Using Scrum for Implementing IT Governance with COBIT 5

Ana Cláudia Martins Amorim

Thesis to obtain the Master of Science Degree in Information Systems and Computer Engineering

Supervisors: Prof.Miguel Mira da Silva
Prof. Rúben Pereira

Examination Committee

Chairperson: Prof.José Luís Brinquete Borbinha
Supervisor: Prof.Miguel Mira da Silva
Members of the Committee: Prof. Rui António dos Santos Cruz

October 2018
Acknowledgments

I would like to thank my parents for their friendship, support, encouragement and caring over all these years, for raising me to be here today and teaching me the values that I will always carry through life. I would also like to thank my grandparents for always believing in me, for always wanted to know what I was doing even if they do not understand exactly what it is, and for even so, being my number one fans.

To my forever inspiration in life, always supportive late grandmother "Avó Jeninha" for being always there for me since I was born, and for keeping me from harm and guiding me through life. She was a warrior and a true role model in my life, without her non of this work would be possible.

I would also like to acknowledge my dissertation supervisors Prof. Miguel Mira da Silva and Prof. Rúben Pereira for their insight, support and sharing of knowledge that has made this Thesis possible.

To my always patient and caring boyfriend, who read all single pages of this Thesis to help me revise the report, while also provided me with emotional support and encouragement.

I have also to thank to my favorite aunt, that is like a sister to me, who have always supported my decisions in every single moment of my life.

Last but not least, to all my friends that were always there for me, specially during this important phase of my life called college. Who cried with me in every difficult task I felt I could not complete, and celebrated every victory I conquered. Thank you all.

To each and every one of you – Thank you.
Abstract

COBIT 5 is a widely-used framework for implementing sound governance of enterprise IT (GEIT). Currently, the ISACA’s official implementation solution follows a sequentially ordered process that is still not the most suited to practitioners in general, raising several issues related with lack of commitment from top management and misaligned solutions. Nevertheless, new project life-cycle strategies have emerged along with the agile paradigm for project management, providing flexible and adaptable environments for projects where the solution is complex and not clear, delivering the product incrementally with feedback loops. With this research we aim to eliminate some known challenges of COBIT 5 by providing a Scrum based methodology to address COBIT 5 adoptions. A Design Science Research Methodology was used to guide this work, where two iterations on the solution development, demonstration and evaluation activities were performed. With two different solutions and demonstrations, we were able to identify some relevant findings with positive results regarding the objectives established.

Keywords

COBIT 5; Project Management Strategies; Scrum; Scaled Scrum; Governance of Enterprise IT; Agile.
Resumo

O COBIT 5 é uma framework muito usada para implementar uma forte governança corporativa de TI (GEIT). Atualmente, a solução oficial de implementação da ISACA segue um processo ordenado sequencialmente que ainda não é o mais adequado para os profissionais em geral, levantando vários problemas relacionados com a falta de compromisso da alta administração e soluções desalinhadas do ambiente da organização. Contudo, novas estratégias para abordar o ciclo de vida de um projeto surgiram junto com o paradigma ágil, fornecendo ambientes flexíveis e adaptáveis para projetos em que a solução é complexa e não muito clara, fornecendo o produto incrementalmente com loops de feedback. Com essa pesquisa, pretendemos eliminar alguns desafios conhecidos do COBIT 5, fornecendo uma abordagem baseada no Scrum para abordar os programas de COBIT 5. Uma Metodologia de Pesquisa em Design Science foi usada para orientar este trabalho, onde duas iterações sobre as atividades de desenvolvimento, demonstração e avaliação da solução foram realizadas. Com duas abordagens e demonstrações diferentes, pudemos identificar algumas descobertas relevantes com resultados positivos no que toca aos objetivos estabelecidos.

Palavras Chave

COBIT 5; Estratégias de Gestão de Projeto; Scrum; Scrum Escalável; Governança Corporativa de TI; Agile.
Contents

1 Introduction 2
  1.1 Research Methodology ................................................. 4

2 Research Problem 6
  2.1 Problem Identification and Motivation .............................. 7
    2.1.1 Interviews .......................................................... 7
    2.1.2 Problem Statement ............................................... 8
  2.2 Objectives ............................................................... 9

3 Theoretical Backgrounds and Related Works 11
  3.1 Theoretical Background ............................................... 12
    3.1.1 Governance of Enterprise IT .................................. 12
    3.1.2 Using COBIT 5 for GEIT Implementation ..................... 13
    3.1.3 Project Management Methodologies ............................ 15
      3.1.3.A Traditional Project Management: Waterfall ............ 15
      3.1.3.B Agile Project Management: Scrum ...................... 16
  3.2 Requirement analysis and project planning ......................... 18
    3.2.1 Water-Scrum-Fall ................................................ 19
    3.2.2 Business Process Modeling with Scrum ..................... 20
  3.3 Scaling agile methods ................................................ 21
    3.3.1 SAFe ............................................................... 22
    3.3.2 LeSS ................................................................. 23
    3.3.3 Nexus ............................................................... 23
    3.3.4 Scaling agile frameworks’ analysis ............................ 24
  3.4 Agility in Non-software Development Areas ......................... 26

4 First Iteration 28
  4.1 Description ............................................................. 29
  4.2 Design and Development .............................................. 29
    4.2.1 Specification of Scrum concepts .............................. 29
4.2.2 The life-cycle model .................................................. 32
4.3 Demonstration ......................................................... 33
  4.3.1 Context .............................................................. 34
  4.3.2 Results ............................................................. 35
4.4 Evaluation .............................................................. 36
  4.4.1 Interviews with experts .......................................... 36
  4.4.2 Evaluation of the demonstration ................................ 39
  4.4.3 Conclusion .......................................................... 42
5 Second Iteration .......................................................... 43
  5.1 Design and Development ............................................ 44
    5.1.1 Specification of SAFe concepts ................................ 44
    5.1.2 The life-cycle model ........................................... 51
  5.2 Demonstration ....................................................... 52
    5.2.1 Context .......................................................... 52
    5.2.2 Results .......................................................... 54
  5.3 Evaluation ............................................................ 55
    5.3.1 Evaluation of the demonstration ............................. 55
    5.3.2 Moody and Shanks quality model framework .............. 58
    5.3.3 Wand and Weber method ..................................... 61
    5.3.4 Conclusion ...................................................... 62
6 Conclusion ...................................................................... 64
  6.1 Discussion .............................................................. 65
  6.2 Limitations ............................................................ 66
  6.3 Contributions .......................................................... 66
  6.4 Communication ....................................................... 67
  6.5 Future Work ............................................................ 67
A Appendix A: Interviews with experts ................................. 72
B Appendix B: Interviews for the 1st demonstration .................. 80
C Appendix C: Interviews for the 2nd demonstration .................. 83
D Appendix D: Results from the interviews ............................... 87
List of Figures

1.1 DSRM process (adapted from [1]) ................................. 5
3.1 COBIT 5 life-cycle .................................................. 14
3.2 Project Management Triangle ................................. 17
3.3 Scrum life-cycle .................................................. 18
3.4 Waterfall (traditional) vs Scrum (agile) (taken from [2]) .................................................. 19
3.5 Water-Scrum-Fall (taken from [3]) ................................. 20
3.6 BPM with Scrum (taken from [4]) .................................. 21
3.7 Essential SAFe (taken from [5]) .................................. 23
3.8 LeSS events (right image) and LeSS teams (left image) .................................................. 24
3.9 Nexus life-cycle (taken from [6]) .................................. 25
4.1 Example of a COBIT 5 practice (APO05) of the process APO12 Manage Risk (taken from [7]) .................................................. 30
4.2 Scrum roles and artifacts ........................................... 30
4.3 COBIT 5 life-cycle model ........................................... 33
5.1 Scaled roles and artifacts for a COBIT 5 programme .................................................. 50
5.2 Scaled events for a COBIT 5 programme .................................................. 51
5.3 Life-cycle model .................................................. 51
5.4 Backlog management in Trello ...................................... 58
5.5 Rating results .................................................. 59
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Described phases for GEIT Implementation with COBIT 5 (from [8])</td>
<td>15</td>
</tr>
<tr>
<td>4.1</td>
<td>Scrum artifacts for a COBIT 5 programme</td>
<td>31</td>
</tr>
<tr>
<td>4.2</td>
<td>Scrum roles for a COBIT 5 programme</td>
<td>31</td>
</tr>
<tr>
<td>4.3</td>
<td>Scrum events for a COBIT 5 programme</td>
<td>32</td>
</tr>
<tr>
<td>4.4</td>
<td>Participant’s information</td>
<td>34</td>
</tr>
<tr>
<td>4.5</td>
<td>Criteria select for evaluating the artifact’s performance</td>
<td>36</td>
</tr>
<tr>
<td>4.6</td>
<td>Interviewee’s information</td>
<td>37</td>
</tr>
<tr>
<td>4.7</td>
<td>Results from the interviews</td>
<td>37</td>
</tr>
<tr>
<td>4.8</td>
<td>Quotes from the interviews</td>
<td>38</td>
</tr>
<tr>
<td>4.9</td>
<td>Results from close response questions</td>
<td>41</td>
</tr>
<tr>
<td>5.1</td>
<td>SAFe roles for a COBIT 5 programme</td>
<td>45</td>
</tr>
<tr>
<td>5.2</td>
<td>SAFe events for a COBIT 5 programme</td>
<td>46</td>
</tr>
<tr>
<td>5.3</td>
<td>SAFe artifacts for a COBIT 5 programme</td>
<td>49</td>
</tr>
<tr>
<td>5.4</td>
<td>Role’s information</td>
<td>53</td>
</tr>
<tr>
<td>5.5</td>
<td>Interviewee’s information</td>
<td>55</td>
</tr>
<tr>
<td>5.6</td>
<td>Quality factors and metrics used for assessing the model using Moody and Shanks</td>
<td>59</td>
</tr>
<tr>
<td>5.7</td>
<td>Ontological deficiencies identified by the Wand and Weber method</td>
<td>61</td>
</tr>
</tbody>
</table>
# Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>GEIT</td>
<td>Governance of Enterprise IT</td>
</tr>
<tr>
<td>COBIT</td>
<td>Control Objectives for Information and Related Technologies</td>
</tr>
<tr>
<td>SDLC</td>
<td>Software Development Life Cycle</td>
</tr>
<tr>
<td>DSRM</td>
<td>Design Science Research Methodology</td>
</tr>
<tr>
<td>DSR</td>
<td>Design Science Research</td>
</tr>
<tr>
<td>BPM</td>
<td>Business Process Modeling</td>
</tr>
<tr>
<td>UniLEO</td>
<td>Unidade de Implementação da Lei do Enquadramento Orçamental</td>
</tr>
<tr>
<td>SAFe</td>
<td>Scaled Agile Framework</td>
</tr>
<tr>
<td>LeSS</td>
<td>Large Scale Scrum</td>
</tr>
<tr>
<td>EA</td>
<td>Enterprise Architecture</td>
</tr>
</tbody>
</table>
1 Introduction

Contents

1.1 Research Methodology .................................................. 4
Information Technology (IT) and related information technologies are pervasive not only in simple
day to day functions everywhere, but also playing critical roles in most organizations. As it grows, IT
has become a business enabler, taking major responsibility in effective and efficient product delivery,
bringing more value to stakeholders [9,10].

Adding value, while managing enterprise's use of IT and mitigating the IT risks, made Governance
of Enterprise IT (GEIT) a necessity for accomplishing enterprise's goals, and therefore, a fundamental

A succeeded GEIT implementation requires a culture of well-defined enablers as organizational
structures, principles and structured governance and management processes according the enterprise's
vision [11]. Frameworks are the best tools to help this implementation [12] since they provide adaptable
solutions for any type of organization. For that purpose, ISACA developed Control Objectives for In-
formation and Related Technologies (COBIT) [11] as a framework to “provide guidance in evaluating,
directing and monitoring an enterprise's use of IT” [8], giving an objective way for companies to align
business strategies with IT goals.

Although COBIT 5 is widely recognized as one of the most used GEIT frameworks [13], it is also
considered too large and complex, taking years for its full adoption [14, 15]. When we add to its natural
complexity the challenges associated with implementing IT Governance solutions, the task becomes
even more demanding [16].

Concerning the fact, ISACA developed an implementation guide [8] which suggests a linear strategy
of seven successive phases for a COBIT 5 adoption. Traditional methods are defined by an exhaustive
planning phase, which eliminates the need for any changes, so the process follows the original plan,
going through all phases in an orderly fashioned way [17].

Meanwhile, agile approaches such as Scrum and Kanban appeared as an alternative paradigm to
the traditional ones. They are specially focused on building solutions that are not clear at the beginning
[17], by putting the client's interests first, embracing constant feedback and welcoming changes in the
requirements at any time of the process [18]. Although these methodologies are mainly applied in
the software development field, organizations are starting to adopt this new paradigm for managing
their projects in several other areas beyond software development such as consulting, manufacturing,
coaching, etc. [19].

In this document, we focus our research on Scrum, discussing the application of agile methodologies
on GEIT implementations with COBIT 5, with the main goal of using agile principles to overcome some
known challenges of COBIT 5 such as lack of support from top management, failure to understand the
environment, resistance to change and scope misalignment [8].

Two iterations on Design Science Research (DSR) process were performed in order to produce the
most accurate solution possible. Starting on the design and development step, each iteration of the
solution was demonstrated in a distinct COBIT 5 programme, and evaluated with interviews in order to compare our goals with the results observed from its usage.

This document is structured as follows. We begin this work in chapter 1 with a brief explanation of the motivation and the research methodology used in the Thesis in Section 1.1. Next we introduce the research problem in Chapter 2, where we start by explaining our problem and describe the main objectives for this research in Section 2.2. This is followed by the state-of-art regarding the scope of this research in Chapter 3, where we start by explaining the basic concepts to understand the problem, with an introduction to the theoretical background in Section 3.1. Chapter 4 presents the design and development, demonstration and evaluation of the first iteration to the DSR process. In Chapter 5 we describe the same process steps for the second iteration. Finally, in Chapter 6 the document concludes with an overview on lessons learned, limitations, contributions, communication and future work.

1.1 Research Methodology

Design Science Research Methodology (DSRM) was the methodology chosen to guide this research work. DSRM provides a process model for doing research in Information Systems and other applied resource disciplines, as well as a mental model for reviewers to evaluate researchers [1]. The main goal of design science in IS research is to produce and evaluate an IT artifact that supports the solution for an identified organizational problem [20]. The artifacts produced can be constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), and instantiations (implemented and prototype systems). The DSR process (Fig. 1.1) establishes 6 phases [1] and our research conforms as described next:

1. **Problem identification and motivation**: Identify and define the research problem, justifying the value of a solution and motivating the researcher to investigate the answer.

2. **Define the objectives for a solution**: Infer the quantitative or qualitative objectives of a desirable solution for the problem defined and knowledge of what is possible and feasible.

3. **Design and development**: Create the design research artifact. Such artifacts can be constructs, models, methods or instantiations created to address the designated problem.

4. **Demonstration**: Demonstrate the use of the artifact to solve one or more instances of the problem.

5. **Evaluation**: Observe and measure how well an artifact supports a solution to the problem, comparing the objectives to the results observed from use of the artifacts in the demonstration.

6. **Communication**: Communicate the problem, the artifact, its utility, novelty and effectiveness to researchers and other relevant audiences.
**Figure 1.1:** DSRM process (adapted from [1])

- Identify Problem & Motivate:
  - Lack of adoption of formal project management methodologies on COBIT 5 programmes

- Define Objectives of a Solution:
  - Facilitate COBIT 5 adoptions, decreasing some known challenges

- Design & Development:
  - Serum-based methodology for COBIT 5

- Demonstration:
  - Field study at Portuguese Finance Ministry - 1st iteration
  - Field study at bank X - 2nd iteration

- Evaluation:
  - Metrics, Analysis, Knowledge
    - Models and Frameworks
    - Interviews
    - Field studies

- Communication:
  - Dissertation and papers

Possible Research Entrypoints
2

Research Problem

Contents

2.1 Problem Identification and Motivation ........................................... 7
2.2 Objectives ................................................................. 9
2.1 Problem Identification and Motivation

This section is related with the “Problem Identification and Motivation” step from DSR model (step 1), introduced in section 1.1 as the research methodology used in this work.

To better understand how practitioners deal with COBIT 5 adoptions and what are the main challenges, we made some interviews at ISACA Lisbon Conference to three international members of ISACA and two IT auditors from Portugal. Next, we synthesize the problem and explain the main motivation for this thesis work. We conclude with the objectives established.

2.1.1 Interviews

As aforementioned, ISACA provides guidance on how to implement COBIT 5 in the last version of COBIT [8]. However, many are the problems remaining when organizations try to adopt this framework.

Unfortunately, the research on COBIT 5 is scarce, as we can confirm from a quick search on Google Scholar, where COBIT 5 [11] has 86 citations and COBIT 5 implementation guide [8] has just 5. As Steven De Haes et al. said “academic knowledge base to analyze and leverage COBIT 5 from a research perspective is limited” [14].

There is much more to explore in this framework as it is still not completely suited to the current environment of most organizations, which is proved by the challenges raised from adopting or migrating to COBIT 5. We can even find a full chapter of “practical issues that need to be overcome” on the implementation guide [8], that aims to raise awareness for common challenges found during the implementation such as lack of senior management commitment and support, communication issues, failure to understand the environment, resistance to change and scope misaligned with requirements.

