Universidade do Minho
Escola de Engenharia

Francisca de Sousa Ribeiro

Development of a grooming process for an agile software team in the automotive domain

Dissertation Thesis
Integrated Master's Program in Industrial Management and Engineering

Work done under the guidance of
Professor Anabela Tereso
Doctor Deborah Perrotta de Andrade

October 2017

“I never lose, I either win or learn”

Nelson Mandela

ACKNOWLEDGEMENTS

I would like to express deepest gratitude to the people that empowered me to work on this dissertation.

To my supervisor Anabela Tereso and advisor Deborah Perrotta, for the continuous guidance, availability, willingness to help and for all the given suggestions that enriched the present dissertation.

To the Cloud Team, for letting me be part of them and trust on my work since the beginning. Their cooperation and commitment were really important for this dissertation development. And a special thanks to my mentor at Bosch, André Ferreira, for sharing his knowledge and experience. Thank you for always supporting and believing in me and my work.

To my team at Bosch, Agile Council, who contributed, in one way or another, to the development of this work. I am grateful for all the assistance provided, the encouragement and friendship.

I cannot forget my friends who went through hard times together, cheered me on and celebrated each accomplishment. To my colleagues, who became friends for life, for making these five years the best years of my life.

I am grateful to my family for the unconditional love, endless patience and for encouraging me in the worst moments. A special thanks to Diogo for his unconditional trust, timely encouragement, and for being an inspiration throughout last years.

ABSTRACT

At the present VUCA (Volatile, Uncertain, Complex and Ambiguous) world, customer's requirements demanding companies' flexibility are growing. At this unstable market pace, and being software one of the fastest growing business areas, it becomes even more difficult to fulfil and adapt the product to changes required by the customer. To overcome this situation, the *Agile* methodology was created with the purpose of continuously deliver product increment, allowing teams to collect feedback from stakeholders during the process and to adapt their work between work iterations.

Bosch Group, and specifically the Development department at Bosch Braga is taking the first steps into adopting the *Agile* methodology, more specifically Scrum framework. However, considering the dimension of the company, the lack of communication between the teams and the teams' involvement on the management level are transversal problems.

This action research project was meant to find a solution to this problem with the objective of achieving more autonomous and committed teams to their projects. The first step was to define the grooming/ refinement process known in Scrum that teams should follow. Since the impact of its application on all software teams could be huge, it was decided to focus on one team only. The purpose was to collect feedback at the end and then deciding about its application to the remaining teams.

The best approach to implement the grooming process for the researcher was to be part of the team, more precisely to be their scrum master. This allowed a better understanding of the team's dynamic and how they managed their work, to smoothly implement the process. This collaborative participation also enabled the improvement of parallel topics that had a large impact on their performance and, of course, on a better implementation of the grooming process. This dissertation is focused on describing the implementation of a grooming process within Scrum and on a specific software team. Moreover, other improvements were implemented as a result of the exercise of research. The feedback from the team was very positive, providing the trigger to extend the process implementation on remaining teams.

KEYWORDS

Project Management; Agile Methodology; Scrum; Product Backlog Grooming

RESUMO

No atual VUCA (*Volatile, Uncertain, Complex and Ambiguous*) *world*, a flexibilidade exigida pelos clientes às empresas é crescente. A este ritmo de mercado cada vez mais instável e sendo o desenvolvimento de *software* uma das áreas de negócios com crescimento mais rápido, torna-se ainda mais difícil responder e adaptar o produto às mudanças exigidas pelos clientes. Para colmatar esta situação, surgiu a metodologia *Agile*, cujo objetivo é entregar continuamente pequenos incrementos do produto, ao contrário das abordagens tradicionais de gestão de projeto. Isto permite assim às equipas adaptar o seu trabalho através do *feedback* contínuo dos *stakeholders.*

O departamento de Desenvolvimento da Bosch Braga, na qual o presente projeto de dissertação teve lugar, já adotou a metodologia *Agile* com a implementação do *Scrum*. No entanto, e considerando a dimensão da empresa, a falta de comunicação entre as equipas e o envolvimento das mesmas ao nível da gestão são problemas transversais.

A presente dissertação teve como propósito encontrar uma solução para este problema, com o objetivo de tornar as equipas mais autónomas e comprometidas com os seus projetos. Para isso, definiu-se um processo de refinamento, conhecido em *Scrum* como *Grooming,* que as equipas pudessem seguir para guiar o seu trabalho. Uma vez que o impacto da sua aplicação em todas as equipas de *software* era elevado, foi decidido implementar primeiramente apenas numa equipa e só depois, após o resultado, decidir a implementação nas restantes.

