present positive and low similarity, and SP1.2 presents negative and low similarity.

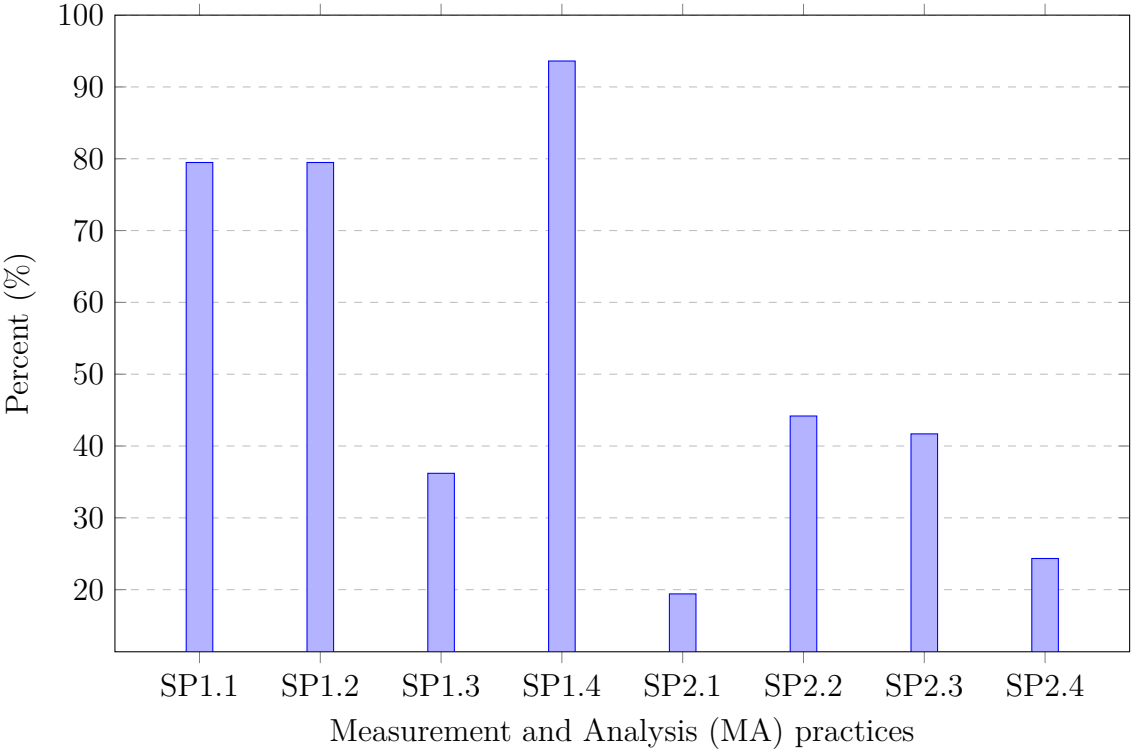
The Figure 26 shows the result of REQM area considering the positive PStories.

Figure 21 – Configuration Management (CM) positive PStories



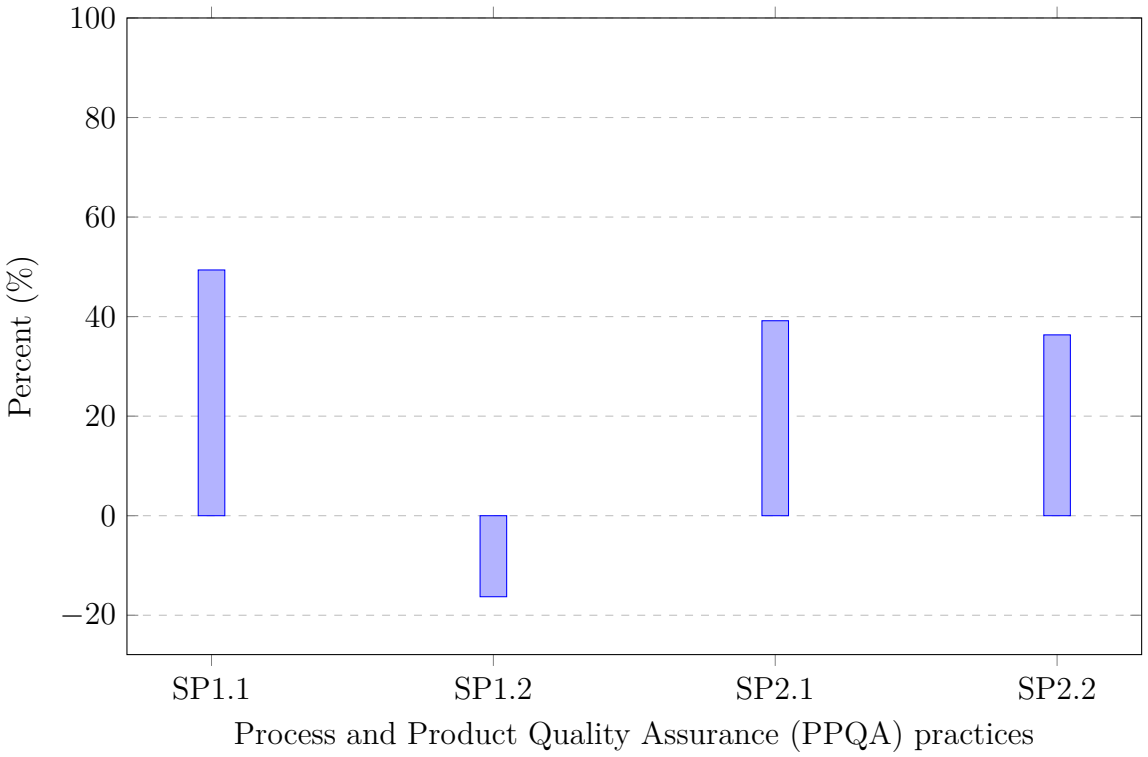
Source: Author.

Figure 22 – Measurement and Analysis (MA) positive PStories



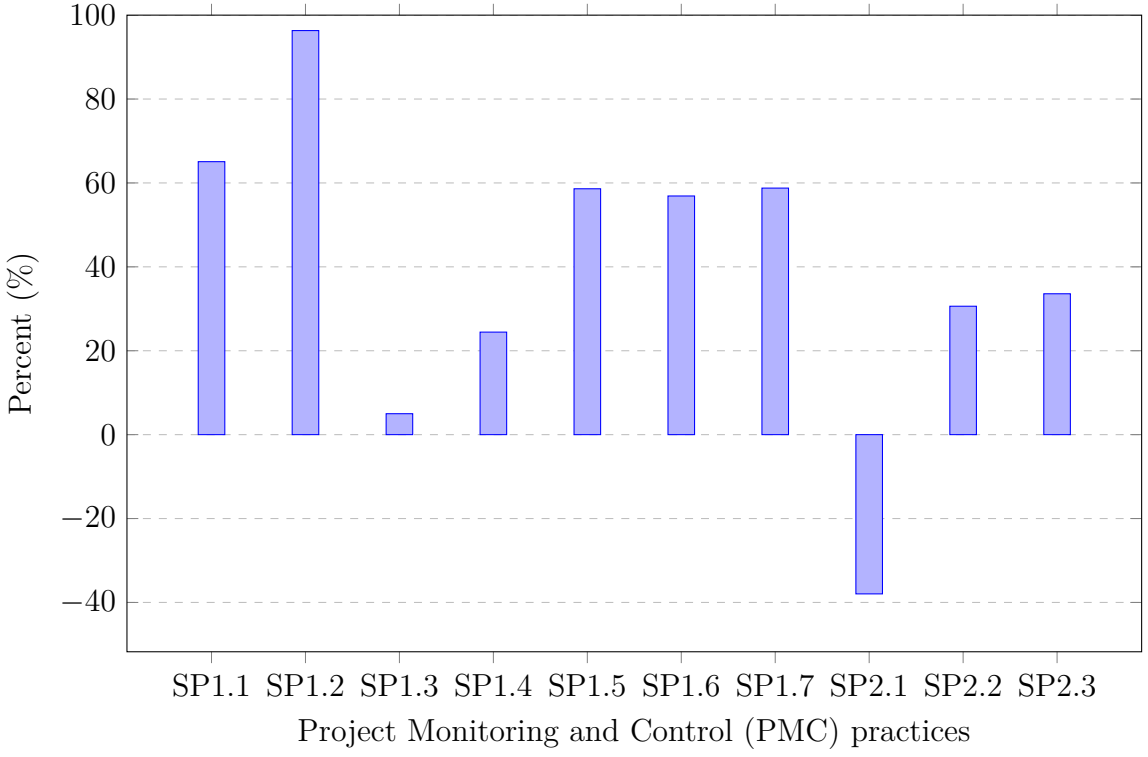
Source: Author.

Figure 23 – Process and Product Quality Assurance (PPQA) positive PStories



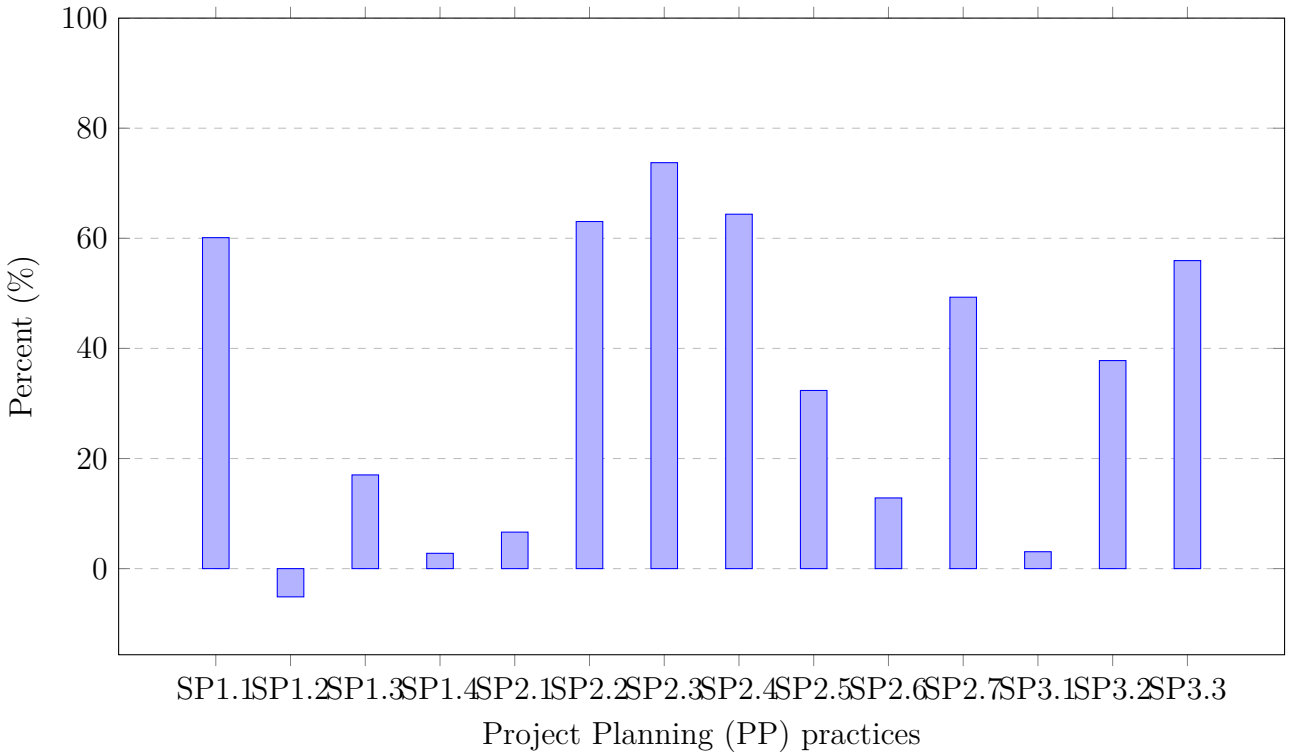
Source: Author.