In order to overcome this lack of literature in COBIT 5 and to better understand the challenges lived by organizations when trying to adopt this framework, we went to the ISACA conference in Lisbon (2ª Conferência Internacional ISACA Lisbon Chapter) and interviewed some of the most important members of ISACA all over the world.

Mike Hughes

*ISACA International Board Member Director*

“I think there is a big challenge around the implementation of COBIT, there is a lot of misunderstanding about what COBIT is […] organizations take the COBIT publications try to implement it, they go to training courses, they open the books and think: this is more difficult than I thought.”.

“We can always do more; we can provide organizations some tools that makes it a lot easier to understand and to implement.”

1 https://scholar.google.pt/, last visited on September 22
António Ramos  
*ISACA Madrid Chapter Former President*

“Very few companies implement COBIT 5. In Spain is very difficult to find companies that have implemented. Some companies have implemented COBIT 4 but did not make the transition to the new one.”

“I have used 3 years ago COBIT focused on security [...] The implementation is usually partial, no one implements the full version.”

“People do not think in using COBIT because they see the implementation as a daunting task and want to be more agile, they want to be more quick.”

Marc Vael  
*ISACA Belgium Chapter President*

“People pick the pieces out of it [...] where I come from, in Belgium, people do not use it 100%. They follow the framework, the concept, but do not follow it religiously, they pick the pieces.”

“The problem with COBIT 5 is that if you do not have enough maturity in applying it and you take a topic, for example privacy, you do whole of COBIT and that is too much and it takes too long, it has to be used as an inspiration source. [...] The implementation is very generic”.

IT auditor X  
*IT auditor in a bank*

“Some areas such as security use COBIT, however use it more for consultation.”

IT auditor Y  
*IT auditor in an international IT organization*

“COBIT 5 is very overarching, in Portugal is only used in an audit perspective. I do not know any organization who decides to implement the whole framework.”

“When well implemented, COBIT 5 can bring much value to the organization. However, it requires a lot of knowledge.”

“Organizations think that is almost impossible to implement.”

2.1.2 Problem Statement

From the previous statements, we can understand that COBIT 5 is a well-known framework that brings much value to organizations. Some (more governmental) have to adopt COBIT to be in com-
pliance with regulations (as an obligation), some use it to minimize the risk in some specific IT-related areas, but few are the ones that cover the whole organization sectors with the full version of COBIT 5.

Despite the existence of official manuals to support its adoption [8,12], studies on COBIT 5 are scarce [14], as previously mentioned, specially regarding lessons learned of previews implementations and practical examples. Case studies and lessons learned can be fundamental for practitioners, especially when COBIT 5 is such a complex and overarching framework, having 7 enablers, 37 processes and more than 6188 interdependencies between them, which makes most organizations to consider these adoptions a daunting task [15,21,22] - “Putting processes in places at any organization that covers even a fraction of what is encompassed in, for example, COBIT 5, is highly challenging” [16].

Furthermore, we can also realize that the official guidance [8] for adopting COBIT 5 is more used as an inspiration source, and most organizations do not follow the method strictly, performing instead ad-hoc implementations.

Its complexity is part of the reason why organizations avoid the official guides, preferring to apply an ad-hoc and partial implementation. This problem leads to non guided adoptions which can consequently result in waste of resources such as time and budget.

More than a project, COBIT 5 adoption is considered in its manual as a programme for implementing GEIT enablers. A programme is considered by Project Management Institute as a “group of projects managed in a coordinated way to obtain benefits not available from managing them individually.”

Being a programme, COBIT 5 initiatives must be managed accordingly, using project development methodologies, and the desired scaled methods, in order to bring consistency, flexibility and security to project teams, increasing the probability of project’s success [23].

Thus, the problem identified is the lack of adoption of formal project management methodologies on COBIT 5 programmes.

2.2 Objectives

This section is related with the “Define the objectives for a solution” step from DSRM explained in section 1.1, as the we intend to identify the objectives of a solution for the problem stated in the previous section 2 (step 2 of DSRM).

The main goal is to facilitate COBIT 5 adoptions, decreasing some known challenges through the usage of an agile approach for managing the programme.

Aware of the difficulty within IT Governance implementation programmes, with this research we aim only to address a few challenges identified in [8]. The challenges selected are mostly related with stakeholder engagement, scope and requirements alignment, environment awareness and the development

---

of appropriate solutions.

Therefore, the objectives for this research are the following:

- **Objective 1**: Increase senior management commitment and support during the whole programme;
- **Objective 2**: Detect misalignments earlier in the development;
- **Objective 3**: Decrease the resistance to change;
3

Theoretical Backgrounds and Related Works

Contents

3.1 Theoretical Background .................................................. 12
3.2 Requirement analysis and project planning ............................. 18
3.3 Scaling agile methods ...................................................... 21
3.4 Agility in Non-software Development Areas ........................... 26
In this chapter, we introduce a literature review on Scrum-based approaches that consider an initial phase of requirement analysis, followed by a scaled agile framework analysis for addressing programmes. We conclude with an overview of the scrum in non-software development areas theme, which is still a controversial subject.

3.1 Theoretical Background

In order to understand the problem context, in this section we explain the concept of Governance of Enterprise IT, followed by an introduction on COBIT 5 and how it is used to implement GEIT in organizations. Project management methodologies are also discussed with special focus in the agile paradigm and the main differences from the old traditional one.

3.1.1 Governance of Enterprise IT

There is a certain ambiguity concerning governance definition. For future reference, the definition considered will be the one provided by ISACA [11]: “Governance ensures that stakeholder needs, conditions and options are evaluated to determine balanced, agreed-on enterprise objectives to be achieved; setting direction through prioritization and decision making; and monitoring performance and compliance against agreed-on direction and objectives”.

The main goal of corporate governance is to produce value, which means releasing benefits, while managing risk and optimizing resources the best way possible [11]. IT was considered for many years just a support area of the main business, not being, for that reason, a corporate governance concern. Nowadays, IT is recognized as being not only a powerful resource to achieve enterprise objectives, but also a business enabler since it is pervasive bringing myriad benefits, such as lower costs, increased performance and efficiency, risk control and electiveness [12,24,25].

Regarding its widely significance, IT should be managed in a strategic way. This is the main concern of IT governance, which is defined as being “the leadership and organizational structures and processes that ensure that the organisation’s IT sustains and extends the organization’s strategy and objectives” according to Steven De Haes et al. [10].

Thus, as a core element, IT must be managed as an integral part of corporate governance, and not as a separate discipline. This is the reason why the term IT governance is not used anymore. Nowadays the used term is GEIT.

As aforementioned, being IT more and more used as service provider for most of the main processes in organizations, it is very important to have a managed use of IT and control over IT associated risk. This dependency calls for specific focus on GEIT, and frameworks are the best tools for a guided and
controlled implementation, aligned with the organization specific strategy, as they provide a guideline when implementing GEIT [10].

3.1.2 Using COBIT 5 for GEIT Implementation

COBIT was first created in 1996 by ISACA members for helping with a guided execution of IT audit assignments. As the framework was growing, it transitioned from being an IT audit framework to a good-practice framework for organizations. First versions were built around the concept of control objectives as policies, procedures, practices and organizational structures to ensure the enterprise goals achievement. In the last version - fifth - of COBIT (2012) [11] "control objective" concept was replaced by "good practices". Nowadays, COBIT 5 is widely recognized as being one of the most used frameworks for IT Governance [13,21].

ISACA developed COBIT 5 as a framework to "provide guidance in evaluating, directing and monitoring an enterprises use of IT" [8], giving an objective way for companies to align business strategy with IT goals. For this purpose, the framework relies on a set of five principles, namely [11]:

1. Meeting Stakeholder Needs
2. Covering the Enterprise End-to-End
3. Applying a Single Integrated Framework
4. Enabling a Holistic Approach
5. Separating Governance from Management

Principle 1 advocates that organization’s main goal is to create value for their stakeholders. To support this notion, COBIT 5 provides guidance on how to implement the required enablers to support business value creation through use of IT, which invokes the notion of strategic alignment [14]. To achieve this alignment, COBIT 5 provides a tool, called “Goals Cascade”, that enables the identification of key enabler goals based on identified stakeholder needs. Goals cascade, supported by a set of mapping tables, transforms stakeholder needs into enterprise goals, which will cascade into IT-related goals and finally enabler goals [11].

Principle 2 states that COBIT 5 covers all the enterprise’s functions, defining both IT processes and IT-related processes, including the required business and IT roles and their responsibilities.

The third principle says that COBIT 5 is a single and integrated framework, since it is aligned with other relevant standards and frameworks.

Principle 4, one of the most important, explains that an effective and efficient implementation of GEIT implementation must take into account the interdependence between several components (enablers)
such as processes, organizational structures, information, culture, principles, services and people. As De Haes et al. said [14]: “Such organizational system requires the definition and application of structures (e.g. organizational units and functions) and processes (to ensure tasks are coordinated and integrated), and attention to people and relational aspects (e.g. culture, values, joint beliefs, etc.).” Therefore, an “Enabler” is defined as a factor that “individually and collectively, influence whether something will work - in this case, governance and management over enterprise IT” [14].

Finally, the last principle aims to distinguish governance from management, defining separate roles and responsibilities.

The implementation guide [8] was developed along with COBIT 5 in 2012 due to the numerous challenges found by organizations when trying to adopt previous versions of COBIT [14]. The approach is based on a seven-stage life-cycle (Fig. 3.1), with three interrelated components: the core GEIT life-cycle, the enablement change (which addresses the behavioral and cultural aspects) and the programme management. This is a sequential and ordered cycle of continuous improvement based on the implementation of enabler goals according to the enterprise’s specific needs.

In the last phase, new objectives will be identified and a new cycle of improvement begin. Table 3.1 shows detailed information on the activities encompassed in each phase.

![Figure 3.1: COBIT 5 life-cycle](image-url)
Table 3.1: Described phases for GEIT Implementation with COBIT 5 (from [8])

<table>
<thead>
<tr>
<th>Phase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>What are the drivers?</td>
</tr>
<tr>
<td>Phase 1 is for identifying the current change drivers, while it is established the desire to change at executive management levels. Regarding programme management dimension, it is identified the risk associated, the objectives, success factors and high-level responsibilities, which will be described in the business case.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Where are we now?</td>
</tr>
<tr>
<td>Phase 2 is for aligning the IT-related goals with business strategies and risk. For this purpose, COBIT 5 provides a generic mapping (goals cascade) of enterprise goals to IT-related goals to enabler goals and to the key processes involved. Then it is assessed the current level of capability of the identified processes. Critical processes must be of sufficient capability to ensure successful outcomes. At programme management level, it is reviewed and evaluated the business case.</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Where do you want to be?</td>
</tr>
<tr>
<td>Phase 3 is for setting the target for improvement, followed by a gap analysis between AS-IS and TO-BE to identify solutions that must be communicated to stakeholders.</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>What needs to be done?</td>
</tr>
<tr>
<td>Phase 4 is for plan feasible and practical solutions by organizing potential projects into the overall programme supported by justifiable business cases. At the change enablement level, it must be identified the quick wins build on existing strengths.</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>How do we get there?</td>
</tr>
<tr>
<td>Phase 5 is for implementing the proposed solution into day-to-day practices, monitoring the projects within the programme. Thus, it is required engagement and communication with stakeholders for the programme success.</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Did we get there?</td>
</tr>
<tr>
<td>Phase 6 is for assuring a sustainable transition of the improved governance and management practices into normal business operations. Monitor the achievement of improvements using performance metrics, for example monthly, and documenting lessons learned.</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>How do we keep the momentum going?</td>
</tr>
<tr>
<td>Phase 7 is for preparing for a new cycle of continuous improvement, by review the overall success of the initiative and identify further requirements.</td>
<td></td>
</tr>
</tbody>
</table>

3.1.3 Project Management Methodologies

A project management methodology is defined by Project Management Institute as a "set of methods, techniques, procedures, templates and best practices used on a project".

As aforementioned, COBIT 5 is a well-known framework that when successfully used, can bring much value to organizations. However, to use its principles and adopt its practices very often means not only to understand deeply the organization and its processes but also to redesign and implement new structures (and the other enablers).

In order to minimize some of the risks of implementing GEIT using such a complex framework, a strategy for managing the project that undertakes the adoption of COBIT 5 must be adopted so it can bring consistency, flexibility and security to project teams, increasing the probability of project’s success.

3.1.3.A Traditional Project Management: Waterfall

In the history of project management, the traditional paradigm is the oldest. Traditional approaches are known for being very strict and disciplined methods, where the mindset is to follow the plan as it is, based on the idea that projects are predictable and linear so changes are not expected.

The process is quite simple, the approach comprises a few phases of project development to be
followed in a sequential order without any overlap \cite{17, 27}. Thus, each phase is only done once, as soon as the phase is completed we must go to the next one, like a waterfall scheme. Phases vary somewhat according the scope of the project, however they all have the basic skeleton of initiate, plan, execute, control and close.

As the first programming languages appeared, new strategies were established around the management of software systems development, specially around their life-cycle as a project. These approaches are called Software Development Life Cycle (SDLC). The first and mostly known approach was officially proposed by Winston Royce in 1970 \cite{28}.

The waterfall approach, as a traditional methodology, is sequentially ordered and follows the principles described before, however with some additional phases specific to software development projects such as requirement analyses, design, implementation, verification and maintenance \cite{29}.

When Royce developed the SDLC waterfall methodology, he stated the obvious: the only time the code is going to be tested is at the end of the waterfall process and most certainly the code is not going to be perfect at the first iteration, in that case we must go back almost at the beginning of the process.

He added a new variable to the equation: the possibility of change: “Yet if these phenomena fail to satisfy the various external constraints, then invariably a major redesign is required. Either the requirements must be modified, or a substantial change in the design is required” \cite{28}.

3.1.3.B Agile Project Management: Scrum

At the end of XX century, as new technologies appeared, business processes became more complex, the market became more competitive and companies were running for better products, faster, with cost reduction in the process. This new competitive world was calling for innovation in methods for project development. “The old sequential approach won’t get the job done” stated Takeuchi and Nonaka in 1986 \cite{30}.

As in every situation, from a necessity a solution was born, and agile project management approaches appeared intimately connected with the field of software development. A traditional strategy was not well suited for software development, since we have to face an environment where the requirements are elusive, so we can not follow a strict plan without returning almost to the beginning in every test not succeed, as it was discussed before \cite{28}.

All these objections to the traditional approach hold the development of a new paradigm in project management area. These new methodologies were based on the agility concept of creating value under constant changing environments during project lifetime, being adaptability the key characteristic. Agile methodologies also emphasize the need for communication and collaboration within project team members and with stakeholders in order to provide transparency \cite{18, 27}, in opposition to traditional methodologies.
We can also see the change in paradigm when we inspect the Iron Triangle (or Project Management Triangle) (Fig. 3.2). This model is used to describe projects using success criteria such as cost, time and scope [31].

Figure 3.2: Project Management Triangle

In waterfall approaches the scope is fixed. Since it is plan-oriented it follows the initially conceived process, thereby budget and time are adjusted according the plan. Contrariwise, in agile approaches, the triangle is turned and the scope is the only thing that may vary during the project lifetime. Agile approaches are established on the value of responding to change over following a plan, which enables a fixed time and budget and no restrictions in scope [31].

Although it was initially created as a solution for software development, agile principles can be applied in other disciplines and it is starting to be adapted progressively more in areas beyond software [19,32].

Scrum was developed in the early 1990s by Ken Shwaber and Jeff Sutherland as an agile iterative and incremental approach for software development. The mindset is about building the product incrementally using several short iterations (sprints) with feedback loops of the work done (Fig. 3.3), instead of deliver a complete product near the end [17, 18, 27, 33]. Its mindset is the empiricism [33], which makes it simple and straightforward, since it is based on the idea that knowledge comes from experience “simplicity is essential” [18].

When applying Scrum, we must adopt some specific rules derived from its values of transparency, adaptation and inspection [33]:

- There are three main roles: Product Owner, Scrum Master and Development Team. Together, they form the Scrum Team.

- The project is divided into sprints – time boxes between 1 to 4 weeks. At the end of each sprint it must be delivered some valuable product increment.

- Inspection and adaptation are assured through formal events: at the beginning of each sprint (Sprint Planning), at the end of each day (Daily Meeting) and at the end of each Sprint (Sprint
3.2 Requirement analysis and project planning

The current life-cycle of COBIT 5 adoption proposed by ISACA suggests a linear strategy, as explained in section 3.1.2. This choice is sustained on the fact that there are a few steps that depend on a previous analysis of the current environment to be performed. For instance, we have a life-cycle of 7 phases (Fig. 3.1), but only in phase 5 we start to develop and implement the solution, as we have to know the drivers of change (phase 1), identify the enabler’s goals and assess the AS-IS state (phase 2) to establish a target for improvement (phase 3) and build the projects (phase 4) that will implement the solutions (phase 5) [8].

From this perspective, adopting COBIT 5 is different from a software development project since it requires a strong initial analysis of the organizational environment regarding culture, principles, processes, organizational structures, information, applications and people (seven enablers [11]).

There is a lot of work that must be addressed before and after a project that will affect the way management and governance works at an organization, and a COBIT 5 adoption is the kind of programme that must not begin unless we know what we want, so we can establish a first initial target.

Traditional life-cycle approaches are known for having an exhaustive planning phase, so the rest of the cycle has the minimum changes possible, while agile strategies execute short full-cycles (iterations) with feedback loops and the planning is done along the process [27] (see Fig. 3.4).