A melhor abordagem para implementar o processo foi a integração do investigador na equipa como *Scrum Master*. Isto permitiu uma melhor compreensão da dinâmica da equipa e da forma como gerem o trabalho, de modo adaptar e inserir o processo sem mudanças abruptas na sua eficiência. Adicionalmente, esta participação colaborativa permitiu implementar outras melhorias que otimizaram o desempenho da equipa, o que facilitou também a implementação do processo. Assim sendo, a presente dissertação foca-se na descrição da implementação de um processo *grooming* numa equipa de *software*. O feedback positivo por parte da equipa demonstrou que seria benéfico ampliar a implementação do mesmo processo às restantes equipas.

PALAVRAS-CHAVE

Gestão de Projeto; Metodologias Ágeis; *Scrum*; *Product Backlog Grooming*

INDEX

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CM	Car Multimedia
CC	Chassis Control
CR	Customer Requirement
DoD	Definition of Done
IT	Information Technologies
PB	Product Backlog
PBI	Product Backlog Item
PM	Project Manager
PO	Product Owner
ROI	Return On Investment
SB	Sprint Backlog
SM	Scrum Master
SP	Story Points
SWPM	SoftWare Project Manager
US	User Story

1. INTRODUCTION

1.1 Background

Project Management is a knowledge area that has shown its first steps in the Egyptian era, with empirical management evidences. With the evolution of time, tools and techniques were developed, and nowadays it is already possible to use different methodologies created by specific entities in project management to drive the projects. However, all software development projects that follow v-model or waterfall, whose focus is on delivering the final product at once, have this specific characteristic in common: all project requirements and scope must be settled at an early stage, which seems not to be appropriate to an increasing need to cope with unpredictability of customer needs (Balaji, 2012).

One of the main concerns of business leaders is how to deal with the present unstable business environment, known as VUCA – Volatile, Uncertain, Complex and Ambiguous world. Coined by the US Army College on the 90s, this acronym *"reflects an increasingly unstable and rapidly changing business world"* according to Lawrence (2013, p. 2). At this market pace, customers require scope changes during the project execution, in a way that it turns to be a challenge for sponsors to achieve that flexibility. In this sense, more important than reducing the lead time is to create adaptive projects that embrace changes during the project's development.

To fulfil this need in the software market pace, the Agile software methodology was created in 2001 by a group called Agile Manifesto. This methodology deals with the continuous value delivery, incrementally, welcoming changing requirements, communications, and sustainability, among others (Hundermark, 2015). Its main principle is being adaptable and iterative, following the Inspect and Adapt cycle (Dean, Pessanha, Langfeldt, Pritchard, & Stanger, 2006). This will allow changes to the project scope, that is built dynamically, making it possible to be changed up to 30% in each re-iteration (Špundak, 2014). Moreover, the feedback and lessons learned are important weapons to improve, evaluate short parts of work and adjust them to the main goal (Sliger, 2006).

"Agile is all about continuous improvement" (Crisp & McKenna, 2016, p. 24)

1.1.1 BOSCH Group

Robert Bosch created, at the age of 25 years old, his first mechanical office of electronics precision in Stuttgart. Since then and until now, Bosch Group became a technology and services leader supplier. The group's strategy goes along the long-term economic success, not neglecting a social and philanthropic orientation. With this, in 1964, Robert Bosch Foundation was created to develop training, art, culture and science areas (Bosch, 2016b).

We are Bosch is the theme of this group's vision, and it aims to guarantee the company's future by preserving its financial independence, and ensuring its strong and significant development.

"Invented for life: we want our products to spark enthusiasm, improve quality of life, and help conserve natural resources" (Bosch, 2016a)

In the beginning of 2016, the group incorporated 375.000 associates in 150 different countries. Around 3.600 of them are in Portugal, being represented by Bosch Thermo Technology S.A., in Aveiro, Bosch Car Multimedia Portugal S.A, in Braga, and Bosch Security Systems S.A., in Ovar. In the end of this year, the company earned 73.1 billion euros in sales, from which 60% were from Mobility Solutions. This business area includes Car Multimedia (CM), which supplies infotainment, display, connectivity and Human-Machine Interface (HMI) solutions for passenger cars, trucks, coaches, two-wheelers, and off-highway vehicles (Bosch, 2015).

Bosch Car Multimedia Portugal, S.A.

The unit in Braga is one of the main factories of Car Multimedia (CM) division from Bosch. The Bosch Car Multimedia Portugal, S.A. keeps its activity in the development hub and production for the automotive industry, including electronics equipment, mainly car radios, and navigation systems. Nowadays, Bosch is one of the biggest private companies in the region, with approximately 2800 employers. The CM division focuses on developing intelligent solutions, inside the vehicle, in order to turn entertainment functions, navigation and drive assistance integration more flexible and efficient. The business core goes through the instrumentation systems manufacturing and development for automotive industry, from prototype conception to mass production (GlobalNet, 2016b).