Figure 24 – Project Monitoring and Control (PMC) positive PStories



Source: Author.

Figure 25 – Project Planning (PP) positive PStories



Source: Author.

The chart shows that SP1.1, SP1.3, and SP1.4 present positive and medium similarity, while SP1.2 and SP1.5 present positive and low similarity.

Considering the negative list, Coptic presented 43.75% of coherence with professional analysis.

The Figure 27 shows the result of CM area considering the negative PStories. The chart shows that SP1.1, SP2.1, SP2.2, and SP3.1 present low negative similarity, while SP1.2, SP1.3, SP2.2, and SP3.2 present positive low similarity.

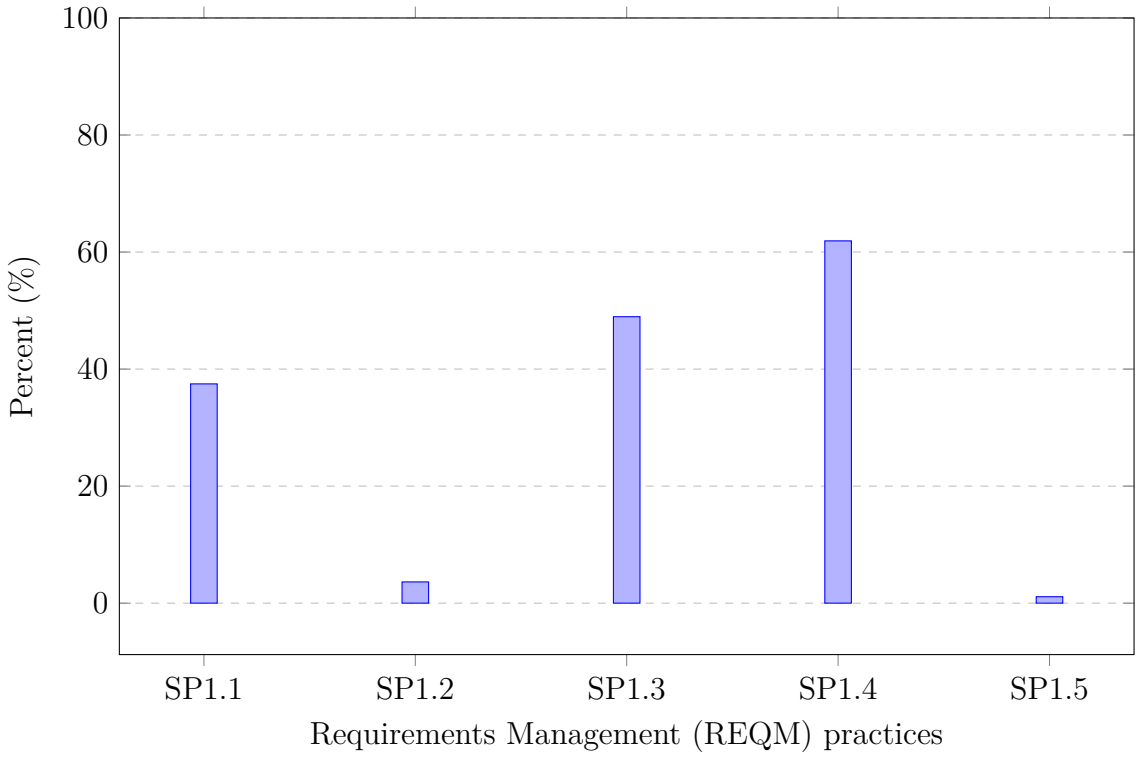
The Figure 28 shows the result of MA area considering the negative PStories. The chart shows that SP1.1, SP1.2, and SP2.3 present negative low similarity, SP1.4, SP2.1, SP2.2, and SP2.4 present positive and low similarity, while SP1.3 shows positive medium similarity.

The Figure 29 shows the result of PPQA area considering the negative PStories. The chart shows that SP1.1 presents low negative similarity, SP1.2, SP2.2 present positive medium similarity, while SP2.1 presents positive medium similarity.

The Figure 30 shows the result of PMC area considering the negative PStories. By the chart, we can analyze that SP1.3, SP1.6, and SP2.1 present negative low similarity, SP1.1, SP1.2, SP1.5, SP2.2, and SP2.3 present low positive similarity, while SP1.4 and SP1.7 present positive medium similarity.

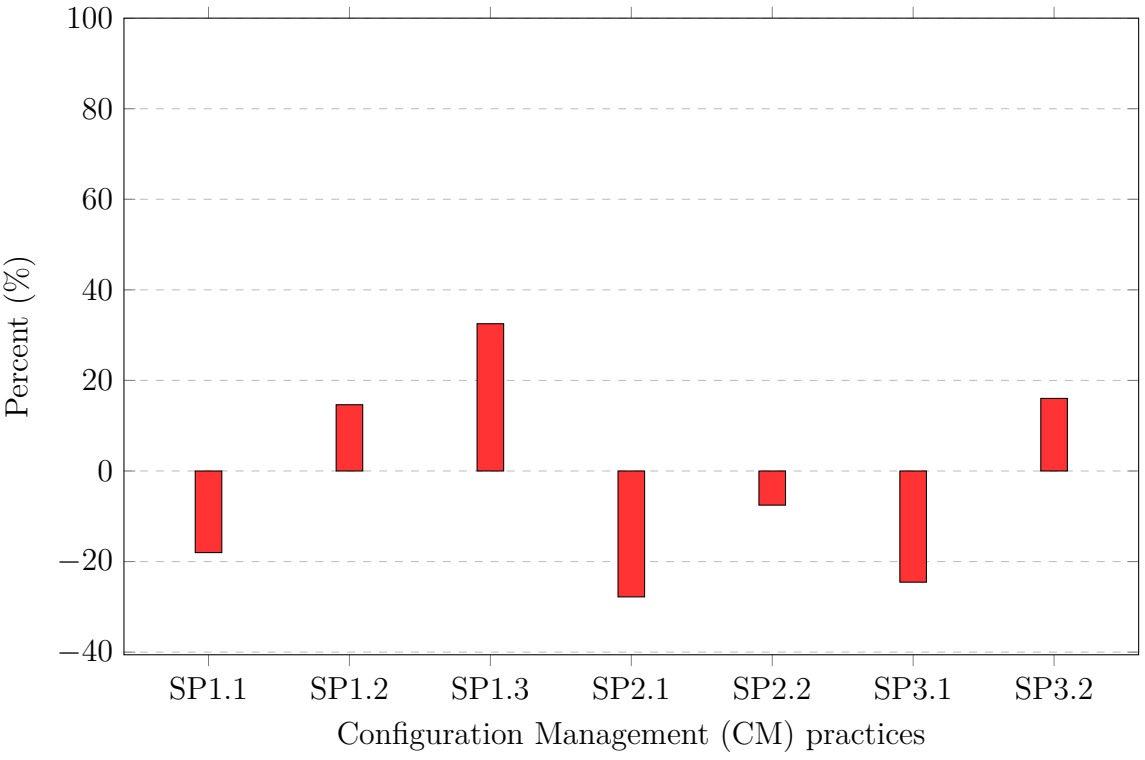
The Figure 31 shows the result of PP area considering the negative PStories. The

Figure 26 – Requirements Management (REQM) positive PStories



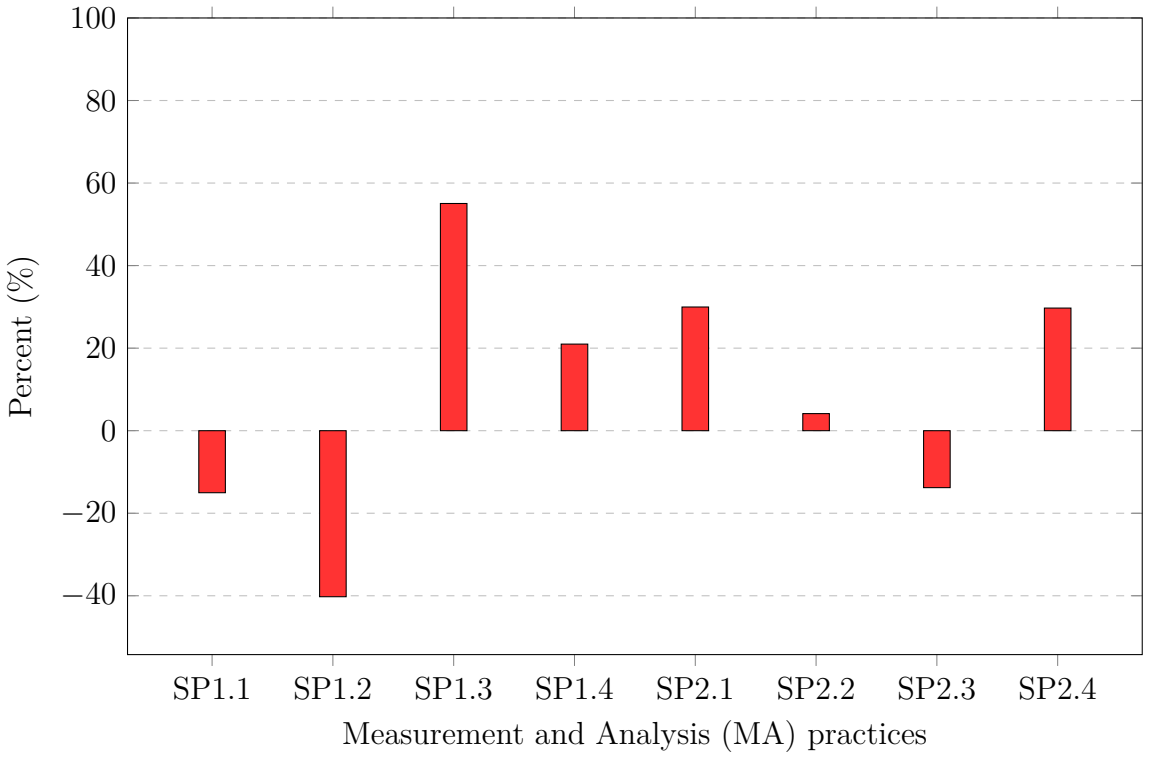
Source: Author.