As explained before, this was one of the reasons why a linear strategy was chosen [8]. However, we found some studies that prove that is possible to use Scrum and to have that initial requirement analysis with and hybrid approach, and that is what in fact people do when they say they use Scrum, as we will following explain.
3.2.1 Water-Scrum-Fall

Actually, almost every project has this need for prior requirement analysis. This is basically the idea that supports the theory that majority of Scrum projects are indeed hybrid traditional-scrum approaches [34].

Water-Scrum-Fall is presented as “the reality” of Scrum [34], and states that although organizations say they use agile approaches, continue to follow traditional processes at the beginning and end of the project, leaving Scrum practices to product development [34], which turns out to be an hybrid approach with three distinct phases: Water, Scrum and Fall (Fig. 3.5). Sometimes, to combine waterfall and Scrum techniques is an intermediate transition during the migration to an agile methodology. However, regularly, is a deliberated decision to combine the advantages of both approaches.

Within the first step of Water-Scrum-Fall (Water), the high-level requirements needed to set the project direction, timeline and budget are defined. Following is the implementation in Scrum: “The Waterfall model typically divides the development tasks into multiple work packages and organizes them with milestones. Assuming that requirements are likely to change throughout the project, Scrum instead advocates a planning from iteration to iteration and aims to cut “through complexity to focus on building software that meets business needs”. [6].

Although some initial requirements were established, they are not definitive, the main goal of this approach is to define an initial idea of the project that only means to set direction, requirements may change within the next sprints. Finally, the last phase (Fall) of the project is a rollout step to test the product, this formal step is to determine the acceptance of the product as a whole [3, 34].
3.2.2 Business Process Modeling with Scrum

Regarding this subject, other solutions were considered particularly relevant, namely C. Thiemich and F. Puhlmann study [4] of Business Process Modeling (BPM) in which they propose to improve each process incrementally using Scrum, with process releases each 5 to 10 sprints.

For defining the project direction and evaluate BPM maturity, before the project starts there is a dedicated scoping phase. Additionally, at the beginning of each release there is a sprint 0 for defining the initial requirements, where Process Backlog is an output expected at the end. The next sprints are dedicated to the process increment (same as product increment).

Considering project development management, this study provides some relevant findings following described:

• To make several iterations to the implementation, being each one of them a process improvement of the same process.

• Division of work in two different time-boxes. There is the notion of sprint (same as the one provided in [26]), and there is the process release that comprises 5 to 10 sprints, having as final result a new and different version of the process.

• The addition of a backlog grooming activity before each new process release begins. This activity is used in many scrum projects and allows the declaration of a specific activity for product backlog preparation, which is necessary for dealing with the complexity of redefining a process (is done in sprint 0).

• Each sprint still requires the development of a product increment to be evaluated. The product increment is considered a process increment. Therefore, a process release is developed incrementally by small process increments each sprint.
The concept of process releases may be an interesting addition, however, when considering COBIT 5 one must not forget that is not just about process design and implementation. There are other enablers that must be contemplated in a GEIT implementation such as information, culture or even roles. Although the enablers must be implemented under specifically identified processes, not all cases of process maturity improvement will be about designing a new process and implement new procedures; often means, for example, to help them designing a given output or adding a new role.

Looking at these two solutions for scoping and requirements analysis, both scope and kick-off phases of [4] are traditional steps, which makes these two solutions very similar at the Water level of the Water-Scrum-Fall approach (see Fig. 3.6).

![Figure 3.6: BPM with Scrum (taken from [4])](image)

### 3.3 Scaling agile methods

One of the challenges of developing a Scrum-based approach for COBIT 5 is the scalable factor. The original Scrum framework [33] was designed for a single project, however COBIT 5 is approached as a programme [8], which is defined as being "a group of projects managed in a coordinated way to obtain benefits not available from managing them individually" [35].

The major challenges to address in a complex project or a programme are mainly related with scaling the coordination mechanisms between teams [36, 37]. Therefore some factors must be considered and practices must be encountered to:

- Cross-team coordination and collaboration;
• Scale requirements management;

• Coordinate the dependencies between projects;

Several frameworks were developed such as Scaled Agile Framework (SAFe), Large Scale Scrum (LeSS), Nexus, and others, in order to address the stated problems resulting of having to work with multiple agile teams. All these frameworks combine a set of different scaling methods to deal with the full life-cycle of the product [38]. Following we will briefly discuss each framework main scaling agile practices.

3.3.1 SAFe

There are four versions of SAFe², considering the development environment: Essential, Portfolio, Large Solution and Full. We will only consider the simplest version, for this is the basic structure of this framework and the foundation of the other configurations.

The Essential SAFe configuration (Fig. 3.7) considers only two levels - programme level and team level. A programme is defined as a group of agile teams that are dedicated to the programme and capable of supplying business capability or value.

• **Life-Cycle**: Programme level has its own iterations that are called Programme Increments. Programme releases composed by four iterations and an additional one to plan the next releases.

• **Events**: The strategy used is to scale all agile events to the programme level as well. So we have Iteration Planning, Review and Retrospective at Team level, and then Programme Increment Planning, Inspect and Adapt (a combination of both Review and Retrospective). Other events were added such as Scrum of Scrums and Product Owner Sync to keep the integrity between teams.

• **Artifacts**: The same strategy repeats also for the artifacts, where we have a Programme Backlog at Programme level and two Backlogs at Team level: Team Backlog (part of the Programme Backlog assigned to a specific team) and Iteration Backlog (same as Sprint Backlog for Scrum). Additionally, we have also Iteration Goals, Programme Iteration Objectives, Team Programme Iteration Objectives and Stories.

• **Roles**: Scrum roles are the usual and operate at Team level: Product Owner, Scrum Master, Development Team. To scale to Programme Level new roles were added such as: Product Management, Release Train Engineer and System Engineer.

²https://www.scaledagileframework.com/
3.3.2 LeSS

LeSS\(^3\) is defined by minimalism and minimal process and for that use as little methods as possible to get multiple Scrum teams to work.

There are two versions of LeSS: LeSS and LeSS Huge. LeSS Huge is recommended for programmes where we have more than eight Scrum Teams working on the same Product Backlog.

- **Events:** The strategy is to have multi-team meetings. Events run at the same time for all teams. A single Sprint Planning and Sprint Review for all teams, for which Sprint Planning is divided into two parts, the first part is a team retrospective and the second one is a joint meeting with representatives from each team (see Fig. 3.8). Sprint Retrospective is performed by each team individually, and then an Overall Sprint Retrospective is conducted to discuss cross-team, organizational and systemic problems within the organization.

- **Artifacts:** Single Product Backlog and each team has its own Sprint Backlog.

- **Roles:** At its most basic version, LeSS recommends to have a single Product Owner for multiple teams, in LeSS Huge, we have a single Product Owner and several Area Product Owners, each one responsible for a group of teams.

3.3.3 Nexus

Nexus was developed by Ken Schwaber, one of the Scrum founders as a framework for scaling Scrum \(^6\) (see Fig. 3.9).

\(^3\)https://less.works/
Its main goal is to provide a structure of roles, events and artifacts that integrate the work of multiple scrum teams (3 to 9) working under the same Product Backlog in order to build a single and integrated product [8].

- **Events:** We have a Nexus Sprint Planning, Nexus Sprint Review and Nexus Sprint Retrospective. These are single meetings for all scrum team members and nexus integration team. At the end of each Nexus Sprint Planning, each team performs its own Sprint Planning. The same happens with Nexus Sprint Retrospective. Nexus Sprint Review replaces individual Sprint Reviews, because we only want to obtain feedback on the integrated product increment.

- **Artifacts:** There are three main artifacts: Product Backlog, Nexus Sprint Backlog and the usual Sprint Backlog. Product Backlog is a single Backlog for the entire Nexus and its Scrum Teams. Nexus Sprint Backlog is the aggregation of all individual Sprint Backlog’s, and it was designed to identify dependencies during the sprint.

- **Roles:** Each Scrum Team is composed by the known Scrum roles (Product Owner, Scrum Master, Development Team) and additionally there is a special team called Nexus Integration Team with a single Product Owner, Scrum Master and other needed members, whose responsibility is to ensure the development of an integrated increment at the end of each Sprint of Nexus.

### 3.3.4 Scaling agile frameworks’ analysis

As aforementioned, numerous frameworks have been designed to provide guidance and structure for scaling agile development. In this section, we compare the three main frameworks discussed before in order to understand the best practices to use in a COBIT 5 adoption.

Technical reports show that SAFe provides higher ROI and increases productivity [38]. This framework is highly prescriptive and detailed, which provides a strong sense of guidance for practitioners as...
well as enforces a method. A negative aspect, or not depending on the scope dept, is the fact that is a very broad, but still complete, framework. It does not only scale up some of the agile practices defined by Scrum, but also introduces new scaled methods and concepts.

According to Version One 2017 survey, the most used agile scaling methods are: SAFe (28%), Scrum/Scrum of Scrums (27%), internally created methods (13%), Lean Management (4%), Agile Portfolio Management (4%), Large Scale Scrum (3%), Disciplined Agile Delivery (1%), Recipes for Agile Governance in the Enterprise (RAGE) (1%) and then Nexus (1%) [39].

From the three analysed frameworks, SAFe is the most used, but is also the most extensive, which can be a problem to COBIT 5 adoptions.

LeSS's idea is to be in compliance with agile principles of focusing more on individuals and less in the process, aiming to lighten the process. In that aspect, LeSS is less strict when compared to SAFe.

Nexus and LeSS were both developed based on the idea of Scrum. Nexus adoption is basically using Scrum with additional Nexus practices.

A success factor analysis comparing scaling agile methods was made by Rossi et al. [37] to better understand the practices that work best in a given context.

Concerning the life-cycle, from the three, SAFe is the only that provides a structure of two levels (three in portfolio version) and it is considered the most complex agile method, while LeSS tries to be as minimalistic as possible [37]. SAFe also differs from the other two frameworks at this point on the fact that the programme has a different timeframe - Team level is divided into iterations and Programme level into Programme Increments.

Scaled meetings are similar for LeSS and Nexus, having common scrum meetings for all teams divided in two parts (one part common for every team and a second part for each team to perform their individual meeting). In SAFe we have separate meetings at Team level and then the scaled meetings at
programme level.

Scaled requirements management requires a group of scaling practices that are concerned about managing requirements. These problems in SAFe are addressed by a hierarchical structure of Product Owners - each team has a Team Backlog that is part of Programme Backlog managed by a Team Product Owner and then an Iteration Backlog for each iteration of work, and at Programme level we have a Programme Backlog managed by a Product Manager that manages the work of all Product Owners. In LeSS we have a single Product Owner with a single Product Backlog, except for LeSS Huge version that divides Product Backlog in several Requirement Areas managed by Area Product Owners (this version is similar to SAFe). Nexus has a single Product Backlog managed by Product Owner from Nexus Integration Team, and each team is managed by a Product Owner with Sprint Backlogs for each Sprint.

Overall, there are many scaling methods and frameworks for scaling agile development, but as the study [37] concludes, a successful scaling of agile does not need to follow a specific scheme, for there is no right solution: "Scaling agile within an organization does not need to follow a specific scheme, rather the process can be tailored to the needs while keeping the core values and principles of agile methodologies." [37].

3.4 Agility in Non-software Development Areas

This section introduces a overview of the most recent studies on Scrum in areas beyond software development and explains how agile is starting to expand to myriad other areas.

As it was said before, agile arose linked with new approaches for software development, however, its values can be applied to many non-software projects. In fact, the agile software development movement made massive progress, and it is starting to be adapted more and more in other disciplines [4,19,32,40].

Many are the authors that embraced this mindset and gave the agile movement a voice, trying to open everyone’s eyes to the necessary change. It is not just about software, agility is fundamental to face the current competitive context in any organization [41]. Many are the organizations that despite having steady products, deliver agile services such as Apple, Amazon, Spotify, Facebook and Google [19]. Harvard Business Review published an article from Jeff Sutherland and Hirotaka Takeuchi [32] that states “Now agile methodologies [...] are spreading across a broad range of industries and functions”.

There are many reasons why organizations in general avoid agile methods, and some of them are the fear of change and the immediate costs of rethinking an approach that is fomented in the organization for years without immediate value return. Nevertheless, a state of inertia will prone to fail in a changing world: “Though the challenges associated with implementing transformational change remain great in the short-term, the new reality is that refusing to adapt to environmental changes comes at the much
Agile is a mindset. Organizations are starting to understand the concept and we can slowly realize that those who do, can apply agility in everything from management, to manufacturing, telecommunication, transportation, consulting, coaching and so on. Steven Denning et al. [19] analyzed a field study made by Scrum Alliance in 2015 and concluded from the results that agile approaches are not just for small business, or simple software development projects, on the contrary, it was adopted successfully in large and old firms such as Microsoft and Erickson, in long and complex projects with reliability issues in areas beyond software development.
4
First Iteration

Contents

4.1 Description ................................................................. 29
4.2 Design and Development ............................................. 29
4.3 Demonstration ............................................................. 33
4.4 Evaluation ................................................................. 36
4.1 Description

DSRM, being an artifact-centered research method, has as main goal, to create and evaluate IT artifacts designed to solve a given research problem.

For the problem stated in section 2 and to achieve the previously mentioned objectives, we propose to design and develop a Scrum based methodology specification to address COBIT 5 programmes.

We chose Scrum due to its nature of welcoming changes, perform continuous delivery and obtain constant feedback [33], which we intend to verify if contributes to strengthen the relation with stakeholders, and provide a set of solutions that best enables IT governance, considering the constant changing environment most organizations live.

For that purpose, two artifacts were designed: a) a set of constructs to define the rules and practices in which the implementation will operate; b) a model that illustrates the life-cycle of implementation.

To obtain a result as accurate as possible, we intend to perform two iterations to the DSRM process described in section 1.1.

This chapter relates to the first iteration. We start by introducing the initial solution designed for the methodology, followed by its demonstration and results. Finally, we present the evaluation process to assess the artifacts and the results obtained from the demonstration.

4.2 Design and Development

This section describes the first approach of a Scrum based methodology specification to address COBIT 5 programmes.

4.2.1 Specification of Scrum concepts

The first artifacts developed were the constructs. As defined in [20], a construct provides the language in which the solution is defined, which will allow us to build the model.

The set of constructs are here represented in the form of three tables. In Tables 4.1, 4.2 and 4.3 the authors explain how this solution differs from the original definition in terms of events, artifacts and roles.

In section 3.3, a set of solutions for scaling scrum were analysed. Due to the complex nature of COBIT 5, we believe that the best strategy is to create the simplest approach possible. For that purpose, we selected only the indispensable elements for scaling scrum, using known scaled methods already analysed.

Starting by the scaled methods used for addressing hierarchy of roles and requirement management, the first question is what defines a project within this so called COBIT 5 programme. ISACA does not define boundaries, however we believe that the processes are the answer. We chose to define that each
project should respect a given COBIT 5 process, since these concepts are more like knowledge areas with sub-processes (Fig. 4.1). Whatever the enabler we want to implement, will be over those process areas, so they will define the area/scope of implementation of each project.

Figure 4.1: Example of a COBIT 5 practice (APO05) of the process APO12 Manage Risk (taken from [7])

Regarding teams, the rules will be the same for each project as for a normal Scrum Team with a sprint backlog and a scrum master, but the product owner will be unique for the programme. He must develop and maintain a single product backlog (see Fig. 4.2), similar to the LeSS1 basic version: “Multiple teams building a single product work from a single Product Backlog that defines all of the work to be done on the product. They do not each have their own Product Backlog. Product Backlog Items are not preassigned to the teams.”.

Figure 4.2: Scrum roles and artifacts

Respecting scrum meetings, to minimize the complexity, the authors chose to keep them just at project level, since product backlog is also unique, so no scaled methods for scaling events were added. Product Owner should maintain the integrity of the programme, and solve problems related with dependencies between the projects together with the respective scrum masters, considering possible Scrum

1https://less.works/
of Scrum meetings.