Bosch CM Portugal, S.A. is organized in two functional areas - commercial and technical, as illustrated in Figure 1.

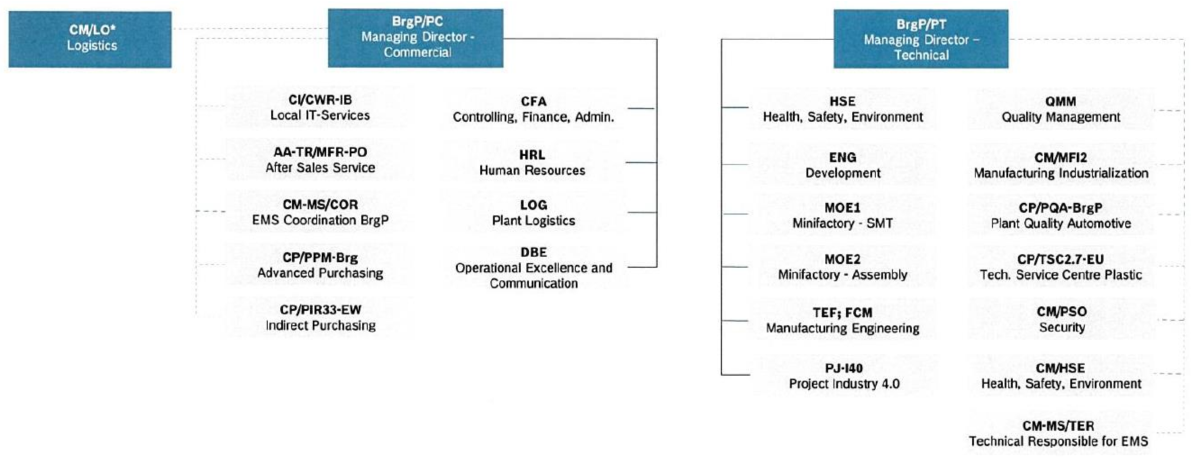

CM/LO* Logistics
BrgP/PC Managing Director - Commercial
CI/CWR-IB Local IT-Services
AA-TR/MFR-PO After Sales Service
CM-MS/COR EMS Coordination BrgP
CP/PPM-Brg Advanced Purchasing
CP/PIR33-EW Indirect Purchasing
CFA Controlling, Finance, Admin.
HRL Human Resources
LOG Plant Logistics
DBE Operational Excellence and Communication
BrgP/PT Managing Director – Technical
HSE Health, Safety, Environment
ENG Development
MOE1 Minifactory - SMT
MOE2 Minifactory - Assembly
TEF; FCM Manufacturing Engineering
PJ-I40 Project Industry 4.0
QMM Quality Management
CM/MFI2 Manufacturing Industrialization
CP/PQA-BrgP Plant Quality Automotive
CP/TSC2.7-EU Tech. Service Centre Plastic
CM/PSO Security
CM/HSE Health, Safety, Environment
CM-MS/TER Technical Responsible for EMS

Figure 1 - Organizational Chart of Bosch Braga Portugal
(Bosch, 2017)

Engineering and Development Department

This action research was developed in the Engineering and Development Department (ENG) created in 2015, which is divided into three sections that represent three different business areas: Car Multimedia (CM-CI2), Chassis Control (CC) and Innovation. ENG aims for innovation and prototypes development, to create new products and patents and collect projects. Its goal is to produce new optimized products, with quality and in time (GlobalNet, 2016a).

1.1.2 General Problem

Bosch Group has, as shown, a complex organizational structure, due to the company's dimension and its large number of employees. This complexity brings a huge problem, which is the root cause of many other problems faced by the software teams at CM-CI2: lack of efficient communication.

In fact, the teams never have direct contact with the client, instead, it is represented by a Bosch collaborator, working in the client's company, called the Resident. Thus, he is the only contact that Bosch has with the client, and only the Project Manager (PM) is allowed to contact him. The role of the PM is to keep the different knowledge areas of the projects aligned: software, hardware, and mechanics. But for each one of these areas, there is a Technical Leader which, in case of software, concerns the Product Owner (also called the software project manager - SWPM).