Figure 27 – Configuration Management (CM) negative PStories



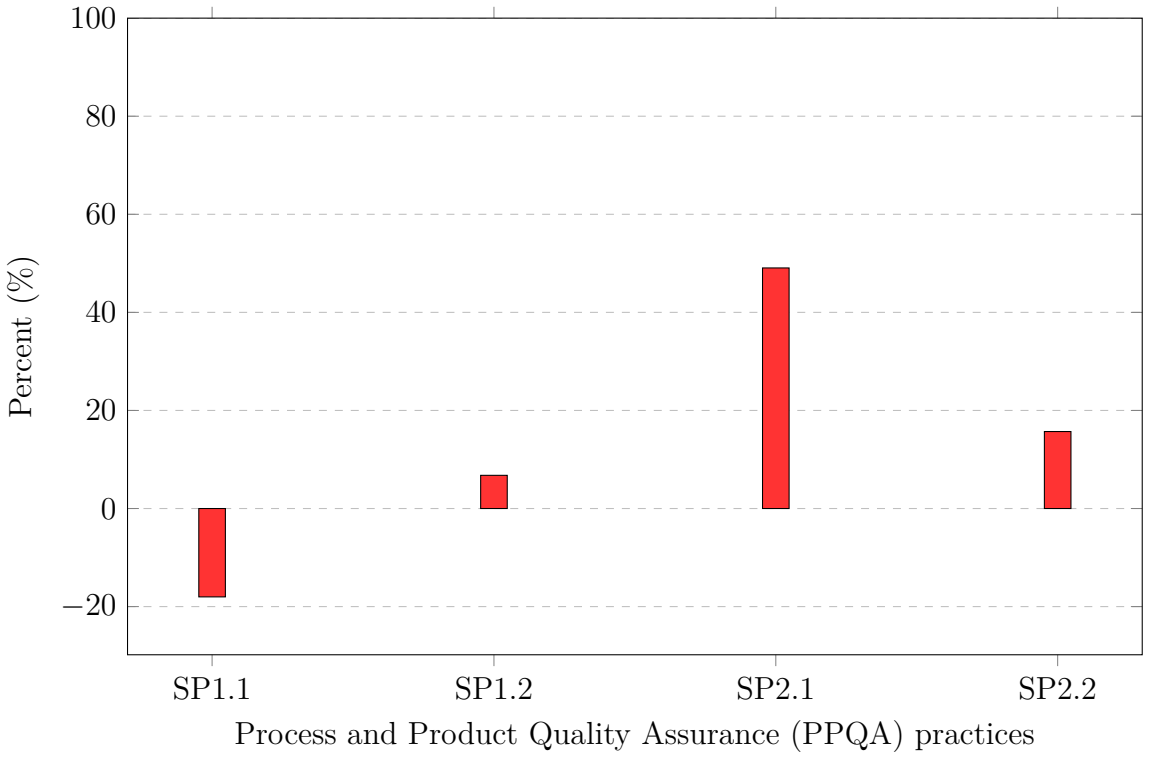
Source: Author.

Figure 28 – Measurement and Analysis (MA) negative PStories



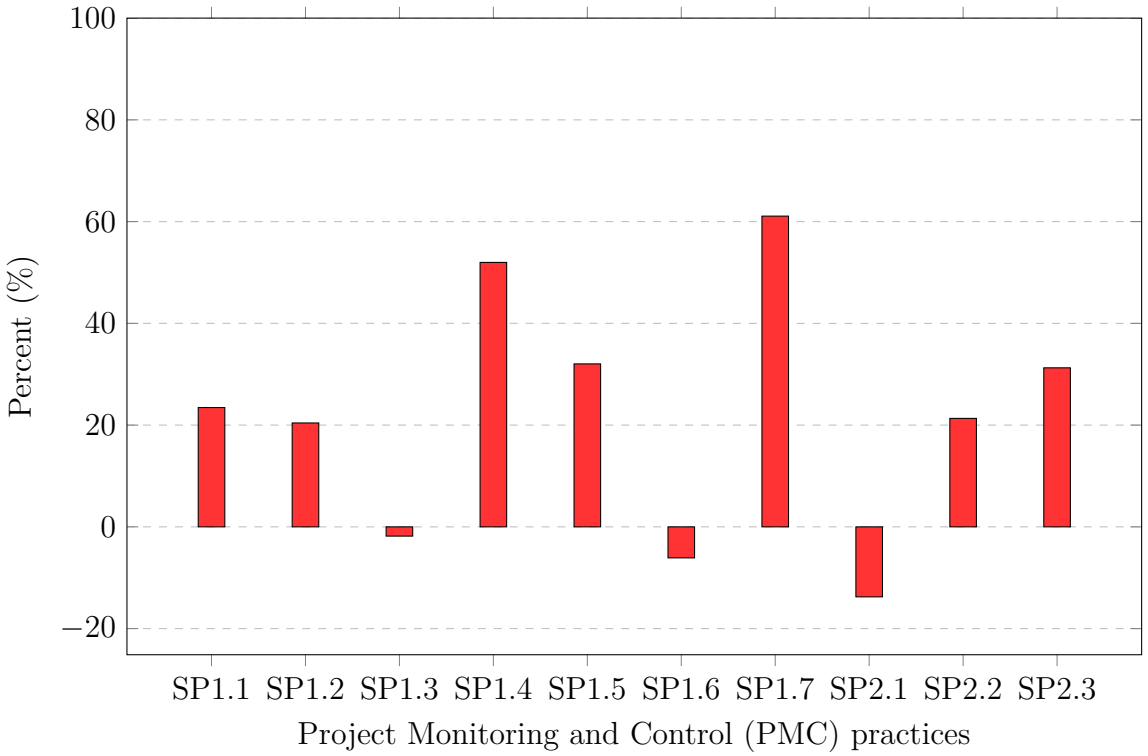
Source: Author.

Figure 29 – Process and Product Quality Assurance (PPQA) negative PStories



Source: Author.

Figure 30 – Project Monitoring and Control (PMC) negative PStories



Source: Author.

chart shows that SP1.4, SP2.1, SP2.2, SP2.4, SP2.5, and SP3.1 present low negative similarity, SP1.1, SP1.2, SP1.3 SP2.3, SP2.6, and SP3.2 present low positive similarity, while SP2.7 shows positive medium similarity.

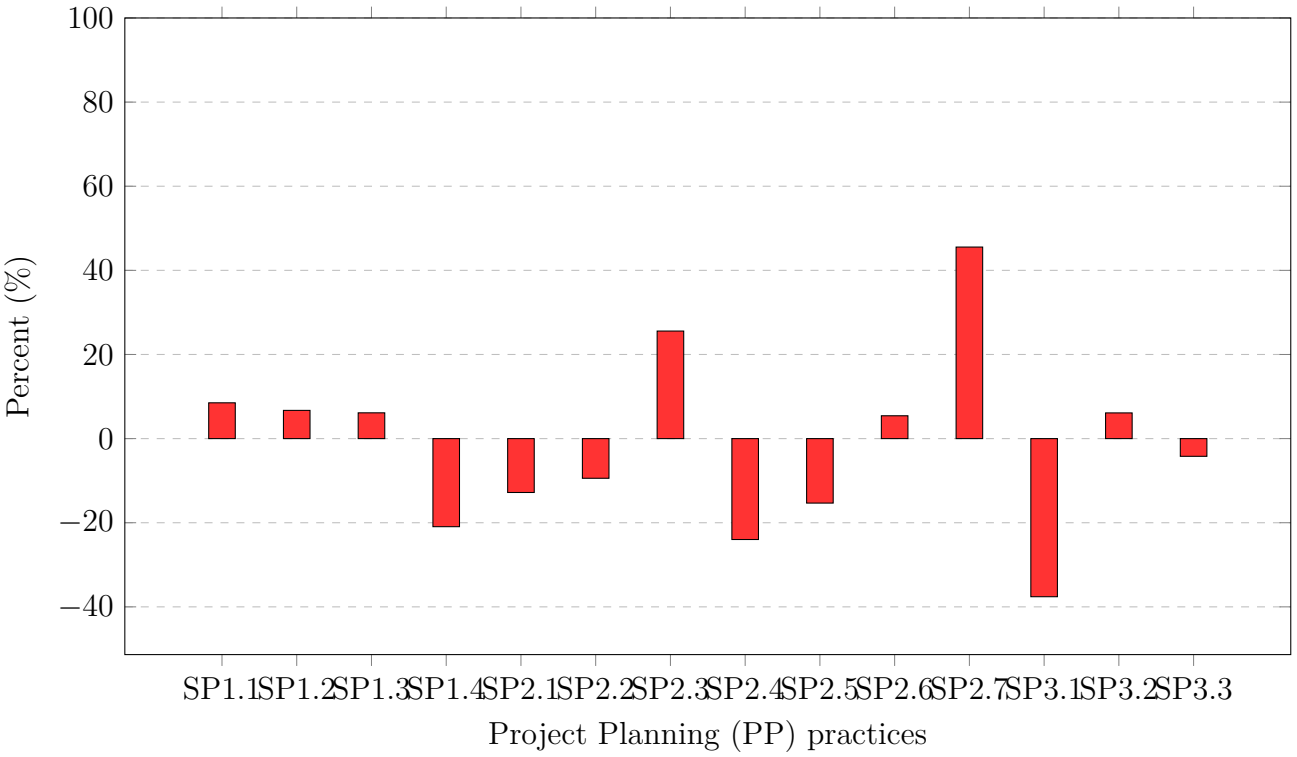
The Figure 32 shows the result of REQM area considering the negative PStories. The chart shows that SP1.2, SP1.3, and SP1.4 present negative low similarity, SP1.1 presents low positive similarity, while SP1.5 presents positive medium similarity.

Getting back to our GQM questions. For the first question, we consider that Coptic results are coherent with the reality, fine-tuning of corpora entries are needed, but the first result is promising. For the second question, we consider that Coptic results help professionals to make diagnostic analysis with better performance, once part of the analysis is done by Coptic.