Table 4.1: Scrum artifacts for a COBIT 5 programme

<table>
<thead>
<tr>
<th>Scrum Artifacts</th>
<th>Original Definition</th>
<th>Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Backlog</td>
<td>Product Backlog is an ordered list of all the functionalities, functions, requirements, improvements and corrections that represents a change in the product. It is Product Owner's responsibility during the whole project, including its creation, manage its content, availability, and ordering its items.</td>
<td>Product Backlog is unique to the entire programme. It is an ordered list of user stories related to the agreed deliverables to implement governance enablers.</td>
</tr>
<tr>
<td>Sprint Backlog</td>
<td>Sprint Backlog is a subset of Product Backlog. It is the list of selected items from Product Backlog for a specific sprint. It is a result of the development team selection of user stories to work on in that iteration.</td>
<td>Sub set of items from Product Backlog, with the same purpose of a typical Sprint Backlog. Each team has its own Sprint Backlog since they have separate contexts to work on.</td>
</tr>
<tr>
<td>User Story</td>
<td>Representation of a requirement as an item of Product Backlog. Is a simple and easy way to create context around a task. User stories are short and should be something like: As a &lt;type of user &gt;, I want &lt;some feature &gt;, so that &lt;some goal &gt;.</td>
<td>Must be simple and short.</td>
</tr>
<tr>
<td>Product Increment</td>
<td>A usable part of work done in each sprint. Product Increment is the group of all items from the Product Backlog that were “done” during the last sprint. This increment will sum to the previous product increment if its complete enough to demonstrate its working functionality.</td>
<td>Processes are not as easy to deliver as features of software applications. The main goal will be to improve process capability with small deliveries of necessary work products to achieve the established target. A possible deliverable (product increment) may be a work product or any other necessary output, for example: architecture design, process documentation, workshops, etc.</td>
</tr>
</tbody>
</table>

Table 4.2: Scrum roles for a COBIT 5 programme

<table>
<thead>
<tr>
<th>Scrum Roles</th>
<th>Original Definition</th>
<th>Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Owner</td>
<td>This person is the client representative throughout project. He/she is responsible for creating and managing the content of the Product Backlog. His/her job is to manage the Product Backlog according to the clients’ interests, ordering the items to reflect the client’s priorities.</td>
<td>Because of the several dependencies encompassed in a COBIT 5 programme, it was established that the role of Product Owner is unique to programme and project levels. He must maintain programme integrity and monitor the dependencies between projects. In the scoping phase, he is the one responsible for identifying the business priorities and issues triggering need to act. He should keep the same responsibilities as the original product owner regarding product backlog.</td>
</tr>
<tr>
<td>Implementation Team</td>
<td>Also known as development team. Group from 3 to 9 professionals whose main responsibility is to turn Product Backlog’ items into tangible Product Increments. This team is self-organized and cross-functional, deciding autonomously which part of the Product Backlog is done in each Sprint.</td>
<td>Since COBIT 5 is a programme, the authors established that each project should correspond to a process identified in sprint 0. For each project, there must at least one implementation team and one scrum master. Implementation team is responsible for designing and implement the established governance solutions in the process (see product increment). They must also participate in the AS-IS capability assessment.</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>Scrum Master is responsible for promoting and supporting scrum among team and non-team members, as he/she must also coach the development team.</td>
<td>Same responsibilities as a typical Scrum Master.</td>
</tr>
</tbody>
</table>
| Scrum Team | Scrum team = Implementation Team + Product Owner + Scrum Master | Scrum Team = Implementation Team + Scrum Master. Product Owner is unique for the whole programme.
### Scrum events for a COBIT 5 programme

<table>
<thead>
<tr>
<th>Scrum Events</th>
<th>Original Definition</th>
<th>Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Backlog Refinement</td>
<td>Product Backlog Refinement is the activity in which the Product Owner and the Development Team collaborate for revising Product Backlog items, adding detail, estimating value, and reordering the items. This activity happens at the end of each sprint.</td>
<td>This activity is fundamental for a COBIT 5 programme given its numerous process dependencies. Same purpose as a typical backlog refinement.</td>
</tr>
<tr>
<td>Daily Scrum</td>
<td>Is a stand-up meeting with maximum duration of 15 minutes, every day in the same place at the same time. The goal is for every team member to expose any conflicts or issues with the remaining team members.</td>
<td>May not be applied. It is difficult in a complex programme where the resources are sometimes outsourced to reunite everyone in the same place at the same time every day.</td>
</tr>
<tr>
<td>Sprint Planning</td>
<td>This meeting is for planning the next sprint. Every development team member selects a few user stories from Product Backlog to implement in the next iteration and the result is the Sprint Backlog.</td>
<td>Only at project level. At the beginning of each sprint, with the same purpose as a typical sprint planning meeting.</td>
</tr>
<tr>
<td>Sprint Review</td>
<td>Meeting at the end of each Sprint, where the Scrum Team and relevant stakeholders inspect the increment and agree on changes in the Product Backlog. The main goal is to show the items “done” and get feedback on the increment, which may be accepted or rejected by the client.</td>
<td>Only at project level. At the beginning of each sprint, with the same purpose as a typical sprint review meeting.</td>
</tr>
<tr>
<td>Sprint Retrospective</td>
<td>Meeting at the end of the Sprint, where the team inspects itself and the work performed and creates a plan with improvements for them to apply in the next sprint. Occurs after each Sprint Review and before each Sprint Planning.</td>
<td>Only at project level. At the beginning of each sprint, with the same purpose as a typical sprint retrospective meeting.</td>
</tr>
</tbody>
</table>

#### 4.2.2 The life-cycle model

The second artifact is here represented in (Fig. 4.3). This model was built upon the concepts presented before (Tables 4.1, 4.2 and 4.3) and also based in the related work presented in section 3. The model illustrates the life-cycle of the methodology. According to the definition [35], project life-cycle consists in all the development phases of the process. We decided to use a model for representing the life-cycle, similar to other approaches of life-cycle representations. This is the first iteration of the artifact, so our intention is to refine the model, the constructs and the methodology itself.

The model designed (Fig. 4.3) identifies the seven implementation phases (Table 3.1) in red circles aiming to show how they match with each phase of this new approach.

The model was designed following the West’s et al. Water-Scrum-Fall approach [3, 34], previously analysed.

Phase 1 of a COBIT 5 programme (Table 3.1) is for identifying the drivers of change, therefore it was considered part of project initiation (scope planning for defining the product, deliverables and objectives), so it was clear for us that is prior work to be performed before the execution. Phase 2 is for selecting the key processes to improve, and we decided to adopt the practice from [4] of including this process in a special sprint for requirements definition, having product backlog as an output, to introduce sprints as
much as possible.

At the end of the Water part of this approach, the programme objectives were discussed, the initial requirements were established, the key processes were identified and the initial product backlog was created.

For the Scrum part of this hybrid solution, the work is divided into teams. Each Scrum Team becomes responsible for one process, starting by its capability assessment to the actual implementation of the governance solutions (phase 2 to phase 5 from COBIT 5 Table 3.1). These solutions will be designed iteratively so that the requirements initially defined can be reevaluated and priorities can be readjusted. With this approach, the authors expect to design better solutions getting feedback periodically.

After the solutions are implemented, they must be controlled. This corresponds to phase 6 (Table 3.1) and it is for controlling the processes and "test" the solutions, measuring the defined metrics. Thus, phase 6 and 7 from COBIT 5 are not addressed with Scrum, because it is a roll-out period of test, corresponding to the Fall step of this solution.

Phase 6 is to monitor the achievement of improvements using performance metrics and phase 7 is to enable the process of continuous improvement and identify further necessary improvements.

4.3 Demonstration

This section covers step 4 of the DSR method. To demonstrate the feasibility of this solution and to prove that it can be used to improve the outcomes of COBIT 5 programmes, we will next discuss the results from applying it on a real programme of GEIT implementation with COBIT 5 in the Portuguese Finance Ministry.
4.3.1 Context

The programme was carried out under the special mission Unidade de Implementação da Lei do Enquadramento Orçamental (UniLEO) of the Portuguese Finance Ministry. With this GEIT implementation programme, they sought to obtain reliable guidance from a well-known framework, such as COBIT 5, to start their work system based on internationally accepted good-practices. This programme had a duration of 5 months with 8 sprints of 2 weeks each, and a pause for Christmas holidays of 2 weeks.

The programme was composed by 5 projects, where each of them was focused on a specific process. The selected processes were the following:

- APO02 - Manage Strategy;
- APO03 - Manage Enterprise Architecture;
- APO06 - Manage Budget and Costs;
- APO10 - Manage Suppliers;
- BAI01 - Manage Programmes and Projects;

More information about the people involved in the programme can be found in Table 4.4.

<table>
<thead>
<tr>
<th>Role</th>
<th>External/Internal</th>
<th>Processes Involved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Owner</td>
<td>Internal</td>
<td></td>
</tr>
<tr>
<td>Programme Manager</td>
<td>External</td>
<td>BAI01</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>External</td>
<td>APO02 BAI01</td>
</tr>
<tr>
<td>Implementation Team Member 1</td>
<td>External</td>
<td>APO10 APO02</td>
</tr>
<tr>
<td>Implementation Team Member 2</td>
<td>External</td>
<td>APO03 APO06</td>
</tr>
<tr>
<td>Implementation Team Member 3</td>
<td>External</td>
<td>APO03</td>
</tr>
<tr>
<td>Implementation Team Member 4</td>
<td>External</td>
<td>APO03</td>
</tr>
<tr>
<td>Implementation Team Member 5</td>
<td>External</td>
<td>APO02 APO10</td>
</tr>
</tbody>
</table>

For each process was established a scrum team, however, since the resources were limited in number, some implementation team members participated in more than one team. Furthermore, Scrum master was the same for every scrum team and participated in one of the projects as an implementation team member as well, being more than just a team manager, which worked out very well.

The deliverables produced were the following: the business and application layers of the enterprise architecture, the TO-BE processes designed and documented, workshops of COBIT 5 and Scrum and documentation to support the establishment of some new organizational structures.
4.3.2 Results

To test the approach in a real environment, allowed the identification of some relevant issues.

First of all, the programme manager role is not part of the designed approach. It was agreed, during the scoping, that a programme manager would be essential for managing the upper level and keep the integrity of the product. Programme manager worked as a project manager for the programme, with scrum master responsibilities at the programme level.

Regarding scrum meetings, they were conducted at project level as established, however, since only 8 people were involved and scrum master was the same for all the projects, meetings were together, resulting in one common sprint planning, sprint review and sprint retrospective at the end of each sprint.

These meetings did not have the expected results. At the end of this initiative, when questioned, team members complained that these common meetings did not help solving their particular problems, stating that they were too broad, even though they recognized their importance for sharing information with the other teams.

Considering the programme life-cycle: phase 1 began before the involvement of the rest of the team, as established by the Water-Scrum-Fall approach. The planning process was responsibility of a joint work between product owner, programme manager and a COBIT 5 expert that is also member of the implementation team. Sprint 0 turned out to be a little mixed with the rest of the programme planning, and ended up to be just a big scoping.

Phases 2 and 3 were also addressed as prior work, since the processes chosen had very low capability levels (between 0 and 1), so the target for improvement was stated at the beginning as high-level targets. However, for the processes of capability level 1, were conducted a series of interviews with the client to perform a more accurate assessment.

In phase 4 and phase 5, were planned and produced the agreed deliverables, although the high-level requirements were agreed at scope definition, the solutions were constantly evolving during the closely contact with the customer, as foreseen by agile approaches.

Phase 6 was out of this project scope.

To test the approach allowed the authors to find a certain lack of coordination mechanisms to support the higher level of the programme, nevertheless, since there were not many people involved, this gap had not much impact on the results.

The use of an iterative approach for developing the deliverables, especially in phase 5 (Table 3.1), was crucial to find the best solutions, especially since the client was very uncertain about what he was looking for. The opportunity to participate actively in the development and to give feedback contributed to a much more accurate final product, as we will next analyse.
4.4 Evaluation

Evaluation phase (step 5 of DSR method) aims to measure how well an artifact supports a solution to the problem, comparing the objectives proposed (section 2.2) to the results observed from use of the artifacts in the demonstration (section 4.3).

For that purpose, we will evaluate the results considering a few criteria selected from the hierarchy of criteria proposed by Prat et al. [43].

The criteria that we found most relevant considering the objectives established are presented in Table 4.5.

Table 4.5: Criteria select for evaluating the artifact’s performance

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Criteria</th>
<th>Description</th>
<th>Form of Evaluation First Iteration</th>
<th>Form of Evaluation Second Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal</td>
<td>Efficacy</td>
<td>The degree to which an artifact produces its desired effect - Did the methodology facilitate the implementation?</td>
<td>Interviews</td>
<td>Interviews</td>
</tr>
<tr>
<td></td>
<td>Validity</td>
<td>The degree to which the artifact correctly achieves its goals.</td>
<td>Interviews with experts</td>
<td>Moody and Shanks</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Wand and Weber</td>
</tr>
<tr>
<td>Environment</td>
<td>Utility for People</td>
<td>Quality of the artifact in practical use for people - Was the use of an agile methodology helpful for the implementation?</td>
<td>Interviews</td>
<td>Interviews</td>
</tr>
<tr>
<td></td>
<td>Utility for Organizations</td>
<td>Quality of the artifact in practical use - Has the methodology used help the organization in a different way than a traditional process would have helped?</td>
<td>Interviews</td>
<td>Interviews</td>
</tr>
<tr>
<td></td>
<td>Ease of Use</td>
<td>How complicated the artifact is to use - The methodology was easy to follow?</td>
<td>Interviews</td>
<td>Interviews</td>
</tr>
<tr>
<td>Activity</td>
<td>Accuracy</td>
<td>Agreement between achieved and expected results.</td>
<td>Interviews</td>
<td>Interviews</td>
</tr>
</tbody>
</table>

The consistency of the artifact with each criteria will be assessed through the field study (the demonstration of the proposal instantiated on a real case). Additionally, to evaluate the correctness of the methodology, we performed a set of interviews with experts that evaluated each element comprised in the solution and gave their opinion as practitioners.

The sub-artifacts that form the main artifact (a model and a construct) were consequently validated through the evaluation of the methodology.

4.4.1 Interviews with experts

To evaluate the correctness of the methodology, 10 qualitative semi-structured interviews with expert practitioners in Scrum and/or COBIT 5. The interviews were conducted by one author over a period of one month and each one lasted about 1 hour. To support and lead the interview, it was developed a questionnaire with both open and closed response questions about COBIT 5, Scrum and the methodology as a whole.
It was also asked for interviewees to rank their expertise with COBIT 5 and Scrum frameworks on a scale from 1 to 5 (1 = marginal knowledge; to 5 = expert knowledge). This and other information about the interviewee’s can be seen in Table 4.6.

Table 4.6: Interviewee’s information

<table>
<thead>
<tr>
<th>Country</th>
<th>Experience in IT (years)</th>
<th>Function in IT</th>
<th>Level of Knowledge in COBIT 5 (1 to 5)</th>
<th>Level of knowledge in Scrum (1 to 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Portugal 15</td>
<td>Project Manager</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Portugal 18</td>
<td>Senior Advisor</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Portugal 26</td>
<td>PMO Director</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>Portugal 24</td>
<td>IT Governance Consultant</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>Portugal 19</td>
<td>IT Governance Consultant</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>Portugal 23</td>
<td>SI Consultant</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>Portugal 9</td>
<td>IT Governance Professor</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>Portugal 5</td>
<td>IT Governance Investigator</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>Portugal 23</td>
<td>Agile Coacher</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>Brazil 23</td>
<td>IT Governance Professor</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

With the questionnaire, it was provided the definition of each element for this specific solution as showed in Tables 4.1, 4.2 and 4.3, as well as the model of the life-cycle (Fi. 4.3). It was asked for them to provide their opinion, in open responses, about the approach based on their knowledge and experience in the matter.

In Table 4.7, it was adopted the following symbology for assessing the opinion gave to each element of the solution: if the interviewee did not agree with the definition gave to a specific element (Tables 4.1, 4.2 and 4.3), the cell is empty; if the interviewee partially agreed, the cell is filled with a semi circle; if the interviewee totally agreed, it was used a full circle.

Table 4.7: Results from the interviews

<table>
<thead>
<tr>
<th>Roles</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Owner</td>
<td>D</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>7</td>
</tr>
<tr>
<td>Implementation Team</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>10</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>9.5</td>
</tr>
<tr>
<td>Product Backlog Refinement</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>9.5</td>
</tr>
<tr>
<td>Daily Scrum</td>
<td>D</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>6.5</td>
</tr>
<tr>
<td>Sprint Planning</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>10</td>
</tr>
<tr>
<td>Sprint Review</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>10</td>
</tr>
<tr>
<td>Sprint Retrospective</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>10</td>
</tr>
<tr>
<td>Product Backlog</td>
<td>D</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>8.5</td>
</tr>
<tr>
<td>Sprint Backlog</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>8.5</td>
</tr>
<tr>
<td>Release</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>9.5</td>
</tr>
<tr>
<td>Product Increment</td>
<td>D</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>8.5</td>
</tr>
<tr>
<td>Scoping</td>
<td>D</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>8.5</td>
</tr>
<tr>
<td>Sprint 0</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>6.5</td>
</tr>
<tr>
<td>Phases 2-5</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>7.5</td>
</tr>
<tr>
<td>Phase 6</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>●</td>
<td>10</td>
</tr>
</tbody>
</table>

For a more quantitative analysis, the authors considered that each full circle has value 1, and each semi circle has value 0.5. For a total greater than 9 (with a maximum of 10), the cell was painted grey to
distinguish the more accepted elements from the rest.

Table 4.8 shows some of the statements collected from the interviews that justify the assessment gave to each evaluated element.