Despite the development teams being located in Braga, the PMs and Product Owners are in different development hubs, mainly in Germany. In this sense, the Product Owners of Braga software teams are not in the same development hub, which implies having secondary Product Owners inside the team, at Braga Development Centre. This complex connection illustrated in

Figure 2 hinders the communication flux and increasingly separates each of the parties from each other. This situation detaches the development team from the Product Owner, not involving them in long-term planning. Although the Product Owner is the responsible for breaking down the work, he is also a PM, therefore not always having the capacity to keep the team updated or providing them with smaller work packages ready to be done. Most of the times, he tends to "spoon feed" the teams with tasks, not allowing, as mentioned before, the team's involvement in the breakdown process. In fact, there is not a defined process for the breakdown from customer requirements to small work packages ready to be implemented. Or if there is, the teams in Braga are not aware, nor have any access to it. This distance generates less team's commitment to the project once they do not understand why they are doing the work and what the main goal is. Therefore, it becomes the secondary PO's responsibility to keep the work packages updated, clarifying with the main PO what really needs to be done. It is also his responsibility to keep the customer needs updated and determine the business value of each increment.

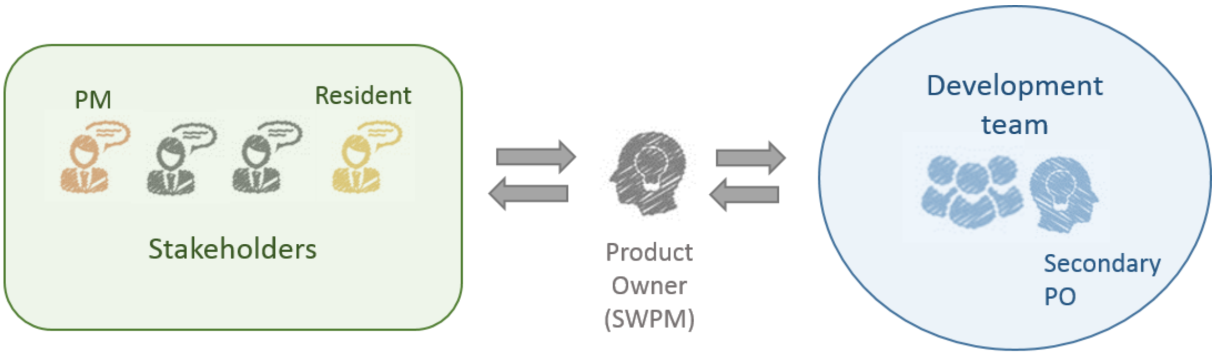


Figure 2 - Communication channels on software teams, at CM-CI2

The teams are sharing, even more, the necessity and willingness to participate in the work breakdown, giving their contribution on the best way to achieve a milestone and giving inputs for the work prioritization. If the teams were able to accomplish this, they would be improving their autonomy and commitment, which would make them more mature and efficient, bringing also more value to the project.

"Forecasting the financial value of a theme is the responsibility of the Product Owner, but it is a responsibility shared with all other team members" (M. Cohn, 2006, p. 91)

1.2 Goals

Scrum is the most used Agile framework in the world for organizing and managing work, combining specific roles, artefacts, and ceremonies. The Product Owner, more related to the project manager role, is the most difficult role due to the responsibility for maintaining and prioritizing the work (Sverrisdottir, Ingason, & Jonasson, 2014).

However, this breakdown and prioritization of the work – named the refinement/grooming process in Scrum – is not a mathematical process and there are many approaches in literature. This process breaks out the customer requirements, given by the stakeholders, into specific work for the team to perform in one work cycle. Depending on the organization and the work methodologies that the company follows, this process can differ. On BOSCH Group, more specifically in Braga, there is no evidence of its existence, teams do not follow any specific practice.

The main goal of this dissertation is to define a process, from the elicitation of customer requirements until its implementation by the team. For this, it is crucial to have the team more involved once they are who have the knowledge, and who can give information about their performance and work. It is also important to define how the team will give their support, how they will benefit from it, and which contributions they can give. The Product Owner and the team will both benefit with the improvement of this communication and interaction. The Product Owner will be able to build a more realistic and accurate release plan. And the team will be able to participate in long-term planning, facilitating their planning sessions, and contributing to their performance improvement. To conclude, all this definition and improvements are also goals that this dissertation intends to achieve.

1.3 Research Methodology

The Action Research methodology was chosen to be followed in order to achieve the goals presented in the previous section. Briefly, it constitutes a cycle of diagnosis, action, and evaluation, whose main intent is to help guide investigations. The investigation was held in close contact with the company, being the researcher inserted in the company's environment. This participative and collaborative methodology allows having both action and research outcomes, in order to better fulfil the goals mentioned.