Considering the overall result, we analyze it as positive. The Coptic tool can identify similar PStories that attend the software engineering practices based on corpora. Even so, the result is not optimal, following we present some details about the results.

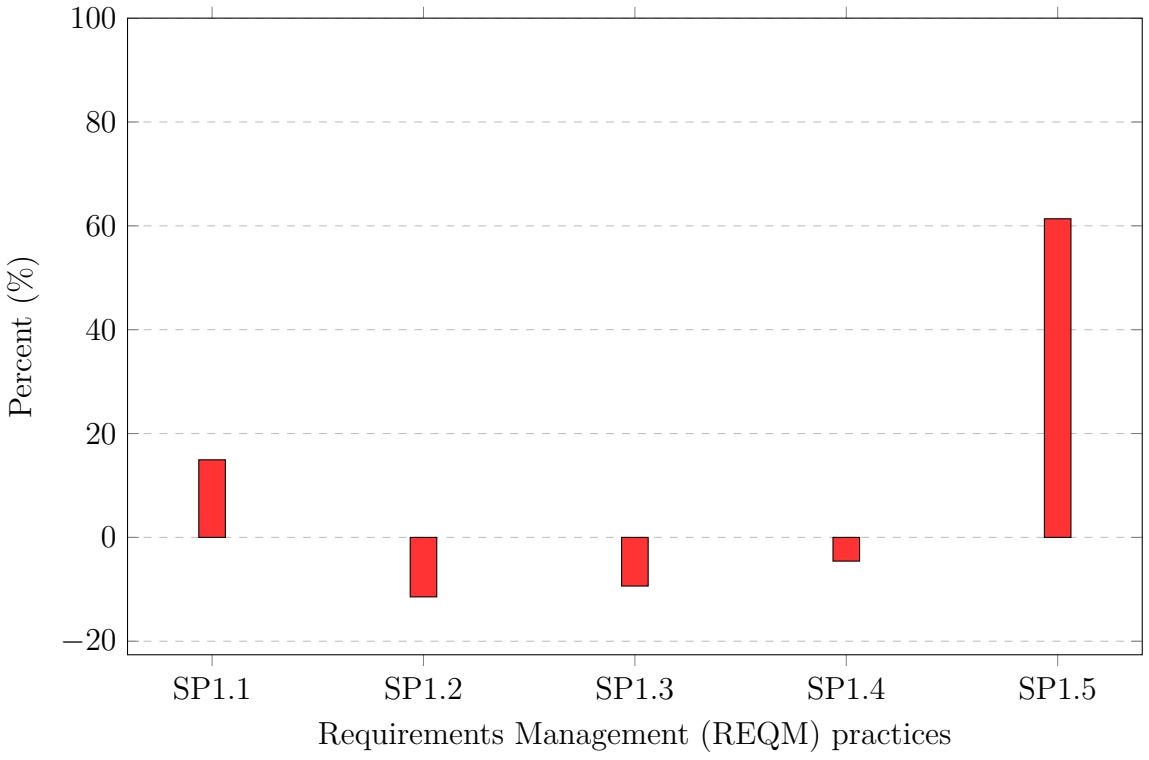
We observed that the similarity engine understands as being semantically different, some expressions that in software development terms represent the same meaning, for example, “to write a plan”, “to develop a plan”, “ to create a plan”, “to make a plan”. Based on this, we consider developing a specific semantic similarity corpus to the similarity engine for future works.

Figure 31 – Project Planning (PP) negative PStories



Source: Author.

Figure 32 – Requirements Management (REQM) negative PStories



Source: Author.

Moreover, some false positives were also returned by Coptic. A possible cause of this result is the limited corpora used in this evaluation, and/or the use of too generic PStories, or even that some practices demand more generic practices. We also understand that it is an improvement for future works by defining a more specific corpus or even represent a negative corpus to balance the evaluation.

Furthermore, in this evaluation, we stated as attendance criteria all positive similarity results. However, we understand that each practice may have a tolerance level based on its characteristics. For example, for certain practices, the attendance criteria may be over 20%; for others, it may be over 40%. This fine-tuning is one of the Coptic functionalities that should be explored in further evaluations.

6.5 Chapter Lessons

In this chapter, we introduced Coptic, which is an intelligent Practice-Evidence classification based on Natural Language Processing and Semantic Similarity.

It attends to the fourth specific goal of developing intelligent software to provide support for evidence classification in SPI Diagnostic. Considering this, we highlight the main findings:

- Coptic works for a first PStories analysis;
- the corpora created by using RUP tasks works for several other PStories;
- for positive PStories Coptic presents a satisfying result;
- for negative PStories the result presented by Coptic need to be improved;
- the software users should perform a fine-tuning for corpora entries to improve the tool performance; and
- other experiments are needed to evaluate better and understand the possibilities.

7 CONCLUSION

In this work, we aim to develop a solution to help SEPGs by providing an intelligent solution to gather and analyze SPI Diagnostic information. Hence, we investigated the state of the art, we specified an ontology to generalize SPI resources, we defined PStory to standardize the information gathering specification, and we developed intelligent software to provide support in SPI Diagnostic.

We presented Base of Knowledge about Software Engineering Practices (Badge), which is a domain ontology that aims to generalize the SPI resources that says "What should be done". The development of this ontology has been vital since it allows us to embed different kinds of resources in the software.

Moreover, we also presented the Practitioner Story (PStory), which is a standard template to write evidence of software engineering practices. To better utilize PStory, we formalized through context-free grammar. This formalization is proper once it enables a more wide user experience in the software.

Furthermore, we presented Collector of Software Process Improvement Evidences (Coptic), which is an intelligent software to match evidence and software engineering practices. This software aims to support SEPGs to perform the SPI Diagnostic. Coptic embed Badge and PStory in order to perform an intelligent interaction with users and an intelligent analysis.

7.1 Future Work

As mentioned previously, there are some future works already identified. Future work is about improving the quality of corpora entries, increasing the number of PStories by practice. Other future work improves semantic similarity considering synonym verbs that have different meanings in the standard English language. Another future work is related to understanding how to fine-tuning for each practice, considering its characteristics and particularities. Moreover, we intend to create a client to perform better evaluation in real world case studies.

BIBLIOGRAPHY

ABRAHAMSSON, P. Commitment to software process improvement—development of diagnostic tool to facilitate improvement1. **Software Quality Journal**, Springer, v. 8, n. 1, p. 63–76, 1999. Cited 2 times at pages 37 and 39.

ANACLETO, A. et al. Método e modelo de avaliação para melhoria de processos de software em micro e pequenas empresas. Florianópolis, SC, 2004. Cited 2 times at pages 24 and 32.

BASILI, V. Software development: a paradigm for the future. In: **[1989] Proceedings of the Thirteenth Annual International Computer Software Applications Conference**. [S.l.: s.n.], 1989. p. 471–485. Cited at page 29.

BEHRENS, H. et al. Xtext user guide. **Dostupné z WWW: http://www. eclipse. org/Xtext/documentation/1_0_1/xtext. html**, p. 7, 2008. Cited at page 72.

BLOOR, M. **Focus groups in social research**. [S.l.]: Sage, 2001. Cited at page 48.

BORST, W. N. **Construction of Engineering Ontologies for Knowledge Sharing and Reuse**. Tese (Doutorado) — University of Twente, Enschede, 1997. Cited at page 43.

BOURQUE, P.; FAIRLEY, R. E. (Ed.). **SWEBOK: Guide to the Software Engineering Body of Knowledge**. Version 3.0. Los Alamitos, CA: IEEE Computer Society, 2014. ISBN 978-0-7695-5166-1. Disponível em: <http://www.swebok.org/>. Cited 2 times at pages 23 and 58.

BOZKAYA, M.; GABRIELS, J.; WERF, J. M. van der. Process diagnostics: a method based on process mining. In: IEEE. **2009 International Conference on Information, Process, and Knowledge Management**. [S.l.], 2009. p. 22–27. Cited 3 times at pages 25, 37, and 39.

BUNDY, A.; WALLEN, L. Context-free grammar. In: **Catalogue of Artificial Intelligence Tools**. [S.l.]: Springer, 1984. p. 22–23. Cited at page 57.

CAMBRIDGE. Cambridge University Press, 2020. Disponível em: <https://dictionary.cambridge.org/dictionary/english/synonym>. Cited at page 69.

CHOMSKY, N. Three models for the description of language. **IRE Transactions on information theory**, IEEE, v. 2, n. 3, p. 113–124, 1956. Cited at page 57.

CHOMSKY, N. Syntactic structures.ś-gravenhage: Mouton. **The Hague, The Netherlands**, 1957. Cited at page 57.

CHOMSKY, N. Reflections on language (pantheon, new york). 1975. Cited at page 57.

CHOWDHURY, G. G. Natural language processing. **Annual review of information science and technology**, Wiley Online Library, v. 37, n. 1, p. 51–89, 2003. Cited at page 69.

CHRISSIS, M. B.; KONRAD, M.; SHRUM, S. **CMMI for development: guidelines for process integration and product improvement**. [S.l.]: Pearson Education, 2011. Cited 3 times at pages 24, 29, and 32.

DAGNINO, A.; CORDES, A. A model to measure organizational readiness for software process improvement. In: IEEE. **2014 9th International Conference on Software Engineering and Applications (ICSOFT-EA)**. [S.l.], 2014. p. 329–336. Cited 3 times at pages 25, 37, and 39.

DAHLGAARD, J. J.; KRISTENSEN, K.; KANJI, G. K. Total quality management and education. **Total Quality Management**, Taylor & Francis, v. 6, n. 5, p. 445–456, 1995. Cited at page 30.

DEMING, W. E. **The new economics for industry, government, education**. [S.l.]: MIT press, 2018. Cited 2 times at pages 23 and 29.

ECAR, M. et al. Software process improvement diagnostic: A snowballing systematic literature review. In: IEEE. **2020 XLVI Latin American Computing Conference (CLEI)**. [S.l.], 2020. p. 156–164. Cited at page 27.

ECAR, M. et al. Badge: Towards a domain ontology for spi resources. In: IEEE. **2020 XLVI Latin American Computing Conference (CLEI)**. [S.l.], 2020. p. 222–231. Cited at page 27.

FERCHICHI, A.; BIGAND, M.; LEFEBVRE, H. An ontology for quality standards integration in software collaborative projects. In: **First International Workshop on Model Driven Interoperability for Sustainable Information Systems. Montpellier, France**. [S.l.: s.n.], 2008. p. 17–30. Cited 4 times at pages 44, 45, 46, and 47.

FOWLER, P.; RIFKIN, S. **Software Engineering Process Group Guide**. Pittsburgh, PA, 1990. Disponível em: <http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=11253>. Cited at page 23.