### Table 4.8: Quotes from the interviews

<table>
<thead>
<tr>
<th>Rules</th>
<th>Total</th>
<th>Quotes</th>
</tr>
</thead>
</table>
| Product Owner          | 7     | 11: “You will find several product owners, instead of just one. You will have a product owner for each process, to give you information about that specific process. However, there should be someone else, and you can still call it product owner, to keep the integrity of the product, and that you have it right.”
|                        |       | 110: “First of all, product owner should be the owner of each process. Second, the identification of the key processes through goals cascade should not be a task for product owner to do, is part of scoping and should be done without the involvement of any of the scrum roles.” |
| Implementation Team    | 10    |                                                                        |
| Scrum Master           | 9.5   |                                                                        |

<table>
<thead>
<tr>
<th>Events</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Backlog Refinement</td>
<td>9.5</td>
</tr>
<tr>
<td>Daily Scrum</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td>11: &quot;Usually, the team is not fully allocated in the client, for those projects that involve process improvements in an organization. Daily scrum is very important, but for a 100% allocated team, when you have 30% of allocation perhaps you should consider a weekly scrum, with a chance of losing information.”</td>
</tr>
<tr>
<td></td>
<td>14: &quot;These kinds of projects have a lot of daily work at the field, and to meet daily can be detrimental. I would not call it daily, but weekly for a two weeks 'spurt'”</td>
</tr>
<tr>
<td></td>
<td>17: &quot;To be daily together at the same place for a 15 minutes meeting is easy for a software development team that works every day in the same place. In a COBIT 5 project is very difficult to keep that event, however there are other ways of doing it, for example you can still talk every day without being together using Skype, for example”</td>
</tr>
<tr>
<td>Sprint Planning</td>
<td>10</td>
</tr>
<tr>
<td>Sprint Review</td>
<td>10</td>
</tr>
<tr>
<td>Sprint Retrospective</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Artifacts</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Backlog</td>
<td>8.5</td>
</tr>
<tr>
<td></td>
<td>11: “If you put everything in your product backlog you have to create a mechanism that allows you to distinguish between the different categories of user stories, the related with processes from the others related with tool, indicators, metrics, etc.”</td>
</tr>
<tr>
<td>User Story</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>10: “I do believe that user stories must be a mandatory thing in COBIT 5 projects, the representation of items should be simpler.”</td>
</tr>
<tr>
<td>Product Increment</td>
<td>8.5</td>
</tr>
<tr>
<td></td>
<td>16: “COBIT 5 is not just about processes, there are other solutions that you should contemplate in this definition.”</td>
</tr>
<tr>
<td>Scoping</td>
<td>8.5</td>
</tr>
<tr>
<td></td>
<td>11: “Sprint 0 is not relevant, you can just call it scooping because the only objective of this prior phase is to construct the product backlog.”</td>
</tr>
<tr>
<td>Sprint 0</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td>12: “The projects that will constitute the programme are only arranged in phase 4 of COBIT 5. So, scrum should be limited to phases 5 and 6 only. And I also believe scooping should extend to phase 3 included.”</td>
</tr>
<tr>
<td>Phases 2-5</td>
<td>7.5</td>
</tr>
<tr>
<td>Phase 6</td>
<td>10</td>
</tr>
</tbody>
</table>

From Table 4.7, we can perceive that are some elements that were not fully accepted by the experts (less than 9).

Some of the interviewees that considered themselves experts (level 5) in Scrum did not agreed with the responsibilities gave to the **Product Owner**. Some criticized the decision of making product owner a single role, as they believe that each process/project should have its own Product Owner/Process Owner, and a similar role should be included to maintain the integrity of the product as a whole, at programme level.

Regarding **Daily Scrum**, the opinions were rather divergent. Some believe that it should be weekly,
others think that is a fundamental part of Scrum and it should be mandatory.

For **Product Backlog**, the main problem was the definition of product that should not be limited to processes and this should reflect also product's backlog definition.

Concerning **User Stories**, mostly all Scrum experts considered that is quite indispensable for breaking resistance, but for COBIT 5 experts it was clear that it should not be used in the format suggested, should be simpler and not limited to a user interaction.

**Product Increment** definition was also criticized due to the definition of product. Product Increment should not be just small pieces of processes (process improvements) because should consider the other enablers as well.

Regarding the life-cycle, some interviewees have the opinion that **Scoping** should be extended to phase 3, eliminating **Sprint 0** for considering that this additional Sprint is rather irrelevant.

### 4.4.2 Evaluation of the demonstration

For evaluating the solution regarding the objectives established in section 2.2, the authors conducted a set of interviews with the team involved in the demonstration programme described in section 4.3, to evaluate the artifact's performance during the demonstration.

For this evaluation, were performed 6 semi-structured interviews to the implementation team members, scrum master and programme manager, conducted by one author over a period of one week and each one lasted about half an hour.

Table 4.9 shows the results of close response questions from the interviews, where it was asked for them to classify a few statements from 1 to 5 (1 = strongly disagree; 5 = strongly agree).

In the next paragraphs, we analyze the artifact according its performance during the field study considering the already introduced criteria. Furthermore, we show some statements collected from the open response questions to prove the statements given, where ITM corresponds to an Implementation Team Member.

**Efficacy**

All members interviewed provided a positive feedback concerning the capacity of the methodology to increase senior management commitment (as stated in Table 4.9), where 10 out of 10 claimed that business executives involved were always interested in setting new meetings to discuss strategies and participate in the solution with every information necessary, and part of the reason was the close contact provided by Scrum meetings.

**Utility for people**
It was noticed by everyone involved in the implementation that when asked for the executive members for their participation, they all agreed and facilitated several meetings to improve some of the solutions.

For example, when it came to design the Enterprise Architecture (EA) model, developing it iteratively with feedback loops foreseen by an agile method was very useful especially for the implementation team. The client participation was especially noticed in the EA development which, for being a model and translating their reality, required more involvement.

The EA solution was developed iteratively with feedback earlier than the 2-week sprint planned. At the end, the project linked to the architecture process had a different time-line than the others. On one hand, it was verified that the sprints were not suitable for its development, but on the other hand gave us the possibility to perceive this fact faster and to adapt this circumstance, than if the whole model had been fully developed or the biggest part of it, when only than we would realize that it was not ideal.

ITM1: “During the several iterations with the client was possible to identify some initial aspects that after all were not what the client wanted.”

ITM2: “The delivered solution was slightly different from the one originally planned.”

ITM3: “The sprint review was essential in adapting the initial solution to the client’s vision and needs. The feedback gathered allowed to iterative refine the solution so it would become closer to what the client needed as a solution that properly addressed the client’s vision.”

Utility for organization

A scrum-based approach to manage this programme was considered a key factor for the client to better understand the solutions proposed, since they were closely involved in the development.

A joint work was crucial to find the solutions that better suited the organization environment and way of working. This joint work enabled better solutions for them and, therefore, decreased the resistance to change.

However, they were not always aware for the changes and available to collaborate. At first, some of them showed a certain lack of expectations on the value obtained from the programme. At the end, almost everyone in the team involved strongly agreed (5) that they acknowledge the benefits of such a GEIT programme.

ITM1: “At the beginning, there was a slight resistance to change that I consider normal when it changes the way people have worked all their professional life. However, as they realized the reasons and benefits that the recommended changes could bring to the organization, they were no longer so resistant to change and were much more available and interested in implementing and promoting such changes.”

ITM3: “At first, the client showed signs of resistance as showed some difficulty in understanding the benefits of the solution. Probably that happened because some had a more technical background and had problems in understanding the value of the approach used.”

40
Table 4.9: Results from close response questions

<table>
<thead>
<tr>
<th>Role</th>
<th>Classify senior management interest in the programme (1 to 5)</th>
<th>Classify senior management involvement on the solution development (1 to 5)</th>
<th>Classify the client acknowledgement of the benefits of this programme (1 to 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Programme Manager</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Implementation Team Member 1</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Implementation Team Member 2</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Implementation Team Member 3</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Implementation Team Member 4</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Implementation Team Member 5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Accuracy

Regarding the life-cycle, as already mentioned, some components were not properly tested such as Sprint 0 and phases 6 and 7 (Fall step, after the Scrum process).

Respecting the methodology practices, a small analysis between the expected outcomes and the results from their application was made:

**Meetings:** Scrum meetings were not performed as suggested by the proposal. The team involved was rather small, which made it difficult to conduct separate meetings for every process. It was more natural to perform a single multi-team meeting of Sprint Planning and Sprint Retrospective at the end of each Sprint, since everyone was involved in more than one process.

**Roles:** For external reasons, a program manager was included in the programme at UniLEO, and ended up being essential to keep the integrity between the teams. Thus, and as stated by the interviewees, a product owner must still exist, but to monitor the product as a whole, with process owners to control each process/project.

**Artifacts:** Product backlog was jointly developed by everyone, since Product Owner was neither Scrum or COBIT 5 expert. Sprint Backlog was developed and managed using Trello\(^2\) a free online tool that organizes projects into boards, where every project had an individual board. This division ended up being slightly confusing since everyone was involved in more than one process/project, and the main board turned out to be used for managing all Sprint Backlogs.

\(^2\)https://trello.com/
4.4.3 Conclusion

Despite all those limitations, and the fact that the solution was proved to be a simpler version of what was expected, developing these organizational changes in an iterative and incremental way was very effective overcoming some issues related with senior management commitment, resistance to change and misaligned solutions from the scope defined, as perceived by the interviews with the team involved in the demonstration, which provided satisfactory results regarding the objectives established.

Of the 6 interviewees, 100% agreed or strongly agreed that Senior Managers showed interest in the programme, acknowledging that an iterative strategy was fundamental to a greater support from top management when developing the solution (objective 1).

Also, when asked if the solution provided at the end was the one planned at the beginning, the answers were unanimous, all 6 recognized the solution was build according the client’s needs and confirmed it was adapted several times (objective 2).

Finally, although they acknowledge the client was not always convinced about the benefits, of the 6 interviewees, 4 strongly agreed and 2 agreed about the client showing a greater knowledge of the programme at the end (objective 3), which implies an improvement that interviewees related to the engagement created during the scrum mandatory meetings.

Although there is still much aspects to improve, this first iteration allowed the authors to verify that developing organizational solutions with agile methods can bring many benefits for the organization and for practitioners.
In this chapter, we will present the work performed during the second iteration of a solution, starting by explaining the changes in design and development of the methodology, followed by its demonstration and results and finally describe the evaluation process and conclusions.

5.1 Design and Development

At the end of the evaluation step of the first iteration, we stated that the results obtained addressed the defined research objectives, however, since some aspects of the methodology were criticized during the interviews with experts and also perceived during the demonstration, we developed a new approach based on the results obtained from the first iteration.

After gathering insights on the first approach of a solution from experts, and analysing the results of its usage in a real environment, we decided to base our solution on SAFe\(^1\) and make a specification of concepts to a COBIT 5 adoption programme, similarly to the approach taken in the first iteration with Scrum.

The decision to use SAFe was sustained on the analysis performed in section 3.3.4 where it was derived the conclusion that its the most complete framework analysed and the one with the most prescriptive structure, which we expect to bring more consistency despite its complexity.

5.1.1 Specification of SAFe concepts

Following we will explain the new concepts adopted and how they can be specified to a COBIT 5 programme.

Tables 5.1, 5.2 and 5.3, show how we address COBIT 5 using some of the SAFe scaled practices. For the elements that are not applicable in a COBIT 5 programme, we marked them in red in every table.

Now, we have two levels of elements, the team level and the programme level, so each table is divided considering this distinction. First we introduce the scaled elements and then the single ones. The definitions were all taken from SAFe\(^2\), and were already discussed in section 3.

Figure 5.1 illustrates how the new scaled roles and scaled artifact from SAFe (see Table 5.1 and 5.3) will operate in this new proposal.

---

\(^1\)https://www.scaledagileframework.com/
\(^2\)https://www.scaledagileframework.com/
| Table 5.1: SAFe roles for a COBIT 5 programme |

<table>
<thead>
<tr>
<th>Scrum Roles</th>
<th>Original Definition</th>
<th>Scaled Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Architect/Engineer</td>
<td>Is an individual or small cross-discipline team that define the overall architecture for the system, help define Nonfunctional Requirements (NFRs), determine the major elements and sub-systems, and help design the interfaces and collaborations among them.</td>
<td>The programme level version of System Architect/Engineer. System Architect/Engineer has same responsibilities as RTE to ensure the flow of value, helps manage the interfaces and collaborations among the teams.</td>
</tr>
<tr>
<td>Product Management</td>
<td>Is the internal voice of the Customer and works with customers and Product Ownersto understand and communicate their needs, define system features, and participate in validation. They are responsible for the Program Backlog.</td>
<td>The programme level version of Product Manager. Product Manager is a client representative, a person and not a team, who has the same responsibilities of assuring the organization's interests in the programme, working with Team Product Owners, understand the stakeholder needs and accordingly define and manage the Programme Backlog.</td>
</tr>
<tr>
<td>Release Train Engineer (RTE)</td>
<td>Is a servant leader and the chief Scrum Master for the train. The RTE facilitates optimizing the flow of value through the program using various mechanisms.</td>
<td>The programme level version of Scrum Master. Programme Scrum Master has the same responsibilities as RTE to optimize the flow of value, helping the Product Manager coordinate the interdependencies between teams, facilitating events at programme level, clarifying the PI objectives of each team, managing their work at the programme level and facilitating communication between the several teams through their Scrum Masters.</td>
</tr>
<tr>
<td>Business Owners</td>
<td>Are a small group of stakeholders who have the business and technical responsibility for fitness for use, governance, and return on investment (ROI) for a Solution developed by an ART. They are primary stakeholders in the ART and actively participate in ART events.</td>
<td></td>
</tr>
<tr>
<td>Agile Team</td>
<td>A cross-functional ScrumXP or Kanban team which consists of the Dev Team as well as Scrum Master and Product Owner. This group of 5 to 11 people has the ability and authority to define, build, and test an element of solution value within an iteration.</td>
<td>A cross-functional scrum team which consists of the Implementation Team, Scrum Master and Product Owner. Since we will use Scrum for this proposal, we will call it Scrum Team. A group of professionals who will design and implement the necessary IT governance solutions to achieve the established enabler goals.</td>
</tr>
<tr>
<td>Development Team</td>
<td>Subset of Agile Team. It consists of the dedicated professionals who can develop and test a Story, Feature, or component. The Dev Team typically includes software developers and testers, engineers, and other dedicated specialists required to complete a vertical slice of functionality.</td>
<td>Development Team is for this programme an Implementation Team. A group of professionals who will design and implement the necessary IT governance solutions to achieve the established enabler goals.</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>Although the Scrum Master role is mainly based on standard Scrum, Agile Teams establish this position to help the team meet its goals and coordinate activities with other teams. The Scrum Master role is taken by a team member whose primary responsibility is assisting the self-organizing, self-managing team achieve its goals. Scrum Masters do this by teaching and coaching team practices, identifying and eliminating impediments, and facilitating flow.</td>
<td>Same definition.</td>
</tr>
<tr>
<td>Product Owner</td>
<td>The Product Owner (PO) is a member of the Agile Team responsible for defining Stories and prioritizing the Team Backlog to streamline the execution of program priorities while maintaining the conceptual and technical integrity of the Features or components for the team.</td>
<td>Same definition.</td>
</tr>
<tr>
<td>Scrum Events</td>
<td>Original Definition</td>
<td>Scaled Scrum for COBIT 5</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Agile Release Train</td>
<td>The Agile Release Train (ART) is a long-lived team of Agile teams, which, along with other stakeholders, develops and delivers solutions incrementally, using a series of fixed-length Iterations within a Program Increment (PI) timebox. The ART aligns teams to a common business and technology mission.</td>
<td>Iterative COBIT 5 programme initiative. The concept will not be adopted, since there this solution is for one programme only.</td>
</tr>
<tr>
<td>Programme Increment</td>
<td>A Programme Increment (PI) is a timebox during which an Agile Release Train (ART) delivers incremental value. PIs are typically 8– 12 weeks long. PI is to the programme level as an Iteration is to the team level, constituting a set of four consecutive Iterations, followed by one Innovation and Planning (IP) Iteration. All teams on the train are synchronized to the same PI length.</td>
<td>This is the programme level version of an Iteration. Programme Increment is a timebox that encompasses a predefined number of iterations. To enforce the jointly development of governance solutions, sprints should be more than 1 week and less than 1 month (4 weeks). Therefore, a correspondent Programme Increment should be more than 2 sprints and less than 4 since PI planning should not have an entire dedicated sprint.</td>
</tr>
<tr>
<td>Programme Increment Planning</td>
<td>Face-to-face event at the beginning of each Programme Increment. Includes a presentation of business context where the teams create their Iteration plans and objectives for the upcoming PI. Is facilitated by the Release Train Engineer (RTE) and includes all members of the ART. Takes place over two days and occurs within the Innovation and Planning (IP) Iteration.</td>
<td>This is the programme level version of Iteration planning. PI Planning should have the same duration as a Sprint Planning, facilitated by the Programme Scrum Master that includes all members of the programme and occurs at the beginning of each Programme Increment, with the same purpose as the original event.</td>
</tr>
<tr>
<td>Scrum of Scrums</td>
<td>The Release Train Engineer (RTE) typically facilitates a weekly (or more frequently, as needed) Scrum of Scrums (SoS) meeting. The RTE, Scrum Masters, and others (where appropriate) meet to review their progress toward milestones, program PI objectives, and internal dependencies among the teams.</td>
<td>Recommended scaled practice with the same purpose defined in SAFe, but not a mandatory event for a COBIT 5 initiative.</td>
</tr>
<tr>
<td>Product Owner Sync</td>
<td>Similarly to the SoS, a PO sync meeting is often held for POs and the Product Manager. Typically occurs weekly, or more frequently, as needed. The purpose is to get visibility into how well the ART is progressing toward meeting the program PI objectives, to discuss problems or opportunities and to assess any scope adjustments. May include Programme Backlog refinement.</td>
<td>Recommended scaled practice with the same purpose defined in SAFe, but not a mandatory event for a COBIT 5 initiative.</td>
</tr>
<tr>
<td>System Demo</td>
<td>Provides an integrated view of new features for the most recent iteration delivered by all the teams in the ART. Each demo provides ART stakeholders with an objective measure of progress during a PI.</td>
<td></td>
</tr>
<tr>
<td>Inspect and Adapt</td>
<td>The Inspect and Adapt (I&amp;A) occurs at the end of each Program Increment (PI) and before the PI Planning, where the current state of the Solution is demonstrated and evaluated. Teams then reflect and identify improvement backlog items via a structured, problem-solving workshop.</td>
<td>Programme level version for Sprint Retrospective consolidated with Sprint Review.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Team Level</th>
</tr>
</thead>
</table>