Given the complexity of the organization and the large number of parts (teams) involved in the problems stated, it was necessary to reduce the sample size to only one team on which this dissertation acts upon. The team that was chosen to try and test new approaches was the Cloud

team (more details in section 3.2), from innovation projects, that develops Cloud Applications for Smart Cars. The decision was made based on the fact that they are the most autonomous and stable team in Scrum, once their Product Owner is in the same work hub. Besides, being from innovation, they have more flexibility to make changes and to adapt to new methods.
The researcher integrated the Agile Team since the beginning. The team's goal is to support all other teams to follow the Agile methodology, mainly Scrum, in order to improve their efficiency and performance. Even after having been nominated as the team's Scrum Master, the researcher's role as Agile coach was always present, also providing help to other teams, giving Scrum workshops, and participating as a project manager in other projects acquisition.

1.4 Structure of Dissertation

This dissertation is structured in eight chapters. This first chapter presents an introduction to the topic, the context where the research was developed, the problem addressed and the main goals of the research. In chapter 2, Literature Review, the main knowledge areas involved are presented, from the macro to the micro abstraction level. It starts with the contextualization of project management, then Agile methodology, and only then the Scrum framework and its main areas. Chapter 3, Methodology, approaches more deeply than previously, the methodology used to guide the dissertation, and the necessary adjustments to fit on the problem's focus. Next, chapter 4, Grooming Process Development, is the largest, where the diagnose phase of the chosen team is described. This chapter also includes other action research stages, mainly the diagnosis of the team, the action plan of each topic approached and the iterations that were needed in order to achieve a possible solution; these possible solutions are described in chapter 5, Results, where the final iteration to accomplish the goal is presented. To finish the action research cycle, Evaluation and Learnings follows on chapter 6. In here, the opinions from the team on whom this action research focused are included, as well as some conclusion and benefits observed after the improvement measures implementation. In addition, in chapter 7, Related Work, a parallel work related to Scrum is presented as well, but associated to other software teams. Finally, and to close this dissertation, the Conclusion chapter is presented on 8.

2. Literature Review

This chapter aims to provide a perspective upon which this study was based, by reporting what already exists in literature about the main topics addressed in the present dissertation. With the aim of keeping a logical order, the chapter starts with the literature review about Project Management, which concerns to a high level of abstraction, and it will be deepened into the Agile Project Management and finally to Scrum. Scrum is the thematic that will be more developed including general explanations and also other more specific topics, distributed in three sub-chapters: Scrum Framework, Scrum Tools, and Velocity: Measuring and Monitoring.

2.1 Project Management

As a knowledge area, project management has shown an exponential growth in the last years. However, it has been applied since the earliest times of civilization. According to Kwak (2005), project management has been applied for thousands of years, since the Egyptian era. Traces of empirical projects management were found, associated with the construction of Egypt's pyramids (2550 B.C.) as well as the Wall of China (220 B.C.).

From the 1950s onwards, this discipline became relevant as the larger, and more complex, projects in organizations felt the need to be supported by tools and project management techniques. This decade was marked by its application in the United States Navy, more exactly in the Polaris Project (Kwak, 2005). Given the magnitude of the project, involving hundreds of contracts, it was necessary to develop a management method as an attempt to control it. Thus, the basic concept of PERT analysis arose, applied on a project jointly with a consulting firm. The main objective was to emphasize the milestones, instead of activities, to facilitate management control as a whole, through progressive key points.

By the end of this decade (1958), the networking method was developed, later named Precedence Diagramming Method (PDM). Then, the "Circle and Connecting Line", later designated Activity-on-node Networks, was also developed in 1959, showing a simpler application in relation to previous methods (Fondahl, 1987).

In the 1960s, project cost management appeared, along with the resource levelling, which joined time management as project management techniques. In this way, the network techniques were deepened in this decade and used to plan, program and control projects. In the meantime, the National Aeronautics and Space Administration (NASA) was created back in 1958, but only later did the most prominent project of the decade start - the Apollo Project. In

1969, NASA set up the Apollo Program Office, in order to have more project management control, whose objective was to provide the following functions:

- Schedule missions using PERT;
- Acquire and contract suppliers;
- Measure performance through management systems;
- Establish the scope of the Apollo Program (Kwak, 2005).

To conclude, project management principles and tools were mostly used by the US Department of Defence, NASA and large construction industries to manage high budgets by that time.

The creation of the first project management bodies in this decade expressed the expansion and the fast development of the project management area. First, in 1965, the International Project Management Association (IPMA) emerged at European level, originally a professional forum where it was possible to share knowledge and experiences related to network planning. It is currently a global organization encompassing fifteen project management national associations, mostly European. Later, in 1969, the North American Project Management Institute (PMI) was created (Stretton, 2007).