GAFFO, F.; BARROS, R. Gaia risks: A risk management framework. In: **Proceedings of the 25th International Conference on Computer Applications in Industry and Engineering**. [S.l.: s.n.], 2012. v. 1, n. 2, p. 57–62. Cited 2 times at pages 37 and 39.

GAFFO, F. H. et al. Tool to assess the maturity level of the risk management of a software development process. In: IEEE. **2013 8th Iberian Conference on Information Systems and Technologies (CISTI)**. [S.l.], 2013. p. 1–5. Cited 2 times at pages 37 and 39.

GARTISER, N. et al. A semantic layered architecture for analysis and diagnosis of sme. **Procedia Computer Science**, Elsevier, v. 35, p. 1165–1174, 2014. Cited 4 times at pages 24, 32, 37, and 39.

GASCA-HURTADO, G. P.; HINCAPIÉ, J. A.; MUÑOZ, M. Software process improvement assessment for multimodel environment tool to diagnose an organization. In: IEEE. **2017 12th Iberian Conference on Information Systems and Technologies (CISTI)**. [S.l.], 2017. p. 1–6. Cited 5 times at pages 25, 31, 37, 38, and 39.

GRUBER, T. R. A translation approach to portable ontology specifications. **Knowledge Acquisition**, Academic Press Ltd., London, UK, UK, v. 5, n. 2, p. 199–220, 1993. ISSN 1042-8143. Cited at page 43.

GUIZZARDI, G. **Ontological foundations for structural conceptual models**. Tese (Doutorado) — University of Twente, Enschede, 2005. Cited at page 43.

HERRERA, E. M.; RAMÍREZ, R. A. T. A methodology for self-diagnosis for software quality assurance in small and medium-sized industries in latin america. **The Electronic Journal of Information Systems in Developing Countries**, Wiley Online Library, v. 15, n. 1, p. 1–13, 2003. Cited at page 25.

HUMPHREY, W. S. The software engineering process: definition and scope. In: **Proceedings of the 4th international software process workshop on Representing and enacting the software process**. [S.l.: s.n.], 1988. p. 82–83. Cited at page 23.

HUMPHREY, W. S. **Managing the software process**. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 1989. Cited 2 times at pages 23 and 29.

ISO. ISO, **Information technology — Process Assessment - Part 4: Guidance on use for process 950 improvement and process capability determination**. 2004. Cited at page 29.

IVERSEN, J. Problem diagnosis software process improvement. Citeseer, 1998. Cited at page 23.

JALALI, S.; WOHLIN, C. Systematic Literature Studies: Database Searches vs. Backward Snowballing. In: **Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM Í12**. New York, New York, USA: ACM Press, 2012. p. 29. ISBN 9781450310567. ISSN 19493770. Cited 2 times at pages 33 and 40.

JIANG, J. J.; CONRATH, D. W. Semantic similarity based on corpus statistics and lexical taxonomy. **arXiv preprint cmp-lg/9709008**, 1997. Cited at page 70.

JOHNSON, C. N. The benefits fo pdca. **Quality Progress**, American Society for Quality, v. 35, n. 5, p. 120, 2002. Cited at page 29.

KAMBHATLA, N. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the ACL 2004 on Interactive poster and demonstration sessions**. [S.l.], 2004. p. 22. Cited 2 times at pages 69 and 70.

KHAN, A. A. et al. Systematic literature reviews of software process improvement: A tertiary study. In: SPRINGER. **European Conference on Software Process Improvement**. [S.l.], 2017. p. 177–190. Cited at page 32.

KITCHENHAM, B.; CHARTERS, S. **Guidelines for performing Systematic Literature reviews in Software Engineering (Version 2.3)**. Durham, GB, 2007. 57 p. Cited at page 29.

KOLB, P. Disco: A multilingual database of distributionally similar words. **Proceedings of KONVENS-2008, Berlin**, Citeseer, v. 156, 2008. Cited at page 70.

KOLB, P. Experiments on the difference between semantic similarity and relatedness. In: **Proceedings of the 17th Nordic Conference of Computational Linguistics (NODALIDA 2009)**. [S.l.: s.n.], 2009. p. 81–88. Cited at page 70.

KONTIO, J.; BRAGGE, J.; LEHTOLA, L. The focus group method as an empirical tool in software engineering. In: **Guide to advanced empirical software engineering**. [S.l.]: Springer, 2008. p. 93–116. Cited at page 48.

KONTIO, J.; LEHTOLA, L.; BRAGGE, J. Using the focus group method in software engineering: obtaining practitioner and user experiences. In: IEEE. **Proceedings. 2004 International Symposium on Empirical Software Engineering, 2004. ISESE04.** [S.l.], 2004. p. 271–280. Cited at page 48.

KROLL, P.; KRUCHTEN, P. **The Rational Unified Process Made Easy: A Practitionerś Guide to the RUP: A Practitionerś Guide to the RUP**. [S.l.]: Addison-Wesley Professional, 2003. Cited at page 72.

LIMA, P.; SANTOS, G. Diagnóstico do cenário atual da organização para implementação de iniciativas de melhoria de processo de software. In: **XII Workshop de Teses e Dissertações em Qualidade de Software (WTDQS)**. [S.l.: s.n.], 2014. p. 37–42. Cited at page 25.

MARTÍNEZ-LORENTE, A. R.; DEWHURST, F.; DALE, B. G. Total quality management: origins and evolution of the term. **The TQM magazine**, MCB UP Ltd, v. 10, n. 5, p. 378–386, 1998. Cited at page 29.

MCFEELEY, B. **IDEAL: A Userś Guide for Software Process Improvement.** [S.l.], 1996. Cited 4 times at pages 24, 30, 31, and 32.

MERRIAM-WEBSTER. Merriam-Webster, Incorporated, 2020. Disponível em: <https://www.merriam-webster.com/dictionary/synonym>. Cited at page 69.

MILLER, G. A.; CHARLES, W. G. Contextual correlates of semantic similarity. **Language and cognitive processes**, Taylor & Francis, v. 6, n. 1, p. 1–28, 1991. Cited 2 times at pages 69 and 70.

MINELLA, C. M. d. S.; THIRY, M.; FERNANDES, A. M. da R. Diagnóstico de processos em organizações intensivas em software usando um sistema especialista. **Anais do Computer on the Beach**, p. 169–178, 2015. Cited 5 times at pages 25, 37, 39, 40, and 71.

MONTEIRO, P. et al. Mapping cmmi and rup process frameworks for the context of elaborating software project proposals. In: SPRINGER. **International Conference on Software Quality**. [S.l.], 2013. p. 191–214. Cited at page 72.

MOREIRA, D. S. et al. Autodiagnóstico de processo de software baseado em sistema especialista. **Anais SULCOMP**, v. 6, 2013. Cited 8 times at pages 23, 24, 25, 32, 37, 39, 40, and 71.

MÜNCH, J. et al. **Software process definition and management**. [S.l.]: Springer Science & Business Media, 2012. Cited at page 29.

NEUENDORF, K. A. **The content analysis guidebook**. [S.l.]: Sage, 2016. Cited at page 35.

NOY, N. F.; MCGUINNESS, D. L. et al. **Ontology development 101: A guide to creating your first ontology**. [S.l.]: Stanford knowledge systems laboratory technical report KSL-01-05 and ..., 2001. Cited 3 times at pages 43, 45, and 46.

PARDO, C. et al. An ontology for the harmonization of multiple standards and models. **Computer Standards & Interfaces**, Elsevier, v. 34, n. 1, p. 48–59, 2012. Cited 4 times at pages 44, 45, 46, and 47.

PARDO-CALVACHE, C. J. et al. A reference ontology for harmonizing process-reference models. **Revista Facultad de Ingeniería Universidad de Antioquia**, Universidad de Antioquia, n. 73, p. 29–42, 2014. Cited 2 times at pages 44 and 45.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: An update. **Information and Software Technology**, Elsevier, v. 64, p. 1–18, 2015. Cited at page 40.

PRESSMAN, R. S. **Software engineering: a practitionerś approach**. [S.l.]: Palgrave macmillan, 2005. Cited at page 23.

PRESSMAN, R. S. **Software Engineering-A Practitionerś Approach**. [S.l.]: McGraw-Hill, 2010. Cited at page 23.

PRIES-HEJE, J.; JOHANSEN, J. et al. Spi manifesto. **European system & software process improvement and innovation**, 2010. Cited at page 23.

PRIKLADNICKI, R.; BECKER, C. A.; YAMAGUTI, M. H. Uma abordagem para a realização de diagnóstico inicial em empresas que implementam o mps. br. In: **I Workshop de Implementadores (W2-MPS. BR), Brasília**. [S.l.: s.n.], 2005. Cited at page 25.

QUINQUIOLO, J. M. Avaliação da eficácia de um sistema de gerenciamento para melhorias implantado na área de carroceria de uma linha de produção automotiva. **Taubaté/ SP: Universidade de Taubaté**, 2002. Cited 2 times at pages 24 and 32.

RODRIGUES, E.; LOPES, H. S. Inferência de gramáticas livres de contexto usando programação genética. **I Simpósio Brasileiro de Inteligência Computacional**, 2007. Cited at page 57.

RODRÍGUEZ, A. M. G. et al. Kairós: Intelligent system for scenarios recommendation at the beginning of software process improvement. **Informatica**, v. 42, n. 4, 2018. Cited 5 times at pages 25, 37, 39, 40, and 71.

SALO, O.; ABRAHAMSSON, P. Integrating agile software development and software process improvement: a longitudinal case study. In: IEEE. **Empirical Software Engineering, 2005. 2005 International Symposium on**. [S.l.], 2005. p. 10–pp. Cited at page 29.

SEI, S. E. I. Cmmi® for development, version 1.3, improving processes for developing better products and services. **Software Engineering Institute**, 2010. Cited at page 23.

SHEWHART, W. A.; DEMING, W. E. **Statistical method from the viewpoint of quality control**. [S.l.]: Courier Corporation, 1986. Cited 2 times at pages 23 and 29.

SILVA, M. P. d.; BRANCHER, J. D. Sarasvati: Diagnostic method for software process improvement. In: IEEE. **2017 12th Iberian Conference on Information Systems and Technologies (CISTI)**. [S.l.], 2017. p. 1–6. Cited 7 times at pages 23, 31, 32, 33, 37, 38, and 39.

SOFTEX. Mps. br–melhoria de processo do software brasileiro. 2011. Cited 2 times at pages 24 and 32.

SOLINGEN, D. R. van; BERGHOUT, E. W. **The Goal/Question/Metric Method: a practical guide for quality improvement of software development**. [S.l.]: McGraw-Hill, 1999. Cited at page 33.