Continued on next page
<table>
<thead>
<tr>
<th>Scrum Events</th>
<th>Original Definition</th>
<th>Scaled Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agile Release Train</td>
<td>The Agile Release Train (ART) is a long-lived team of Agile teams, which, along with other stakeholders, develops and delivers solutions incrementally, using a series of fixed-length iterations within a Program Increment (PI) timebox. The ART aligns teams to a common business and technology mission.</td>
<td>Iterative COBIT 5 programme initiative. The concept will not be adopted, since there this solution is for one programme only.</td>
</tr>
<tr>
<td>Programme Increment Planning</td>
<td>A Programme Increment (PI) is a timebox during which an Agile Release Train (ART) delivers incremental value. PIs are typically 8–12 weeks long. PI is to the programme level as an Iteration is to the team level, constituting a set of four consecutive Iterations, followed by one Innovation and Planning (IP) Iteration. All teams on the train are synchronized to the same PI length.</td>
<td>This is the programme level version of an Iteration. Programme Increment is a timebox that encompasses a predefined number of iterations. To enforce the jointly development of governance solutions, sprints should be more than 1 week and less than 1 month (4 weeks). Therefore, a correspondent Programme Increment should be more than 2 sprints and less than 4 since PI planning should not have an entire dedicated sprint.</td>
</tr>
<tr>
<td>Scrum of Scrums</td>
<td>The Release Train Engineer (RTE) typically facilitates a weekly (or more frequently, as needed) Scrum of Scrums (SoS) meeting. The RTE, Scrum Masters, and others (where appropriate) meet to review their progress toward milestones, program PI objectives, and internal dependencies among the teams.</td>
<td>Recommended scaled practice with the same purpose defined in SAFe, but not a mandatory event for a COBIT 5 initiative.</td>
</tr>
<tr>
<td>Product Owner Sync</td>
<td>Similarly to the SoS, a PO sync meeting is often held for POs and the Product Manager. Typically occurs weekly, or more frequently, as needed. The purpose is to get visibility into how well the ART is progressing toward meeting the program PI objectives, to discuss problems or opportunities and to assess any scope adjustments. May include Programme Backlog refinement.</td>
<td>Recommended scaled practice with the same purpose defined in SAFe, but not a mandatory event for a COBIT 5 initiative.</td>
</tr>
<tr>
<td>System Demo</td>
<td>Provides an integrated view of new features for the most recent iteration delivered by all the teams in the ART. Each demo provides ART stakeholders with an objective measure of progress during a PI.</td>
<td></td>
</tr>
<tr>
<td>Inspect and Adapt</td>
<td>The Inspect and Adapt (I&amp;A) occurs at the end of each Program Increment (PI) and before the PI Planning, where the current state of the Solution is demonstrated and evaluated. Teams then reflect and identify improvement backlog items via a structured, problem-solving workshop.</td>
<td>Programme level version for Sprint Retrospective consolidated with Sprint Review.</td>
</tr>
<tr>
<td>Innovation and Planning Iteration</td>
<td>Provides the teams with an opportunity for exploration and innovation, dedicated time for planning, and learning through informal and formal channels. In the case where a release is on the PI boundary, teams perform final system verification, validation, and documentation.</td>
<td></td>
</tr>
<tr>
<td>Table 5.2 – continued from previous page</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Scrum Events</strong></td>
<td><strong>Original Definition</strong></td>
<td><strong>Scaled Scrum for COBIT 5</strong></td>
</tr>
<tr>
<td>-------------------</td>
<td>-------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td><strong>Iteration Planning</strong></td>
<td>At the beginning of each iteration. Is an event in which an Agile Team determines the Iteration Goals and how much of the team backlog they can commit to during an upcoming iteration. Prior to the meeting, the Product Owner (PO) will have prepared some preliminary iteration goals, based on the team’s progress in the Program Increment (PI). The Product Owner defines the what; the team defines how and how much. Attendees: Agile Team. Output: Iteration Backlog (and a statement of Iteration Goals).</td>
<td>Since we will use Scrum for this proposal, we will call it <strong>Sprint Planning</strong>.</td>
</tr>
<tr>
<td><strong>Daily stand-up</strong></td>
<td>Each day, the team has a formal ceremony — the Daily Stand-up (DSU) meeting — to understand where they are, escalate problems, and get help from other team members. During this meeting, each team member describes what they did yesterday to advance iteration goals, what they are going to work on today to achieve the iteration goals, and any blocks they are encountering in delivering iteration goals. As this is a daily coordination meeting, the Scrum Master has to keep it short and to the point. The DSU should take no more than 15 minutes and is done standing up in front of the storyboard. Attendees: Scrum Master and Development Team.</td>
<td>Same definition.</td>
</tr>
<tr>
<td><strong>Backlog Refinement</strong></td>
<td>Is an event held once or twice during the iteration to refine, review, and estimate stories in the team backlog.</td>
<td>Same definition.</td>
</tr>
<tr>
<td><strong>Iteration Review</strong></td>
<td>At the end of each iteration. During the iteration review, each Agile Team measures and then demonstrates its progress by showing working stories to the Product Owner (PO) and other stakeholders to get their feedback. Attendees: Agile Team and relevant stakeholders.</td>
<td>Since we will use Scrum for this proposal, we will call it <strong>Sprint Review</strong>.</td>
</tr>
<tr>
<td><strong>Iteration Retrospective</strong></td>
<td>After the Iteration Review. Is an event held at the end of the iteration for the Agile team to review its practices and identify ways to improve. The retrospective is based on the qualitative and quantitative information presented during the iteration review. Attendees: Agile Team.</td>
<td>Since we will use Scrum for this proposal, we will call it <strong>Sprint Retrospective</strong>.</td>
</tr>
</tbody>
</table>
Table 5.3: SAFe artifacts for a COBIT 5 programme

<table>
<thead>
<tr>
<th>Scrum Artifacts</th>
<th>Original Definition</th>
<th>Scaled Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Features</td>
<td>Solutions are developed using features. Each one reflects a service provided by the system that fulfills some important stakeholder need. They are maintained in the Programme Backlog and are sized to fit in a PI so that each delivers new value. Each includes a name, benefits hypothesis and acceptance criteria. They are sized to fit within a PI. Features typically provide functionality for multiple user roles. Ex: In-service software update, Route optimization, etc.</td>
<td>IT governance solutions intended to achieve the established enabler goals and fulfills some important stakeholder need. Regardless, they are treated as a typical feature of SAFe: they include benefit hypothesis and acceptance criteria, are stored in the programme backlog and sized to fit within in a PI. They are the items of a Programme Backlog.</td>
</tr>
<tr>
<td>Programme Backlog</td>
<td>This is the holding area for upcoming Features. Product Manager is responsible for it. The items in this backlog result from research activities and active collaboration with various stakeholders. Program Backlog is refined once or twice in a Programme Increment within a backlog refinement event.</td>
<td>Same definition. However, Programme Backlog must be jointly developed by everyone involved.</td>
</tr>
<tr>
<td>Programme Kanban</td>
<td>It manages the flow of features and enablers through the Continuous Delivery Pipeline.</td>
<td></td>
</tr>
<tr>
<td>PI Objectives</td>
<td>These are a summarized description of the specific business and technical goals that an ART intends to achieve in the next PI.</td>
<td>Same definition.</td>
</tr>
<tr>
<td>Architectural Runway</td>
<td>The runway consists of the existing code, components, and technical infrastructure necessary to support the implementation of prioritized, near-term features, without excessive redesign and delay.</td>
<td></td>
</tr>
<tr>
<td>Team Level</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Story</td>
<td>Stories are short descriptions of a small piece of desired functionality, written in the user’s language. Agile Teams implement small, vertical slices of system functionality and are sized so they can be completed in a single iteration. There are two types of stories: user stories and enabler stories. User stories are the primary means of expressing needed functionality and are written using user-voice format. As a (user role), I want (activity) to, so that (business value). Enabler stories are the ones related with things that are necessary to implement other user stories.</td>
<td>This kind of stories are closer to the enabler stories from SAFe. First because the story may not directly touch any end user, second because the user-centric format is not the most indicated language to describe the work necessary to implement IT governance enablers. The form of expression used must be simple and objective. Ex: “Design the AS-IS process so we can make a gap analysis.”</td>
</tr>
<tr>
<td>Team PI Objectives</td>
<td>Are a summarized description of the specific business and technical goals that an Agile team intends to achieve in the upcoming PI. Is an output of PI planning event.</td>
<td>To simplify, the objectives are merely statements.</td>
</tr>
<tr>
<td>Iteration Goals</td>
<td>Are an output of the iteration planning event. They are a high-level summary of the business and technical goals that the Agile team agrees to accomplish in an iteration. They help ensure alignment with the PI Objectives. Is an output of Iteration planning event.</td>
<td>To simplify, goals are merely statements.</td>
</tr>
<tr>
<td>Team Backlog</td>
<td>Team Backlog contains user and enabler Stories originate from the Program Backlog features. The Product Owner (PO) is responsible for the team backlog. Each PI Planning, the candidate features for the PI split into stories by the teams. Team Backlogs are refined in the Backlog refinement event.</td>
<td>A Backlog for a given agile team that oversees a specific COBIT 5 process – Process Backlog. Includes Program Backlog items for that specific Team to work with, in form of stories. In PI planning where new stories are added to the next PI. The backlog is developed jointly between Process Owner, Scrum Master of the process and the team.</td>
</tr>
<tr>
<td>Iteration Backlog</td>
<td>Selected stories from the team backlog to the iteration, with acceptance criteria where appropriate. Is an output of Iteration Planning events.</td>
<td>Since we will use Scrum for this proposal, we will call it Sprint Backlog.</td>
</tr>
</tbody>
</table>
Briefly, each Agile Team undertakes one process of COBIT 5, as decided in step 4. For this research we decided to use Scrum, so is equally valid to call it Scrum Team. In any case, each Team that will cover a specific process is composed by a Product Owner, a Scrum Master and an Implementation Team.

As explained in 5.3 and previously mentioned, there is an initial Product Backlog designed according the initial high-level requirements for the whole programme and is managed by Product Manager. In the first Programme Iteration Planning (see Fig. 5.2 and Table 5.2) each agile team select their tasks to the upcoming PI into a Process Backlog. For the iterations (sprints) comprehended in the Programme Iteration, there is a Sprint Backlog, developed at the beginning of each sprint at Sprint Planning event, as Scrum prescribes, managed by each Product Owner.

Figure 5.1: Scaled roles and artifacts for a COBIT 5 programme

Figure 5.2 illustrates how the new events (see Table 5.2) are coordinated with each other and where they occur within the life-cycle of the programme.

One of the differences of SAFe from other analysed scaled methodologies, is that distinguishes an iteration of the programme from an iteration of the team. An iteration in the programme is called Programme Increment and encompasses a set of consecutive sprints.
5.1.2 The life-cycle model

Following we describe how we developed the life-cycle (Fig. 5.3).

Although we proposed to have single meetings for each team for the first iteration, they ended up joining every one involved on the same meeting, as already stated. Therefore, we adopted the separation of levels imposed by SAFe (programme level and team level) trying to prevent programme level practices from mingling with team level practices.

Sprint 0 was eliminated, since it was not demonstrated correctly so we have no evidence of its
efficacy. Besides, from the interviews with experts we perceived that not all of them seemed to agree that a special sprint would make a difference.

Now, the scoping phase is fully considered a traditional process (Water) where agile methods are not included and comprise the first 4 steps defined by ISACA for a COBIT 5 adoption [8] from the first task of defining the issues triggering the need to act to the fourth of defining the projects for implementing the IT Governance solutions (see Table 3.1). Although the evaluation of the current state (step 2) is addressed as prior work, this evaluation is an high-level maturity assessment based on information gave from interviews with process owners. Once the processes/projects are established, the team will have to perform the assessment to confirm the current state as explained in [44].

As for the first approach, each project will correspond to one chosen process, so at the end of our scoping phase, the required projects are established and teams are allocated to begin the agile phase of this solution.

The programme level addresses the entire COBIT 5 programme, the team level begins when projects are established.

Once the scoping phase is finished, the high-level requirements are identified and an initial programme backlog can be developed.

Another interesting practice of SAFe is the fact that establishes that programme level and team level operate considering different timeboxes. We believe this can be positive addition for a COBIT 5 programme, since these kind of projects are longer than development projects and to perform status meetings at the same time for both levels can mix the things a bit, as we concluded from the first demonstration.

5.2 Demonstration

In the second iteration, after developing the new proposal, to assess it's effectiveness in addressing the research problem, we used the proposed methodology in a different project of GEIT implementation with COBIT 5, this time at the bank X.

5.2.1 Context

The programme started in July and had a duration of 3 and a half months, with 8 Sprints of 1 week each, and Programme Increments of 3 Sprints each.

The programme was composed by 4 projects, each one focused in a specific process. The processes selected as key-processes to improve in matters of IT governance and management were:

- APO04 - Manage Innovation;
More information about the roles involved in the programme can be found in Table 5.4.

Table 5.4: Role's information

<table>
<thead>
<tr>
<th>Role</th>
<th>External/Internal</th>
<th>Processes Involved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Manager/ Product Owner 1</td>
<td>Internal</td>
<td>APO04</td>
</tr>
<tr>
<td>Product Owner 2</td>
<td>Internal</td>
<td>APO05</td>
</tr>
<tr>
<td>Product Owner 3</td>
<td>Internal</td>
<td>APO12</td>
</tr>
<tr>
<td>Product Owner 4</td>
<td>Internal</td>
<td>DSS02</td>
</tr>
<tr>
<td>Programme Scrum Master</td>
<td>External</td>
<td>-</td>
</tr>
<tr>
<td>Scrum Master/ Implementation Team Member 1</td>
<td>External</td>
<td>APO04, APO05</td>
</tr>
<tr>
<td>Implementation Team Member 2</td>
<td>External</td>
<td>DSS02, APO12</td>
</tr>
<tr>
<td>Implementation Team Member 3</td>
<td>External</td>
<td>DSS02, APO05</td>
</tr>
</tbody>
</table>

Once again, the resources for this field study were limited in number, for that reason the people available for developing the solutions had the responsibility to participate in more than one Scrum Team. We established 4 projects with only 3 implementation team members, which resulted in 3 Scrum Teams (the distribution of responsibilities of this programme is represented in Table 5.4). Moreover, Scrum Master was the same for every Scrum Team and participated in one of the projects as an implementation team member as well.

Scoping

This phase combined the first 4 steps of COBIT 5 adoption method as foreseen by the model proposed (see Fig. 5.3). Had a duration of approximately one and a half month during which the following tasks were executed:

- Selection of the key processes to improve;
- Assessment of their capability level;
- AS-IS process modeling using BPMN;
- Documentation of the AS-IS state of the processes and the COBIT 5 assessment method (PAM);
- Definition of the high-level targets for improvement;

Designing and implementing governance and management solutions
For the step 5 of COBIT 5, as foreseen by the proposal, the solutions were designed and implemented in an iterative and incremental way, using the agile approach presented in section 5.1. Designing and implementing the solutions took the rest of the project’s schedule, which was approximately two months, where the sprints addressed the following tasks:

- **Sprint 1** - First approach to the BPMN model of TO-BE processes;
- **Sprint 2** - Refine TO-BE BPMN models and design templates for the missing outputs;
- **Sprint 3** - Refine BPMN models and templates and start defining new strategies and solutions;
- **Sprint 4** - Refine solutions, BPMN models and strategies and start the documentation of TO-BE processes;
- **Sprint 5** - Refine BPMN, strategies, templates and documentation and define deadlines for implementing the solutions;

The dissertation deadline occurs at the same time as this sprint, meaning, there are still two sprints left until the end of this programme. Following we explain the goal for each remaining sprint.

- **Sprint 6** - Implement and refine part of the solutions and templates;
- **Sprint 7** - Implement the remaining solutions and templates and adapt if required;

### 5.2.2 Results

In section 5.1 we explain that Scrum should only be applied at the development and implementation step (step 5 of COBIT 5 adoption).

To decrease the resistance to the methodology mindset, we implemented the roles as soon as the project started as well as agreed weekly and programme meetings every three weeks. Nevertheless, we only used accurately Scrum during the implementation of solutions, the difference was the "system" was already established.

To implement the system of roles and meetings at the beginning, allowed to increase the involvement of each process owner on the correspondent process from the very first step. We felt their engagement in every decision required to first assess each capability level, then to define where we needed to go - define the target.

Although we introduced weekly meetings and established roles at the beginning of scoping phase, the process encompassed by this process was still waterfall, as foreseen, since we followed a sequential approach despite the weekly feedback. Scrum is not about creating the "system" and still take sequential processes, it is a mindset with the clear notion that requires to break the product in small pieces and refine and adapt those pieces while building the big final product.
5.3 Evaluation

This section corresponds to the Evaluation step of DSRM. Here we will describe how we intend to evaluate how well an artifact represents a solution to the problem identified.

To evaluate the main artifact's performance during the demonstration according the selected criteria from Prat et. al (Table 4.5), we will use the feedback obtained through a set of semi-structured interviews to the people involved in the demonstration. To assess the quality of the life-cycle model we used the Moody and Shanks framework, and to assess the constructs created to build the rules of language, we used the Wand and Weber method.

5.3.1 Evaluation of the demonstration

In this section we will evaluate the results achieved in the demonstration, in order to validate the artifact and confirm that it can be used to solve the problem stated.

The evaluation step from DSR method involves comparing the results observed from its usage in demonstration with the goals established for the solution.

As already mentioned in the first iteration, we will use a few criteria selected from the hierarchy of criteria proposed by Prat et al. [43] to evaluate its performance during the demonstration.

For this second DSR iteration a set of 8 semi-structured interviews were performed to the people involved in the demonstration, similarly to the first iteration, to evaluate the artifact in terms of the criteria selected. The interviews were face to face and have taken an average time of 15 minutes.