The 1970s were marked by the exponential expansion of the project management application, encompassing all types of industry (chemical, pharmaceutical, banking, accounting, etc.) as well as government agencies and the United Nations. In addition to improvements, there is also a wide range of new techniques and tools including Work Breakdown Structure (WBS), responsibility assignment matrices and earned-value methods. With this, the importance of organizational structures increases, where conflict management is a present concern (Stretton, 2007).

The recognition of this area of knowledge as a new field of study and a profession was also achieved in this period of time:

> *"A series of other papers looking at the role of project managers, organizational methods for project management and managerial strategies profiled a new industry, a new field for research and a new management style - a new profession" (Stretton, 2007, p. 10).*

With the industrial revolution of information technology and the advance of software development, software project management methodologies started to arise and have become easily accessible to companies. In this way, project management techniques were available, enabling greater efficiency in the management and control of complex projects.

The 1988 Winter Olympics applied project management techniques to manage the event and it proved to be successful. Thus, unconsciously, it was possible to propagate this area also for management of future events (Kwak, 2005).

Until the 1990s, project management was seen as innovative, but not essential to the survival of organizations. It was from this decade onwards, that companies started to feel a greater competitiveness and commercial pressure resulting from the internationalization of the markets and globalization (Baptista, Santos, Páscoa, & Sändig, 2016).

With the arrival of the Internet in all areas of business, it was possible to lead organizations to achieve higher levels of productivity, being more efficient and customer oriented. Between 1995 and 2000, project management communities aligned themselves with new technologies and the Internet in order to become more efficient in performing project management and control (Kwak, 2005).

2.2 Agile Project Management

At the present time, the business world shows an increase in complexity, since market requirements are changing faster, constituting a challenge for the sponsors (executives with overall accountability for the project). More than ever, they must reduce time to market, without neglecting the delivery of innovative products to achieve customer satisfaction. Agile software development is a methodology that promises to fulfil all these needs, offering benefits as on-time delivery, once it increases value to the business in short iterations (Schön, Thomaschewski, & Escalona, 2016). This methodology started to appear in 2001 but its reputation expanded when a group of recognized specialists joined to create a movement named Agile Manifesto. Agile is also mentioned on the PMBOK as a methodology that can be followed to implement project management (PMI, 2013, p. 2). It includes a set of values and principles shared by all of the authors, such as to continue delivering value incrementally, to welcome changing requirements, to promote communications and sustainability, among others (Beck et al., 2001; Torrecilla-Salinas, Sedeño, Escalona, & Mejías, 2016). Since its conception, it has been adopted all over the world, starting to be adopted by companies from innumerable business areas, besides software.

> *"Agile is a vast global movement that is transforming the world of work" (Denning, 2016a, p. 1).*

With the increasingly uncertain markets, changes during projects are more and more appreciated by the users. Accordingly, Agile allows development teams to adapt their product

to those changes, which meets the actual market revolution (Verner, Brereton, Kitchenham, Turner, & Niazi, 2012). In each iteration, project scope is being settled on a dynamic mode, allowing it to be upgraded up to 30% (Špundak, 2014).
According to Denning (2016b), there are three fundamental laws which Agile organizations share:

1. Small team law;
2. Customer law;
3. Network law.

The first one suggests the use of small and multidisciplinary teams, working in small cycles in order to get a faster feedback. Due to this, if issues occur, they will be detected earlier hence avoiding undesirable costs and rework. The second one aims at increasing customer's product value. Because of this, the company tries to iteratively suit the goals, processes, and practices, to reduce everything that the customers are not willing to pay and continuously produce earned value to the product. Finally, and being the key to the whole process, Network law emphasizes transparency and collaborative work to a common goal, intrinsic to this philosophy (Denning, 2016b).
Agile methodology worries about the past to optimize the future, leading to the Inspect and Adapt cycle. In each iteration, both process and team's progress review are made, in order to help them on decision making about how to better proceed to the next iteration. In this manner, the client inspection allows the development team to adapt their plans, so that they can prioritize future work. This cycle and Agile principles are shared by all Agile methodologies in the market, such as Extreme Programming, Scrum, Crystal Dynamic Systems Development Method, and others (Pikkarainen, Haikara, Salo, Abrahamsson, & Still, 2008; Sliger, 2006).