STELZER, D.; MELLIS, W. Success factors of organizational change in software process improvement. **Software Process: Improvement and Practice**, Wiley Online Library, v. 4, n. 4, p. 227–250, 1998. Cited 2 times at pages 29 and 30.

TRUJILLO-CASAÑOLA, M. et al. Modelo si. mps. cu para valorar las organizaciones al iniciar la mejora de proceso de software. **Revista Cubana de Ciencias Informáticas**, Universidad de las Ciencias Informáticas, v. 8, p. 92–103, 2014. Cited 2 times at pages 37 and 39.

TRUJILLO-CASAÑOLA, Y. et al. Diagnóstico al iniciar la mejora de proceso de software. **Ingeniería Industrial**, Facultad de Ingeniería Industrial, Instituto Superior Politécnico José . . . , v. 35, n. 2, p. 172–183, 2014. Cited 3 times at pages 37, 39, and 40.

TUROFF, M.; LINSTONE, H. A. The delphi method-techniques and applications. 2002. Cited at page 77.

VASCONCELLOS, F. J. et al. Approaches to strategic alignment of software process improvement: A systematic literature review. **Journal of systems and software**, Elsevier, v. 123, p. 45–63, 2017. Cited at page 29.

WEBSTER, J.; WATSON, R. T. Analyzing the Past To Prepare for the Future : Writing a Review. **MIS Quarterly**, v. 26, n. 2, p. 13–23, 2002. Disponível em: <http://www.jstor.org/stable/4132319>. Cited at page 34.

WOHLIN, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: **Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE Í4**. New York, New York, USA: ACM Press, 2014. p. 1–10. ISBN 9781450324762. ISSN 09505849. Cited 2 times at pages 33 and 34.

ZANNI-MERK, C.; ALMIRON, S.; RENAUD, D. A multi-agents system for analysis and diagnosis of smes. In: SPRINGER. **KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications**. [S.l.], 2011. p. 103–112. Cited 3 times at pages 25, 37, and 39.

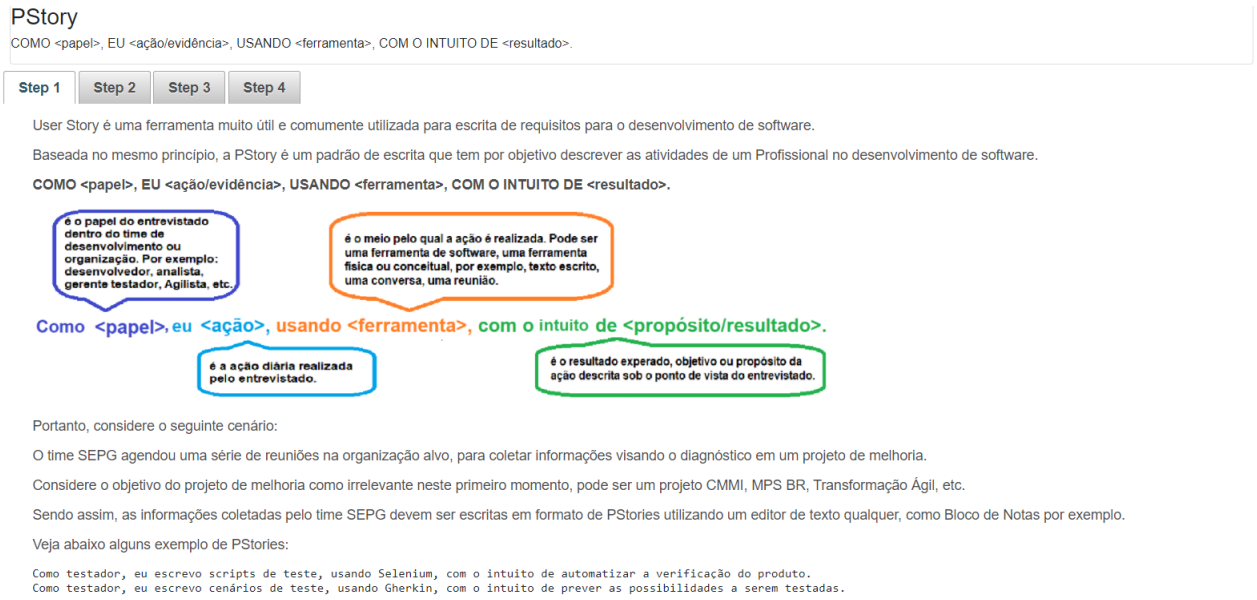
Appendix

APPENDIX A – PSTORY EVALUATION TOOL

Following, we present the execution software screens. The Figure 33 shows the first screen, where there is the template explanation and examples. The Figure 34 shows the second screen, where there is an interview transcription and complete PStories based on it. The Figure 35 shows the third screen, where there is an interview transcription and incomplete PStories based on it. The Figure 36 shows the fourth screen, where there is an interview transcription and no PStories.

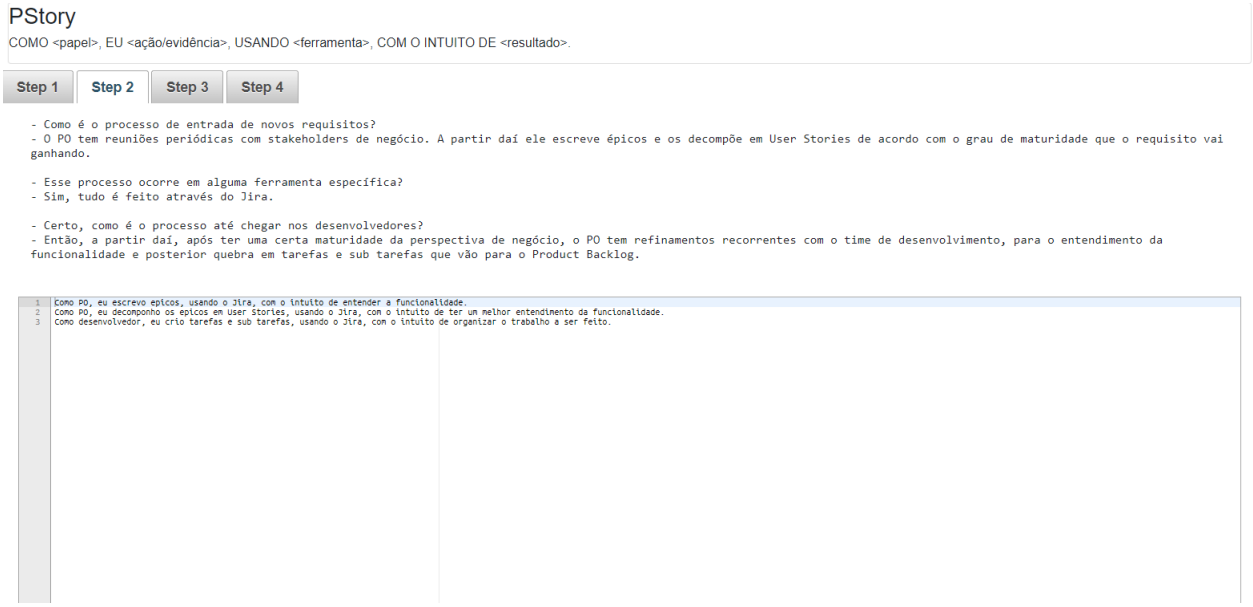
The first and second screens were made in order to understand how the PStory is and how to write and use it. In the third screen the subject should complete the missing information based on the interview transcription. In the fourth screen the subject should write at least one complete PStory based on the transcription.

Figure 33 – First screen.



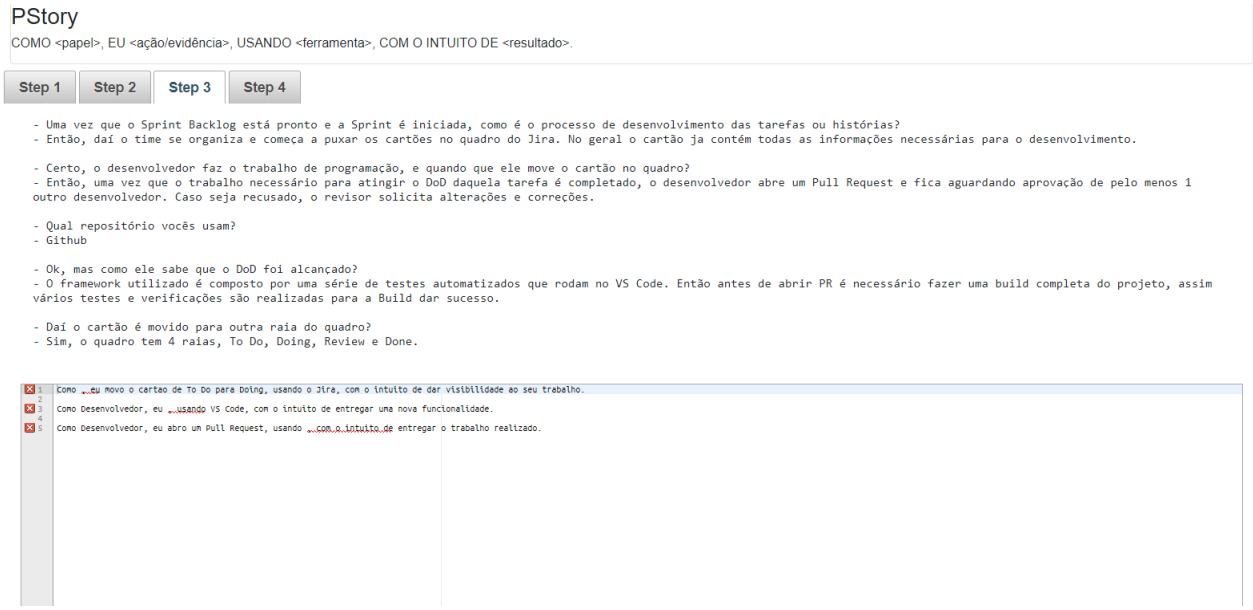
Source: Author.

Figure 34 – Second screen.