We used the same rating scale for all questions, with the following agreement levels: 1-Strongly Disagree, 2-Disagree, 3-Neutral, 4-Agree, 5-Totally Agree.

On 5.5 we present, for each evaluated criteria, the average rating regarding each agreement level.

The information about the interviewees can be seen in Table 5.5.

<table>
<thead>
<tr>
<th>Role</th>
<th>Experience in IT (years)</th>
<th>Function in IT</th>
<th>First COBIT 5 implementation programme?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Manager / Product Owner 1</td>
<td>21</td>
<td>IT Coordinator</td>
<td>No</td>
</tr>
<tr>
<td>Product Owner 2</td>
<td>10</td>
<td>PMO</td>
<td>No</td>
</tr>
<tr>
<td>Product Owner 3</td>
<td>25</td>
<td>Systems Security Director</td>
<td>Yes</td>
</tr>
<tr>
<td>Product Owner 4</td>
<td>22</td>
<td>Associate IT Coordinator</td>
<td>No</td>
</tr>
<tr>
<td>Programme Scrum Master</td>
<td>30</td>
<td>IT Governance Professor</td>
<td>No</td>
</tr>
<tr>
<td>Implementation Team Member 2</td>
<td>1</td>
<td>IT Governance Investigator</td>
<td>No</td>
</tr>
<tr>
<td>Implementation Team Member 3</td>
<td>1</td>
<td>IT Governance Investigator</td>
<td>No</td>
</tr>
</tbody>
</table>

For each one of them was asked to give their feedback of the methodology used regarding the following criteria:
• Efficacy

First, the application of the methodology on the field experiment is for itself a proof that we can manage a COBIT 5 programme with the solution encountered within the time established.

To evaluate whether the methodology was effective producing its derided objective of facilitate the implementation, when compared with a traditional method, we asked for them to classify two statements according the agreement level aforementioned: (1) The methodology facilitated the adoption; (2) Is more easy to achieve success with the methodology used than with a traditional one;

Regarding the first statement, 57% of the interviewees strongly agreed and the other 43% agreed.

For the second statement, we obtained the same results, 57% strongly agreed and 57% agreed.

Overall, the main advantages they pointed were: the weekly meetings, which allowed them to give feedback and keep up with the project as well as the programme meeting each three weeks; the fact that when we were working on the project we were working at the organization; the feedback, which allowed to identify early inconsistencies, while still designing and implementing the solutions without having to wait until the end of the project to realize it.

Therefore, we can conclude that the methodology proposed demonstrated to be effective for managing a COBIT 5 programme, and that the people who worked with this method believe that facilitates when compared to a traditional approach.

• Utility for people

For assessing this criterion, we sought to understand if review meetings, the establishment of process owners and the incremental and iterative development were helpful in the designing and implementing process, for example for building solutions more aligned with their needs.

As we can see looking at the results from 5.5, almost everyone (71%) strongly agreed on the methodology utility for designing better solutions and 29% agreed.

Overall, the feedback was very positive, especially when we asked them to give examples of the methodology utility in the project. We would like to highlight the answer from Product Manager, that resumes the general opinion: "(The methodology helped) Anticipating organizational decisions that had to be adapted to our reality. The fact that we are following up on a weekly basis made it easier for us to make a decision that, if it happened later, could lead us to redo many things or to have a very faithful implementation plan to COBIT but not very faithful to the reality of the organization"

• Utility for the organization

For evaluating the utility for the organization we asked for them to classify two statements: (1) The methodology enabled a greater involvement of process owners (2) The methodology decreased
Regarding the involvement of process owners, the results showed that the majority (71%) agreed and 29% strongly agreed that the methodology contributed for a greater involvement. However, the results were not so positive when we asked for their opinion on the resistance to change felt. The results were inconclusive since the majority of them (43%) classified their agreement level as neutral (neither agreed or disagreed), one of them disagreed that the resistance to change decreased despite the methodology used, another one totally agreed, and the rest (29%) agreed. Overall, 43% agreed or strongly agreed on the statement, and 57% was neutral or disagreed.

In fact, the feedback regarding the involvement of process owners was very positive, since they all agreed on the fact that weekly feedback loops allowed them to be more connected, having to decide on project direction. Still, some of them felt that despite this obligation established they often hold over the weekly meetings when some more important task arise, which is a kind of resistance.

**Ease of use**

For evaluating the ease of use, we tried to understand how easy was for them to follow the methodology, knowing this criterion depends of the organizational context.

6 out of 7 agreed and 1 was neutral. This last one, argued that for him it was not always easy to perform the weekly meeting with us. As showed in Table 5.5, all process owners assigned for the role are senior managers, meaning they deal with numerous problems every day. Meetings were often rescheduled, due to their responsibilities. This factor was not a concern, since we were full time allocated in the organization three days a week.

**Accuracy**

Regarding the life-cycle, we only tested the Waterfall initial phase of scoping and the Scrum process for developing the solutions, the final Waterfall step (Fall) for testing the solutions was not part of the field experiment.

Respecting the methodology practices, a small analysis between the expected outcomes and the results from their application was made:

- **Meetings**: We conducted separate weekly meetings with each process owner. These meetings were divided in two parts: on the first part, we explained and showed the increment produced in the previous week in order to requested feedback (sprint review); on the second part, we discussed what we planed to do in that week (sprint planning). Every three weeks, we performed a programme meeting divided in two parts as well: on the first part we
presented the work done and requested feedback (inspect and adapt); on the second part we discussed the upcoming work for the next three weeks (programme increment planning). Additional meetings such as Scrum of Scrums and PO Sync were not adopted.

– **Roles:** Each process was assigned a process owner, which corresponded to the the senior manager responsible for that area in the organization. The IT Coordinator was both a process owner and the product manager for the programme.

– **Artifacts:** Each sprint backlog and process backlog was jointly created each week in a coordination effort between the process owner, the scrum master and the team. Sprint Backlog was developed and managed using Trello (Fig. 5.4). Due to the previous experience, we decided to use a single board, where we assigned each task with a tag [APO12],[APO04],[APO05] or [DSS02].

![Figure 5.4: Backlog management in Trello](image)

### 5.3.2 Moody and Shanks quality model framework

To assess the quality of our model artifact, we are going to use the Moody and Shanks quality management framework [45]. With this framework for quality model evaluation, we intend to measure with identified metrics [46, 47] some quality factors [45]. In Table 5.6 we show the quality factors and the metrics used on the assessment.
Table 5.6: Quality factors and metrics used for assessing the model using Moody and Shanks

<table>
<thead>
<tr>
<th>Quality Factor</th>
<th>Definition</th>
<th>Metrics Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completeness</td>
<td>Refers to whether the data model contains all user requirements</td>
<td>Number of user requirements which are not represented in the data model.</td>
</tr>
<tr>
<td>Simplicity</td>
<td>Simplicity means that the data model contains the minimum possible constructs. Simpler models are more flexible, easier to implement, and easier to understand</td>
<td>Number of entities.</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Is defined as the ease with which the data model can cope with business and/or regulatory change</td>
<td>(1) Number of elements in the model which are subject to change in the future. This includes changes in definitions or business rules as a result of business or regulatory change. (2) Estimated cost of changes.</td>
</tr>
<tr>
<td>Understandability</td>
<td>The ease with which the concepts and structures in the data model can be understood</td>
<td>Implementation team members rating of understandability</td>
</tr>
<tr>
<td>Implementability</td>
<td>The ease with which the data model can be implemented within the time, budget and technology constraints of the project</td>
<td>Implementation team members rating of implementability within time and budget, providing an important “reality check” on what is technically possible and/or economically feasible.</td>
</tr>
</tbody>
</table>

- **Completeness**: For this assessment, we will compare the construct 5.1, 5.2 and 5.3 that defines the rules with the model that represents the life-cycle 5.3. Since a project life-cycle is defined as “the collection of all project phases, defining its beginning and ending” [35] and a project phase is “a collection of logically related project activities, usually culminating in the completion of a major deliverable” [35], we will only consider as a user requirement the events (phases) and the artifacts (deliverables) defined in the construct. Given this, all events and artifacts (meetings and deliverables) defined in the construct are represented in the model. Objectives are merely statements
with no formality, therefore they are not expressed in the model, moreover not only features but also stories are small parts of a bigger artifact, so regardless they are not officially represented, those artifacts are. For all those reasons, we can consider the model is complete.

- **Simplicity:** The complexity of the model will be measured using the metric “number of entities”, for which we established as a qualitative target that should to be less than the number of elements defined for the events and artifacts (maximum complexity acceptable). Counting the elements on the model we have 11 events represented in the model and 5 artifacts, plus 2 additional elements from a maximum of 10 events and 8 artifacts. Number of events = 11 > maximum = 10, number of artifacts = 5 < maximum = 8. For representing 18 elements, we have 18 elements (11 + 5 + 2 = 16 for representing the construct + 2 additional entities). Therefore, we can infer that the model is not very simple, since does not represent all the elements required with less entities.

- **Flexibility:** The model is sufficiently general so it can be used for any particular instantiation of the artifact. Scrum of Scrums, PO Sync and backlog refinement are not mandatory meetings/activities, however, the model is sufficiently understandable so the cost of not performing those meetings regarding possible changes on the model is merely none, for does not affect the remaining process description. For all the other elements there is no risk of change for changes in the business environment, thus, we can consider it is very flexible model.

- **Understandability and Implementability:** During the interviews with the team involved in the demonstration programme (sub-section 5.3.1), we specifically asked for the implementation team to classify the understandability of the model so we can measure this factor. The authors involved in the programme did not participate in any of the interviews related to the evaluation of either the model or the demonstration. For that reason only two members had the conditions to be part of the evaluation and be able to classify the model.

In a range from 1 to 5 (Very Hard; Very Easy), the ITM2 (Implementation team member 2) classified the understandability of the model as Very easy (5) to understand and ITM3 (Implementation team member 3) classified as Easy (4).

Regarding implementability, since we conducted two field studies where we applied the model in a real environment of a COBIT 5 programme, we proved that is implemented within the time and budget established, since there was not delays or additional costs.

Overall, we could conclude with this evaluation that the model is **complete** for it contains all user requirements for the events and artifacts. Does not **simplify** the requirements using less elements. Is **flexible**, since is general enough to reflect any reality of implementation with the methodology proposed, is **understandable** and **implemented** for we used it in a field study.
5.3.3 Wand and Weber method

Additionally, for the evaluation of the constructs’ ontology we will use the Wand and Weber method (BWW-method) [48] to analyse the effectiveness of the grammar represented in constructs. Using this method, we will be able to identify four ontological deficiencies: incompleteness, redundancy, excess and/or overload.

A grammar is defined as structure that “provides a set of constructs and rules that show how to combine the constructs to model real-world domains” [48], where a construct represents “an element of the union of the sets V and T, describing a set of objects or things that have common properties”, which we can relate with our sub-artifact that we previously identified as constructs (Tables 5.1, 5.2 and 5.3).

As previously mentioned, the constructs were build as a sub-artifact of the methodology to create the language (grammar) of our methodology.

According to Wand and Weber [48], we have to compare mathematically both sets in order to assess the artifact's effectiveness: “to evaluate whether a grammar provides a clear representation of a real-world construct, we rely on basic notions from the mathematics of mappings. In the analysis we focus on two sets: the set of real-world constructs we obtain from the ontological model; and the set of grammatical constructs we obtain from a description of the grammar.”.

In our case, the set of real-world constructs (“constructs that an ISAD grammar must be able to describe” [48]) is the left side of the table (SAFe definitions), and the set of grammatical constructs that describes the grammar is the right side (Scaled Scrum for COBIT 5).

As aforementioned, the method proposed allow us to identify four ontological described in Table 5.7.

<table>
<thead>
<tr>
<th>Deficiency</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incompleteness</td>
<td>Can each element from the first set be mapped on an element from the second set? (the mapping is partial if there are first set elements without mapping).</td>
</tr>
<tr>
<td>Redundancy</td>
<td>Can each first set elements be mapped on more than a single second set element? (the mapping is redundant if it is ambiguous).</td>
</tr>
<tr>
<td>Excess</td>
<td>Can each second set element be mapped on a first set element? (the mapping is excessive if there are second set elements without a mapping).</td>
</tr>
<tr>
<td>Overload</td>
<td>Can each second set element be mapped on exactly one first set element? (the mapping is overloaded if at least one element of the second set is mapped to more that one element of the first set).</td>
</tr>
</tbody>
</table>

- **Incompleteness**: It is clear that the the mapping is partial. When we look, for example, to Table 5.1 we verify that there are cells with no correspondence, elements from the SAFe framework that have no correspondent element on the grammar created, as for example the System Architect/Engineer role from SAFe. The same happens at least with one element in each Table. Therefore, according to the definition, the grammar is ontologically incomplete.

Our justification for this deficiency is supported on the fact that SAFe is a very broad framework (as analysed in section 3.3) and to reduce the complexity to form a simpler version of SAFe was
always the intent of this research work. Although it creates an ontologically deficiency, we consider this specification of elements as an important, and necessary, part of the solution.

- **Redundancy**: The grammar is not redundant, since for each element of SAFe there is at most one correspondent element in the solution set.

- **Excess**: At first sight, may seem that the grammar has not construct excess, however there are some elements, namely: the scoping phase (Water phase) and the Fall phase, that are not directly mapped in none of the SAFe’s concepts. Those elements came from a another solution (Water-Scrum-Fall).

- **Overload**: For every element of the solution set there is a unique correspondent element of SAFe. Hence, the grammar has not construct overload.

Overall, were found incompleteness and excess deficiencies in the grammar. The explanation is sustained on the fact that our goal was always to eliminate some elements of SAFe that are not necessary or either applicable in a COBIT 5 programme. In a similar way, the structure of COBIT 5 required non agile steps that were are not predicted by SAFe, which caused an excess deficiency.

Although we recognize these deficiencies between two sets of constructs, we also have to perceive that we build a new solution based on a few SAFe concepts and a few other practices from literature, which resulted in a specification of SAFe with additional concepts and not exactly a mapping between two frameworks.

### 5.3.4 Conclusion

According to the DSR method, to evaluate the artifact is to "observe and measure how well an artifact supports a solution to the problem, comparing the objectives to the results observed from use of the artifacts in the demonstration" [1].

As for the first iteration, the incremental and iterative strategy of an agile approach proved to facilitate the adoption of new IT governance practices.

Looking at the results, 71% of the interviewees agreed and 29% strongly agreed that the methodology enabled a greater involvement of process owners (senior managers). Therefore, we can conclude the **objective 1** was achieved.

Moreover, 71% of the interviewees totally agreed and 29% agreed the methodology was very useful for the people involved regarding the identification of early inconsistencies, largely because of the weekly and three weeks’ feedback loops at team and programme level, so we can consider the **objective 2** was met.
Respecting objective 3, we can only consider it was poorly met, since some of the people involved argued that despite the methodology forced them to be involved, they often hold back weekly meetings when more important work arise, which is a form of resistance.

Overall, the results obtained from the interviews with the people involved showed that the field experiment was way more accurate with the proposal than the first approach.

In the next chapter, a discussion on the overall research is made. We start by discussing the lessons learned, following we identify the limitations found during this research work, next we explain how we communicated the results and finish with topics for future work.
6

Conclusion

Contents

6.1 Discussion ......................................................... 65
6.2 Limitations ......................................................... 66
6.3 Contributions .................................................... 66
6.4 Communication ................................................... 67
6.5 Future Work ....................................................... 67
To conduct this research, we followed the DSRM process, which comprises 6 phases of development. First we identified the problem of lack of adoption of formal project development methodologies on COBIT 5 programmes. Our main objective was presented as being to facilitate COBIT 5 adoptions, decreasing some known challenges through the usage of an agile approach for managing the programme. To address this problem, and to make the implementation an easier task, increasing top management commitment, detect misaligned solutions and decrease resistance to change, an agile methodology based in Scrum - Scaled Scrum for COBIT 5 - is proposed.

Two iterations on DSR process were performed. Starting on the design and development step, each iteration of the solution was demonstrated in a distinct COBIT 5 programme, and evaluated with interviews in order to compare our goals with the results observed from its usage.

The methodology comprises two artifacts: a set of constructs to define the rules and practices in which the implementation will operate and a model that illustrates the life-cycle of implementation.

At the first DSR iteration we designed the first approach of the methodology, which we had the opportunity to test as a field study in the scope of a COBIT 5 programme within the Portuguese Finance Ministry. The second DSR iteration was a refinement of the first one, where we incorporated the most relevant findings derived of the first experiment and results of the evaluation. For demonstrating the second approach, the methodology was applied in the scope of another COBIT 5 programme on bank X. For evaluating the solution we used the Moody and Shanks’ framework and the Wand and Weber method for evaluating the sub-artifacts as well as performed interviews to the team involved in the experiment.

6.1 Discussion

To perform two iterations of the methodology, testing each one of them in a singular experiment of a COBIT 5 programme in different organizations, allowed us to identify some relevant findings that we are following discuss.

Regarding the first iteration, we realize that less is not always better and, in this case, a lighter approach resulted in confusion specially concerning meetings at team level, which caused a weak distinction between programme level and team level. Moreover, from the interviews with experts in Scrum and COBIT 5, we understood that the part of life-cycle that should incorporate Scrum was not a consensus for everyone, and some solutions like Sprint 0 were not very welcomed.

For the second iteration, our goal was to refine the initial solution according the findings gathered. More research was conducted in order to incorporate new scaled mechanisms to the methodology, and a new approach for the life-cycle was developed, where we eliminated Sprint 0 and limited Scrum adoption only to the implementation step, after the definition of each project scope.