> *"Agile is all about attitude and mentality, and really very little (or nothing) about a pre-set number of documents and rituals"(Christoffer, 2015, p. 1).*

2.3 Scrum Framework

> *"Scrum remains the most used Agile project management method" (Zahraoui & Janati Idrissi, 2015, p. 1)*

Scrum is one of the most popular Agile frameworks, being used all over the world by Information Technologies (IT) companies. It concerns to a management and control process derived from *"knowledge management, complex adaptive systems and empirical process* *control theory"* (Hundermark, 2015, p. 8), which aims at ending complexity and building software that satisfies business needs:

> *[Scrum] "It is an adaptive, iterative, fast, flexible, and effective methodology designed to deliver significant value quickly and throughout a project" (Tobergte & Curtis, 2013, p. 2).*

The Scrum framework is based on project management and is sustained by three pillars: transparency, inspection, and adaptation. First, it is important to have visible the most significant aspects of the process without neglecting that it should be understandable by all users. The inspection must be frequent, as it enables to detect undesirable variances that deviate the process from the main goal. If it happens, the process must be adapted as soon as possible. The framework involves a development team, the Scrum Team, organized by roles within a set of artefacts, ceremonies, and rules.

All ceremonies are time-boxed and guided by the Scrum Master, who represents the team, helping them to efficiently use the Scrum Process (Schwaber & Sutherland, 2016). Therefore, there is no Agile "project manager" role. Instead, his responsibilities and roles, from the "traditional project manager", were distributed and shared between the parts of the Scrum Team, more precisely the Product Owner (PO), Scrum Master (SM) and the Development Team (Dev. Team) (Mountain Goat Software, 2017b).

The Product Owner represents the business, users, and customers, working to orientate the team to the right goal. In contrast, the Scrum Master is the team's coach, or a facilitator, who assures the team respect and follow the Scrum Process to achieve their highest performance level (Mike Cohn, 2017b). The Scrum Team can be briefly described as written below on Figure 3.

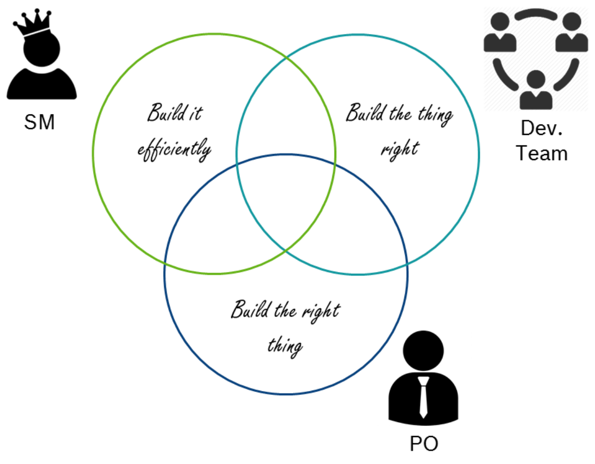


Figure 3 - Scrum roles
(Author's Elaboration)

2.3.1 Scrum Cycle

The heart of Scrum is each one of the small iterations of work, called sprints, which are time-boxed generally with the same duration, and with the aim of delivering a product increment. Thus, a new sprint starts immediately after the end of the previous one. Figure 4 provides an overview of the Scrum work cycle. It starts with a meeting with the stakeholders, where the Project Vision is settled. At that point, the Product Backlog (PB) is created and prioritized by the Product Owner (PO). This Product Backlog is basically a prioritized list of requested project functionalities, also named Product Backlog Items (PBIs), represented on the left side of Figure 4. These PBIs have different sizes and can be features/functionalities, User Stories (US) or EPICS (big user stories). A US can be described as a unit ready to be estimated and initiated in a sprint (Mahnič & Hovelja, 2012).