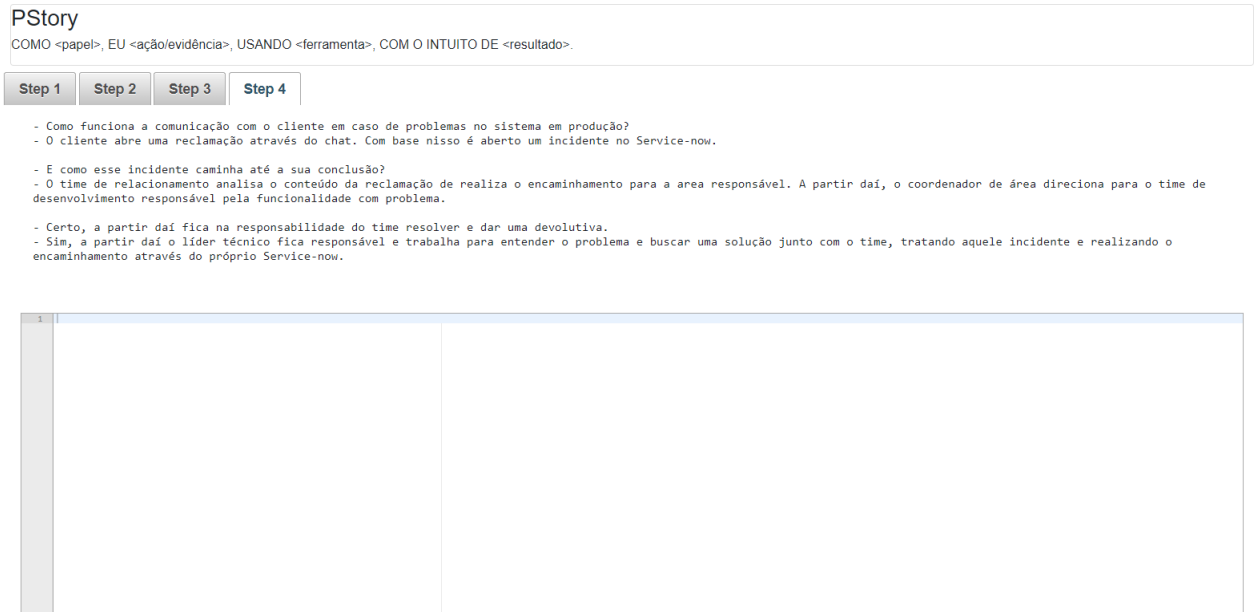
Source: Author.

Figure 35 – Third screen.



Source: Author.

Figure 36 – Fourth screen.



Source: Author.

APPENDIX B – COPTIC EVALUATION

Following, we present the lists of positive and negative PStories with Coptic semantic similarity percent result.

The Table 23 shows the positive PStories for CM area.

Table 23 – CM positive PStories

Code	PStory	Coptic
CM SP1.1	As a project manager, I add all resources that need to be versioned, using GIT, in order to maintain version control.	8.73%
CM SP1.2	As a project manager, I establish the configuration system, using GIT, in order to have version control.	61.16%
CM SP1.3	As a project manager, I establish the workflow, using GitFlow, in order to have a common flow for creating releases.	59.52%
CM SP2.1	As a Project Manager, I monitor the deployment, using Rundeck, in order to track change requests.	26.71%
CM SP2.2	As a Process Engineer, I establish version control process, using GIT, in order to create a common behavior for all.	65.40%
CM SP3.1	As the Project Manager, I review version changes, using GIT, in order to understand the development flow.	32.14%
CM SP3.2	As a Project Manager, I analyze the change history, using Confluence, in order to verify those responsible for the changes.	59.79%

Source: Author.

The Table 24 shows the positive PStories for MA area.

The Table 25 shows the positive PStories for PPQA area.

The Table 26 shows the positive PStories for PMC area.

The Table 27 shows the positive PStories for PP area.

The Table 28 shows the positive PStories for REQM area.

The Table 29 shows the negative PStories for CM area.

The Table 30 shows the negative PStories for MA area.

The Table 31 shows the negative PStories for PPQA area.

The Table 32 shows the negative PStories for PMC area.

The Table 33 shows the negative PStories for PP area.

The Table 34 shows the negative PStories for REQM area.

Table 24 – MA positive PStories

Code	PStory	Coptic
MA SP1.1	As a project manager, I keep the measurement plan, using confluence, in order to always have the updated version.	79.48%
MA SP1.2	As a Process Engineer, I keep the measurement plan, using Confluence, in order to establish the metrics.	79.48%
MA SP1.3	As a Process Engineer, I select the repositories to store collected data, using Confluence, in order to define a versioned storage.	36.20%
MA SP1.4	As the Project Manager, I maintain the measurement plan, using Confluence, in order to always have the updated version.	93.61%
MA SP2.1	As a project manager, I get measurement information, using individual meetings, in order to get the perception of team members.	19.41%
MA SP2.2	As a project manager, I analyze the information collected, using Minitab, in order to generate the indicators.	44.18%
MA SP2.3	As a project manager, I store the information collected, using confluence, in order to maintain a common repository.	41.69%
MA SP2.4	As the Scrum Master, I divulge the result of the improvement program, using Sprint Review, in order to give visibility to the team.	24.34%

Source: Author.

Table 25 – PPQA positive PStories

Code	PStory	Coptic
PPQA SP1.1	As a Project Manager, I apply an evaluative checklist, using Confluence, in order to assess the parameters of the process.	49.37%
PPQA SP1.2	As a technical leader, I perform code inspections, using GitHub, in order to evaluate the code stretches generated by the team.	-16.28%
PPQA SP2.1	As a Project Manager, I organize compliance update meetings, using Google Meet, in order to keep the team informed about the organization's compliance.	39.17%
PPQA SP2.2	As a project manager, I maintain quality practices, using Miro, in order to give visibility to the team of quality goals.	8.78%

Source: Author.

Table 26 – PMC positive PStories

Code	PStory	Coptic
PMC SP1.1	As a project manager, I monitor the progress and use of resources, using spreadsheets, in order to check the project parameters.	65.07%
PMC SP1.2	As a Project Manager, I check the status of each iteration, using Jira, in order to monitor commitment to the project.	96.32%
PMC SP1.3	As a project manager, I do the risk documentation, using a specific spreadsheet, in order to keep the risks under control.	4.99%
PMC SP1.4	As a product manager, I monitor project status using Confluence to adjust milestones.	24.44%
PMC SP1.5	As a project manager, I perform alignment expectation meetings customers, using Partner Radar, in the intention of monitor the customers involvement.	58.61%
PMC SP1.6	As a project manager, I review the land cycle framework, using Jira, in order to align the necessary resources.	56.89%
PMC SP1.7	As the product manager, I review the lifecycle milestone, using Confluence, to monitor project parameters.	58.77%
PMC SP2.1	As a project manager, I report the status of the removal of impediments, using a time from Daily Meeting, in order to give science to the team.	-37.96%
PMC SP2.2	As the product manager, I develop problem solving issues, using Confluence, to assess project iterations.	30.61%
PMC SP2.3	As a project manager, I maintain an impediment monitoring table, using Confluence, in order to increase visibility and progress.	33.58%

Source: Author.

Table 27 – PP positive PStories

Code	PStory	Coptic
PP SP1.1	As a Project Manager, I do a release estimate, using Jira, in order to demonstrate future steps.	60.12%
PP SP1.2	As a Project Manager, I estimate the duration of each task, using MS Project, in order to establish the schedule.	-5.13%
PP SP1.3	As the Project Manager, I establish milestones according to the number of iterations, using MS Project, in order to establish the project life cycle.	17.02%
PP SP1.4	As a project manager, I plan scheduling and work assignment, using Jira, in order to estimate the effort demanded by the project.	2.78%
PP SP2.1	As a Project Manager, I do a release estimate, using Jira, in order to demonstrate future steps.	6.63%
PP SP2.2	As a project manager, I monitor the status of the project, using the JIRA, in order to identify the risks in advance.	63.05%
PP SP2.3	As the Technical Leader, I establish the project directory tree, using Confluence, in order to organize the data.	73.74%
PP SP2.4	As the Infrastructure Coordinator, I define the project group and organization, using confluence, in order to allocate the necessary resources.	64.38%
PP SP2.5	As the Technical Coordinator, I define the skills needed to meet the project scope, using Role Matrix, in order to build an ideal team.	32.36%
PP SP2.6	As the Project Manager, I carry out the scheduling and assignment of work, using Jira, in order to estimate the effort required by the project.	12.84%
PP SP2.7	As a project manager, I refining the development plan, using Jira, in order to detail project planning.	49.30%
PP SP3.1	As a Project Manager, I perform requirements revision, using Jira, in order to visualize alternative paths.	3.07%
PP SP3.2	As the Team Leader, I create reports with the team's deliverability from the past few months using Jira to assess the effectiveness of the current process.	37.79%
PP SP3.3	As a Project Manager, I do a kickoff meeting, using Google Meet, in order to get commitment to the team.	55.95%

Source: Author.

Table 28 – REQM positive PStories

Code	PStory	Coptic
REQM SP1.1	As an analyst, I elicit requirements, using Jira, in order to identify the functionalities.	37.47%
REQM SP1.2	As the Scrum Master, I facilitate Sprint planning, using Jira, in order to get the team's commitment to the Sprint scope.	3.63%
REQM SP1.3	As a PO, I manage requirements changes, using Jira, in order to establish the customer journeys correctly.	48.95%
REQM SP1.4	As the PO, I manage dependencies between requirements, using Jira, in order to clarify the functionality of the system.	61.91%
REQM SP1.5	As a Project Manager, I review the requirements, using Jira, in order to maintain fidelity of the understanding of the features.	1.10%

Source: Author.

Table 29 – CM negative PStories

Code	PStory	Coptic
CM SP1.1	As a Project Manager, I verify the feelings of each team member, using Happiness Radar, in order to understand expectations and individual commitment.	-18.02%
CM SP1.2	As a Project Manager, I keep the measurement polices, using confluence, in order to always have the updated version.	14.63%
CM SP1.3	As a Project Manager, I detail the software requirements, using Jira, in order to manage the requirements changes.	32.52%
CM SP2.1	As a Project Manager, I create release estimatives, using Jira, in order to demonstrate future steps.	-27.80%
CM SP2.2	As a Project Manager, I do Deploys monitoring, using Rundeck, in order to control change requests.	-7.54%
CM SP3.1	As a PO, I manage dependencies between requirements, using Jira, in order to clarify the functionality of the system.	-24.55%
CM SP3.2	As a Project Manager, I conduct follow-up meetings, using Google Meet, in order to get feedback on the software in development.	16.03%

Source: Author.

Table 30 – MA negative PStories

Code	PStory	Coptic
MA SP1.1	As a Project Manager, I design phases and iterations, using Jira, in order to establish project estimates.	-15.03%
MA SP1.2	As an Analyst, I elicit requirements, using Jira, in order to identify functionality.	-40.23%
MA SP1.3	As an Integration Analyst, I write the Configuration Management Plan, using Notion, in order to establish the configuration management system.	55.05%
MA SP1.4	As the Project Manager, I plan the phases and iterations, using Jira, in order to establish project estimates.	20.98%
MA SP2.1	As a Project Manager, I identify and evaluate the risks, using confluence, in order to delimit the scope of the project.	29.97%
MA SP2.2	As a Project Manager, I write third-party software acquisition, using SharePoint, in order to establish the characteristics of the software to be purchased.	4.14%
MA SP2.3	As a Project Manager, I estimate the releases, using Jira, in order to demonstrate future steps.	-13.80%
MA SP2.4	As a Project Manager, I develop a problem solving plan, using Confluece, in order to analyze incidents.	29.71%

Source: Author.