Finally, the evaluation outcome confirmed the achievement of the solution’s objectives by increasing
senior management support, being effective regarding the detection of misalignments early in the development and reducing the resistance to change, which proved to facilitate COBIT 5 adoptions, not only from a solution’ development perspective but also for communicating the benefits for the organization.

6.2 Limitations

Within this research work, we found a few limitations regarding the proposal’ development:

• The greatest limitation found during the solution development was the notorious lack of literature regarding lessons learned on COBIT 5 adoptions. We often felt lack of theoretical support for designing the approach.

• The interviews with experts were performed as an evaluation step, after we tested the approach in the field. We believe we would obtain a greater benefit if it has been done before the demonstration step, to gather more information and fill the gap in literature.

Furthermore, we also found limitations during the demonstrations:

• The group of people responsible for designing and implementing the solutions was small in both demonstrations. We do not have a perception of the impact of this issue on the results.

• To facilitate the adoption of the methodology in the demonstration, looking to reduce the resistance to the approach, we defined and used the roles at the beginning and not only for implementing the solutions as established, although we only defined the projects and divided the work when designing and implementing the solutions, as defined by the solution. For an accurate demonstration, the Scrum approach should have been introduced only in step 5.

6.3 Contributions

Within the scope of this work thesis the following contributions were made: (1) a visual and detailed life-cycle model for COBIT 5 adoptions, as well as a set of definitions for each element foreseen in the solution; (2) lessons learned and findings regarding two real COBIT 5 adoptions, enriching the literature on a subject that has such a considerable gap in literature, and that is so requested by practitioners; (3) an agile alternative to the traditional life-cycle of COBIT 5 for practitioners to use. With the demonstration experiment, we proved its applicability in COBIT 5 initiatives and that it can more effective achieving the final objectives within the schedule in opposition to a traditional approach;
6.4 Communication

Communicating the problem, the solution and the results to the scientific community is the last step comprised by the DSR model.

During the writing of this thesis we were able to compile the first iteration solution, findings and field study at the Portuguese Finance Ministry results into a scientific paper.

The paper was already accepted, presented and published 2018 IEEE 22nd International Enterprise Distributed Object Computing (EDOC 2018):


6.5 Future Work

Regarding the outcome of this thesis work, there are several research opportunities that can be addressed for future work:

- Demonstrating and evaluating the proposed methodology in more organizations, with different sizes and different teams in order to compare results and identify possible constrains;
- More research on COBIT 5 phases, seeking for new and improved ones, which was something we could not address in this thesis context;
- Integrate Lean framework as an opportunity to provide a set of principles as further guidance;
- Further research on new agile opportunities, specially regarding scaled agile methods, in order to find and test new approaches;
- Refine this methodology for the new COBIT that is expected to be released soon;
- Explore the contributions of this methodology for adopting other IT governance and management frameworks (ex: ITIL, CMMI, etc.).
Bibliography


Appendix A: Interviews with experts
Questionnaire about the methodology proposed based on Scrum for managing the COBIT 5 programme

1. Personal information
   1.1. What is your education level?
   1.2. What is your position in the IT function?
   1.3. Experience in IT (in years):

2. COBIT 5 and GEIT (IT Governance)
   Follow the instructions gave at each line so we can correctly answer. When asked, rate each statement according the agreement level indicated.

   2.1. Classify your knowledge regarding COBIT 5.  
   Scale: 1 = marginal knowledge, 2 = know a little, 3 = know, 4 = know well, 5 = expert

   2.2. Do you know the COBIT 5 implementation guide for ISACA?  
   a)Yes □  b)No □

   2.2.1. (If yes) Have you ever used the guide?  
   a)Yes □  b)No □

   2.4. Do you know anyone who used COBIT 5 for implementing IT governance practices? And COBIT 4? (If the answer is no, jump to section 3)  
   COBIT 5: a)Yes □  b)No □  
   COBIT 4: a)Yes □  b)No □

2.5. What is the average time it takes a COBIT 5 initiative.

   a)<3 meses □  b)3-9 meses □  c)12 meses □  d)>12 meses □  e) não sabe □

2.6. Classify the difficulty of implementing IT Governance with COBIT 5.  
   Scale: 1 = very dificult, 2 = dificult, 3 = moderate dificulty, 4 = easy, 5 = very easy

   1  2  3  4  5

73
2.7. In your opinion, what are the main challenges when adopting COBIT 5?

__________________________________________________________________________________________________________________________________________

3. Scrum

3.1. Classify your knowledge regarding Scrum.

Scale: 1 = marginal knowledge, 2 = know a little, 3 = know, 4 = know well, 5 = expert

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

3.2. Mark with an X the elements from Scrum that you know.

Scrum Roles:

- Scrum Master □ Development Team □ Product Owner □

Scrum Artifacts:

- Product Backlog □ Sprint Backlog □ User Story □ Product Increment □

Scrum Events:

- Daily Scrum □ Sprint Planning □ Sprint Review □ Sprint Retrospective □ Backlog Grooming/Refinement □

3.3. In your opinion, what are the main advantages of using agile methodologies?

__________________________________________________________________________________________________________________________________________

3.3.1. Advantages?

__________________________________________________________________________________________________________________________________________
3.4. Do you agree that we can use Scrum for projects outside of software-development?

Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
</table>

4. Scrum for COBIT 5

In this topic, we aim to validate the proposal of using an agile approach for COBIT 5 adoptions.

For the following elements of the proposal, look at Table 1 in order to give your opinion on each element of the table and indicate if it is valid under a COBIT 5 programme.

4.1. Scrum Events

4.1.1. Backlog Refinement

4.1.2. Daily Scrum

4.1.3. Sprint Planning

4.1.4. Sprint Review

4.1.5. Sprint Retrospective
Table 1: Events “Scrum for COBIT 5”

<table>
<thead>
<tr>
<th>Scrum Events</th>
<th>Original Definition</th>
<th>Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Backlog Refinement</td>
<td>Product Backlog Refinement is the activity in which the Product Owner and the Development Team collaborate for revising Product Backlog items, adding detail, estimating value, and reordering the items. This activity happens at the end of each sprint.</td>
<td>This activity is fundamental for a COBIT 5 programme given its numerous process dependencies. Same purpose as a typical backlog refinement.</td>
</tr>
<tr>
<td>Daily Scrum</td>
<td>Is a stand-up meeting with maximum duration of 15 minutes, every day in the same place at the same time. The goal is for every team member to expose any conflicts or issues with the remaining team members.</td>
<td>May not be applied. It is difficult in a complex programme where the resources are sometimes outsourced to reunite everyone in the same place at the same time every day.</td>
</tr>
<tr>
<td>Sprint Planning</td>
<td>This meeting is for planning the next sprint. Every development team member selects a few user stories from Product Backlog to implement in the next iteration and the result is the Sprint Backlog.</td>
<td>Only at project level. At the beginning of each sprint, with the same purpose as a typical sprint planning meeting.</td>
</tr>
<tr>
<td>Sprint Review</td>
<td>Meeting at the end of each Sprint, where the Scrum Team and relevant stakeholders inspect the increment and agree on changes in the Product Backlog. The main goal is to show the items “done” and get feedback on the increment, which may be accepted or rejected by the client.</td>
<td>Only at project level. At the beginning of each sprint, with the same purpose as a typical sprint review meeting.</td>
</tr>
<tr>
<td>Sprint Retrospective</td>
<td>Meeting at the end of the Sprint, where the team inspects itself and the work performed and creates a plan with improvements for them to apply in the next sprint. Occurs after each Sprint Review and before each Sprint Planning.</td>
<td>Only at project level. At the beginning of each sprint, with the same purpose as a typical sprint retrospective meeting.</td>
</tr>
</tbody>
</table>

4.2. Scrum Artifacts
For the following elements of the proposal, look at Table 2 in order to give you opinion on each element of the table and indicate if it is valid under a COBIT 5 programme.

4.2.1. Product Backlog

4.2.2. Sprint Backlog
4.2.3. User Story

User Story is the representation of a requirement as an item of Product Backlog. It is a simple and easy way to create context around a task. User stories are short and should be something like: As a <type of user >, I want <some feature >, so that <some goal >.

4.2.4. Product Increment

Product Increment is a usable part of work done in each sprint. Product Increment is the group of all items from the Product Backlog that were “done” during the last sprint. This increment will sum to the previous product increment if its complete enough to demonstrate its working functionality.

Table 2: Artifacts “Scrum for COBIT 5”

<table>
<thead>
<tr>
<th>Scrum Artifacts</th>
<th>Original Definition</th>
<th>Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Backlog</td>
<td>Product Backlog is an ordered list of all the functionalities, functions, requirements, improvements and corrections that represents a change in the product. It is Product Owner’s responsibility during the whole project, including its creation, manage its content, availability, and ordering its items.</td>
<td>Product Backlog is unique to the entire programme. It is an ordered list of user stories related to the agreed deliverables to implement governance enablers.</td>
</tr>
<tr>
<td>Sprint Backlog</td>
<td>Sprint Backlog is a subset of Product Backlog. It is the list of selected items from Product Backlog for a specific sprint. It is a result of the development team selection of user stories to work on in that iteration.</td>
<td>Sub set of items from Product Backlog, with the same purpose of a typical Sprint Backlog. Each team has its own Sprint Backlog since they have separate contexts to work on.</td>
</tr>
<tr>
<td>User Story</td>
<td>Representation of a requirement as an item of Product Backlog. Is a simple and easy way to create context around a task. User stories are short and should be something like: As a &lt;type of user &gt;, I want &lt;some feature &gt;, so that &lt;some goal &gt;.</td>
<td>Must be simple and short.</td>
</tr>
<tr>
<td>Product Increment</td>
<td>A usable part of work done in each sprint. Product Increment is the group of all items from the Product Backlog that were “done” during the last sprint. This increment will sum to the previous product increment if its complete enough to demonstrate its working functionality.</td>
<td>Processes are not as easy to deliver as features of software applications. The main goal will be to improve process capability with small deliveries of necessary work products to achieve the established target. A possible deliverable (product increment) may be a work product or any other necessary output, for example: architecture design, process documentation, workshops, etc.</td>
</tr>
</tbody>
</table>
4.2. Scrum Roles

For the following elements of the proposal, look at Table 3 in order to give you opinion on each element of the table and indicate if it is valid under a COBIT 5 programme.

<table>
<thead>
<tr>
<th>Scrum Roles</th>
<th>Original Definition</th>
<th>Scrum for COBIT 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product Owner</td>
<td>This person is the client representative throughout project. He/she is responsible for creating and managing the content of the Product Backlog. His/her job is to manage the Product Backlog according to the clients’ interests, ordering the items to reflect the client’s priorities.</td>
<td>Because of the several dependencies encompassed in a COBIT 5 programme, it was established that the role of Product Owner is unique to programme and project levels. He must maintain programme integrity and monitor the dependencies between projects. In the scoping phase, he is the one responsible for identifying the business priorities and issues triggering need to act. He should keep the same responsibilities as the original product owner regarding product backlog.</td>
</tr>
<tr>
<td>Implementation Team</td>
<td>Also known as development team. Group from 3 to 9 professionals whose main responsibility is to turn Product Backlog’ items into tangible Product Increments. This team is self-organized and cross-functional, deciding autonomously which part of the Product Backlog is done in each Sprint.</td>
<td>Since COBIT 5 is a programme, the authors established that each project should correspond to a process identified in sprint 0. For each project, there must at least one implementation team and one scrum master. Implementation team is responsible for designing and implement the established governance solutions in the process (see product increment). They must also participate in the AS-IS capability assessment.</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>Scrum Master is responsible for promoting and supporting scrum among team and non- team members, as he/she must also coach the development team.</td>
<td>Same responsibilities as a typical Scrum Master.</td>
</tr>
<tr>
<td>Scrum Team</td>
<td>Scrum Team = Implementation Team + Product Owner + Scrum Master</td>
<td>Scrum Team = Implementation Team + Scrum Master. Product Owner is unique for the whole programme.</td>
</tr>
</tbody>
</table>

4.3.1. Product Owner

4.3.2. Development Team
4.3.3. Scrum Master

4.4. Life-cycle

For the following elements of the proposal, look at Figure 1 in order to give you opinion on the life-cycle model and indicate if it is valid under a COBIT 5 programme.

4.4.1. Do you agree with the life-cycle? Justify

4.4.2. In your opinion, what is the most adequate duration for an iteration (Sprint)? (between 1 week and 1 month)

4.4.3. What is your opinion about the seven implementation phases proposed by ISACA?

Figure 1: Life-cycle model of “Scrum for COBIT 5”
Appendix B: Interviews for the 1st demonstration
**Questionnaire about the application of Scrum to implement IT Governance with COBIT 5 in the Portuguese Finance Ministry**

1. **Senior Management Commitment and Support**

   1.1. In a scale from 1 to 5, how do you classify the senior management interest on the programme?

      *Scale: 1 = none, 2 = little, 3 = neutral, 4 = few, 5 = a lot*

      | 1 | 2 | 3 | 4 | 5 |
      |---|---|---|---|---|

   1.2. In a scale from 1 to 5, how do you classify the involvement of senior management in the solution development?

      *Scale: 1 = none, 2 = little, 3 = neutral, 4 = few, 5 = a lot*

      | 1 | 2 | 3 | 4 | 5 |
      |---|---|---|---|---|

   1.2.2  *(If the answer was >3)* What do you believe were the factors that contributed for this?

       __________________________________________________________

       __________________________________________________________

       __________________________________________________________

2. **Solution aligned with requirements**

   2.1. The solution provided at the end was the one planned at the beginning?

       __________________________________________________________

       __________________________________________________________

       __________________________________________________________

   2.2. Now that the programme is over, do you believe the initially planned solution was the one they wanted the most?

       __________________________________________________________

       __________________________________________________________

       __________________________________________________________
2.3. Did the sprint review help aligning the solution to their vision?

3. Resistance to change

3.1. Did the client show an understanding of what were the benefits of the solution at the end of the programme better than at the beginning?

Scale: 1 = strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = strongly agree

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

3.2. What was your perception about the client’s interest on accepting the changes suggested for them to implement?

3.2.1. Do you believe it was created a good relation with the client? If yes, that engagement had any impact in their acceptance?
Appendix C: Interviews for the 2nd demonstration
Questionnaire about the methodology used for managing the COBIT 5 programme

1. Personal information
   1.1. What is your education level?
   1.2. What is your position in the IT function?
   1.3. Experience in IT (in years):
   1.4. Have you ever participated in a COBIT 5 programme adoption?
       a) Yes □ b) No □

2. Scrum-based methodology evaluation

   Follow the instructions gave at each line so we can correctly answer. When asked, rate each statement according the agreement level indicated.

   2.1. The methodology facilitated the adoption.  
       Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

   2.2. Scrum meetings were important to align the solutions to the organizational needs, and detect early inconsistencies.  
       Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

   2.2.1. Give an example.

   ___________________________________________________  
   ___________________________________________________

   3. The methodology enabled a greater involvement of process owners.  
       Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

   ___________________________________________________
   ___________________________________________________  
   ___________________________________________________

84
2.4. The methodology decreased the resistance to change.

Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

2.5. The methodology was easy to follow.

Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

2.5.1. If the context of adoption changed? For example, different organization.

Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

2.6. Is more easy to achieve success with the methodology used than with a traditional one.

Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

3. Model evaluation

The aim of this topic is to evaluate the model of the proposal and its correspondence with the reality of the demonstration.
3.4. The model is easy to perceive.

Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

3.5. The methodology used follows the model.

Scale: 1 = Strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, 5 = totally agree

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>
Appendix D: Results from the interviews
<table>
<thead>
<tr>
<th>Criteria</th>
<th>Strongly Disagree</th>
<th>Disagree</th>
<th>Neutral</th>
<th>Agree</th>
<th>Totally Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Goal efficacy</td>
<td></td>
<td>43%</td>
<td>57%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Utility for people</td>
<td></td>
<td>29%</td>
<td>71%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Utility for the organization</td>
<td>14%</td>
<td>43%</td>
<td>50%</td>
<td>22%</td>
<td></td>
</tr>
<tr>
<td>Ease of use</td>
<td></td>
<td>14%</td>
<td>86%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ease of perception of the model</td>
<td></td>
<td>57%</td>
<td>43%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Goal efficacy</th>
<th>Strongly Disagree (Level 1)</th>
<th>Disagree (Level 2)</th>
<th>Neutral (Level 3)</th>
<th>Agree (Level 4)</th>
<th>Totally Agree (Level 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The methodology facilitated the adoption</td>
<td></td>
<td></td>
<td></td>
<td>43%</td>
<td>57%</td>
</tr>
<tr>
<td>Is more easy to achieve success with the methodology used than with a traditional one</td>
<td></td>
<td></td>
<td></td>
<td>43%</td>
<td>57%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td>43%</td>
<td>57%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Utility for the organization</th>
<th>Strongly Disagree (Level 1)</th>
<th>Disagree (Level 2)</th>
<th>Neutral (Level 3)</th>
<th>Agree (Level 4)</th>
<th>Totally Agree (Level 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The methodology enabled a greater involvement of process owners</td>
<td></td>
<td></td>
<td></td>
<td>71%</td>
<td>29%</td>
</tr>
<tr>
<td>The methodology decreased the resistance to change</td>
<td>14%</td>
<td>43%</td>
<td>29%</td>
<td>14%</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>14%</td>
<td>43%</td>
<td>50%</td>
<td>22%</td>
<td></td>
</tr>
</tbody>
</table>