Table 31 – PPQA negative PStories

Code	PStory	Coptic
PPQA SP1.1	As a project manager, I review change requisitions, using TeamCity, in order to revise deployments.	-18.02%
PPQA SP1.2	As a Project Manager, I report impediments removal status, using a time from Daily Meeting, in order to inform the team.	6.77%
PPQA SP2.1	As a Project Manager, I store the collected information, using Confluence, in order to maintain a common repository.	49.06%
PPQA SP2.2	As a Project Manager, I do the risk documentation, using a specific spreadsheet, in order to keep the risks under control.	15.70%

Source: Author.

Table 32 – PMC negative PStories

Code	PStory	Coptic
PMC SP1.1	As a Scrum Master, I publish the improvement program results, using Sprint Review, in order to give visibility to the team.	23.45%
PMC SP1.2	As the Project Manager, I develop project planning, using Confluence, in order to establish project plan.	20.42%
PMC SP1.3	As a Project Manager, I develop the Quality Assurance Plan, using Notion, in order to evaluate the process.	-1.82%
PMC SP1.4	As the Project Manager, I develop vision document, using Confluence, in order to plan stakeholder engagement.	51.98%
PMC SP1.5	As a Project Manager, I update the roadmap, using Roadmap, in order to view the progress of the project.	32.03%
PMC SP1.6	As a Project Manager, I develop requirements management plan, using Confluence, in order to understand the requirements.	-6.10%
PMC SP1.7	As a Project Manager, I analyze the change history, using Confluence, in order to verify those responsible for the changes.	61.08%
PMC SP2.1	As a Project Manager, I analyze the history of changes, using Confluence, in order to check those responsible for changes.	-13.76%
PMC SP2.2	As a Project Manager, I monitor the status of the project, using the JIRA, in order to identify the risks in advance.	21.32%
PMC SP2.3	As a Project Manager, I perform the development plan refining, using Jira, in order to detail project planning.	31.26%

Source: Author.

Table 33 – PP negative PStories

Code	PStory	Coptic
PP SP1.1	As a Project Manager, I evaluate the risks, using Risk Matrix, in order to delimit the scope of the project.	8.51%
PP SP1.2	As a the Project Manager, I do Deploy's monitoring, using Rundeck, in order to control change requests.	6.71%
PP SP1.3	As the Project Manager, I review the version updates, using GIT, in order to understand the development flow.	6.13%
PP SP1.4	As a Tech Lead, I perform code inspections, using GitHub, in order to evaluate the code stretches generated by the team.	-20.94%
PP SP2.1	As a Project Manager, I review the requirements, using Jira, in order to get the requirements' commitment.	-12.81%
PP SP2.2	As a Project Manager, I review the requirements, using Jira, in order to maintain the fidelity of the understanding of the functionalities.	-9.43%
PP SP2.3	As a Solutions Architect, I do architectural analysis, using a diagram of components, in order to establish the project architecture.	25.57%
PP SP2.4	As the Project Manager, I plan phases and iterations, using Jira, in order to establish project estimates.	-23.99%
PP SP2.5	As a PO, I manage requirements changes, using Jira, in order to establish the customer journeys correctly.	-15.33%
PP SP2.6	As the Project Manager, I conduct project reviews, using Notion, in order to establish records.	5.43%
PP SP2.7	As a Project Manager, I review the development progress, using Kanban, in order to understand how progress is.	45.54%
PP SP3.1	As the PO, I write epics, using Jira, in order to understand the functionality.	-37.59%
PP SP3.2	As the Team Leader, I create reports with the team deliverability from the last five sprints, using Jira, in order to assess the effectiveness of the current process.	6.11%
PP SP3.3	As a Project Manager, I identify and evaluate the risks, using confluence, in order to delimit the scope of the project.	-4.21%

Source: Author.

Table 34 – REQM negative PStories

Code	PStory	Coptic
REQM SP1.1	As a Project Manager, I develop a problem solving plan, using Confluence, in order to analyze incidents.	14.94%
REQM SP1.2	As a Project Manager, I verify the status of each iteration, using Jira, in order to monitor the commitment to the project.	-11.46%
REQM SP1.3	As the Project Manager, I review change orders, using TeamCity, in order to review deployments.	-9.37%
REQM SP1.4	As the Project Manager, I plan the phases and iterations, using Jira, in order to establish project estimates.	-4.57%
REQM SP1.5	As a Process Engineer, I monitor the project status, using Notion, in order to get the measurement data.	61.36%

Source: Author.

APPENDIX C – COPTIC EVALUATION - CORPORA

Following, we present the RUP based corpora used to perform the evaluation.

The Table 35 shows the corpora PStories for CM area.

Table 35 – CM PStories

Code	PStory
CM SP1.1	As an integration analyst, I write resources version control polices, using Tool, in order to identify the configuration items.
CM SP1.2	As an integration analyst, I write the Configuration Management polices, using Tool, in order to establish the configuration management system.
CM SP1.3	As an Integration Analyst, I develop the Iteration Plan, using Tool, in order to create a baseline.
CM SP2.1	As an integration analyst, I establish the process of control of change, using Tool, in order to track the change requests.
CM SP2.2	As an integration analyst, I create a development workspace, using Tool, in order to control the configuration items.
CM SP3.1	As an integration analyst, I create an integration workspace, using Tool, in order to establish configuration management.
CM SP3.2	As a integration analyst, I verify the changes in Build, using Tool, in order to perform audit in the configuration.

Source: Author.

The Table 36 shows the corpora PStories for MA area.

The Table 37 shows the corpora PStories for PPQA area.

The Table 38 shows the corpora PStories for PMC area.

The Table 39 shows the corpora PStories for PP area.

The Table 40 shows the corpora PStories for REQM area.

Table 36 – MA corpora PStories

Code	PStory
MA SP1.1	As a process engineer, I develop the measurement plan, using Tool, in order to establish the measurement objectives.
MA SP1.2	As a process engineer, I develop the measurement plan, using Tool, in order to specify the measures.
MA SP1.3	As a process engineer, I develop the measurement plan, using Tool, in order to specify the data collections and storage procedures.
MA SP1.4	As a process engineer, I develop the measurement plan, using Tool, in order to specify the analysis procedures.
MA SP2.1	As a process engineer, I monitor the status of the project, using Tool, in order to obtain the measurement data.
MA SP2.2	As a process engineer, I monitor the status of the project, using Tool, in order to analyze the measurement data.
MA SP2.3	As a process engineer, I monitor the status of the project, using Tool, in order to store the data and results.
MA SP2.4	As a process engineer, I report the status, using Tool, in order to communicate the measurement results.

Source: Author.

Table 37 – PPQA corpora PStories

Code	PStory
PPQA SP1.1	As a project manager, I develop the Quality Assurance Plan, using Tool, in order to evaluate the process.
PPQA SP1.2	As a project manager, I develop the Quality Assurance Plan, using Tool, in order to evaluate the work products.
PPQA SP2.1	As a project manager, I report the status, using Tool, in order to communicate to resolve nonconformities.
PPQA SP2.2	As a project manager, I conduct review, using Tool, in order to establish the records.

Source: Author.

Table 38 – PMC corpora PStories

Code	PStory
PMC SP1.1	As a project manager, I monitor the status of the project, using Jira, in order to maintain project control.
PMC SP1.2	As a project manager, I verify the status of each iteration, using Jira, in order to monitor the commitment to the project.
PMC SP1.3	As a project manager, I identify the risks, using the list of risks, in order to monitor the risks.
PMC SP1.4	As a project manager, I create data monitoring plan, using Confluence, in order to monitor the project information.
PMC SP1.5	As a project manager, I perform follow up meetings, using Google Meet, in order to get feedback on developing software.
PMC SP1.6	As a project manager, I review the progress of the life cycle, using Jira, in order to monitor project progress.
PMC SP1.7	As a project manager, I review the requisitions of change, using TeamCity, in order to revise deployments.
PMC SP2.1	As a project manager, I analyze exceptions and problems, using Confluence, in order to analyze incidents.
PMC SP2.2	As a project manager, I develop a problem solving plan, using Confluence, in order to analyze incidents.
PMC SP2.3	As a project manager, I develop a problem solving plan, using Confluence, in order to analyze incidents.

Source: Author.

Table 39 – PP corpora PStories

Code	PStory
PP SP1.1	As a project manager, I develop the iteration plan, using confluence, in order to estimate the scope of the project.
PP SP1.2	As a project manager, I plan the phases and iterations, using Jira, in order to establish project estimates.
PP SP1.3	As a project manager, I plan the phases and iterations, using Jira, in order to establish project estimates.
PP SP1.4	As a project manager, I plan the phases and iterations, using Jira, in order to establish project estimates.
PP SP2.1	As a project manager, I plan the phases and iterations, using Jira, in order to establish project estimates.
PP SP2.2	As a project manager, I report the status of the tasks, using Jira, in order to identify the risks of the project.
PP SP2.3	As a project manager, I write the project configuration plan, using Confluence, in order to plan data management.
PP SP2.4	As a project manager, I adapt the project development process, using Jira, in order to plan project resources.
PP SP2.5	As a project manager, I define organizational and team project, using Confluence, in order to plan knowledge and skills required.
PP SP2.6	As a project manager, I develop vision document, using confluence, in order to plan stakeholder involvement.
PP SP2.7	As a project manager, I develop project planning, using confluence, in order to establish project plan.
PP SP3.1	As a project manager, I review project planning, using confluence, in order to review work and resources.
PP SP3.2	As a project manager, I review the landmark landmark, using Jira, in order to harmonize work and resources.
PP SP3.3	As a project manager, I compile software development plan, using Jira, in order to obtain a plan to commit.

Source: Author.

Table 40 – REQM corpora PStories

Code	PStory
REQM SP1.1	As a project manager, I develop Requirements Management Plan, using Confluence, in order to understand the requirements.
REQM SP1.2	As a project manager, I review the requirements, using Jira, in order to obtain the commitment of the requirements.
REQM SP1.3	As a project manager, I detail the software requirements, using Jira, in order to manage the requirements changes.
REQM SP1.4	As a project manager, I manage the dependencies, using Jira, in order to maintain the traceability of the requirements.
REQM SP1.5	As a project manager, I submit a change requisition, using Jira, in order to certify the alignment between project and requirements.

Source: Author.