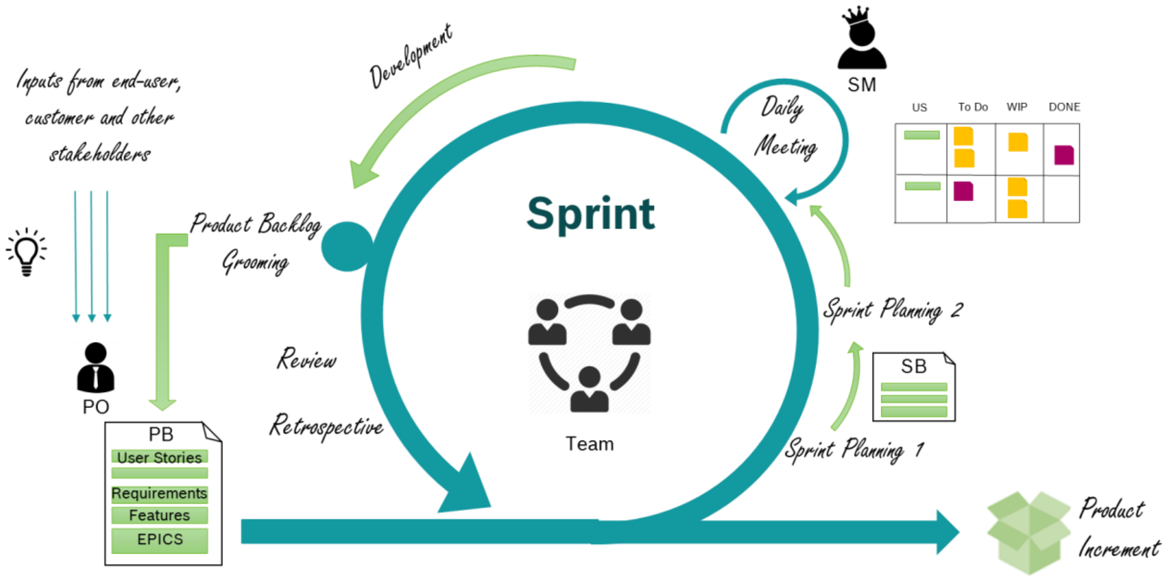


Figure 4 - Scrum cycle
(Author's Elaboration)

The sprint cycle begins with the Sprint Planning where the team decides which work will be included in the sprint, as the PB could represent weeks or even months of work. The sprint planning is divided into two different phases with two different purposes. The Sprint Planning 1 starts with the Product Owner presenting and clarifying the PB to the team, where they also review the Definition of Done (identical to a checklist that guarantees the user story is really finished). Then, they will define the Sprint Goal, which concerns the objective that should be met in the conclusion of the sprint, providing the team with the reason why the increment is built. Considering the circumstances of the sprint (sprint duration, team capabilities, DOD, etc.)

the team votes on the complexity that each PBI implementation requires. Since this ceremony, such as the other ones, is time-box, the team will estimate the complexity for as many PBIs they have time to. After this, the team will decide, based on the estimation, which User Stories from the prioritized PB they will commit to accomplish by the end of the sprint. These committed user stories for the sprint constitute the Sprint Backlog (SB). This process is named by Tobergte & Curtis (2013) as the Approve, Estimate, and Commit User Stories process. In Sprint Planning 2, the team decomposes the SB into individual and more granular tasks in order to specify how to implement the user stories committed earlier.

To conclude, the first half of planning concerns the requirements and answers to *what* the team can deliver at the end of a sprint. The second half of the meeting concerns to a high-level design, and answers to *how* the team will do the work (Hundermark, 2015; Sutherland, 2010).

During the sprints, a Daily Stand-up Meeting or just a Daily Meeting occurs, with 15-minute maximum duration, to synchronize the team. It usually happens at an appointed time in the same place, usually in front of their Kanban board, described below in section 2.4.2. At this moment, the team members share what they have done, what they plan to do next, obstacles they are facing and highlights, to keep the motivation.

"This is the key inspect and adapt meeting" (Schwaber & Sutherland, 2016, p. 11).

In each sprint, the Scrum team works to create an increment or a shippable product, entitled Accepted Deliverables. Before the end of sprints, stakeholders and Scrum team meet in the Sprint Review meeting to check (inspect) the increment delivered by the development team and to gather feedback to adapt. As an output, it is expected to gather topics or improvement measures to update or improve the product, and also update the PB if needed. The purpose of demonstrating a "Sprint Demo" is to assure that the PO approves and accepts the increment. This demo must demonstrate the product increment done during the sprint, in a way that it is understandable by the Product Owner and stakeholders.

The Sprint Retrospective meeting is the last ceremony, which concerns more to the processes the team follows to perform their work and relationships, in the same way the Sprint Review concerns the product. It is an opportunity for the team to inspect themselves, reflecting and sharing what went well and what did not. Herewith, the team should find potential improvements, what could be changed on their work methods and agree on future actions, besides documenting lessons learned to apply to future works (Tobergte & Curtis, 2013).

2.3.2 Product Backlog

To develop a project it is necessary to drive away from its vision and compile a list that contains the expected work to be done in order to achieve its objectives. In Scrum, this is named Product Backlog (PB), which includes work packages from different sizes. Each unit from this list, independently of its size, is called a Product Backlog Item (PBI).

In the beginning of the project, the PO and the team gather all the functionalities they are expecting to work on in the future. Over the project development, the backlog will be adapted and changed to a more updated one, with features that were decomposed from other features. Also, when the time to implement a feature is near, it is broken down to smaller PBIs, until its size is small enough to fit in a sprint. So, these small and detailed PBIs are the ones that will be worked on soon, named the US, which corresponds to the base unit of the PB. An illustration of what a backlog should look like is presented below, in Figure 5.

"Product backlog is a living document" (Hundermark, 2015, p. 27)

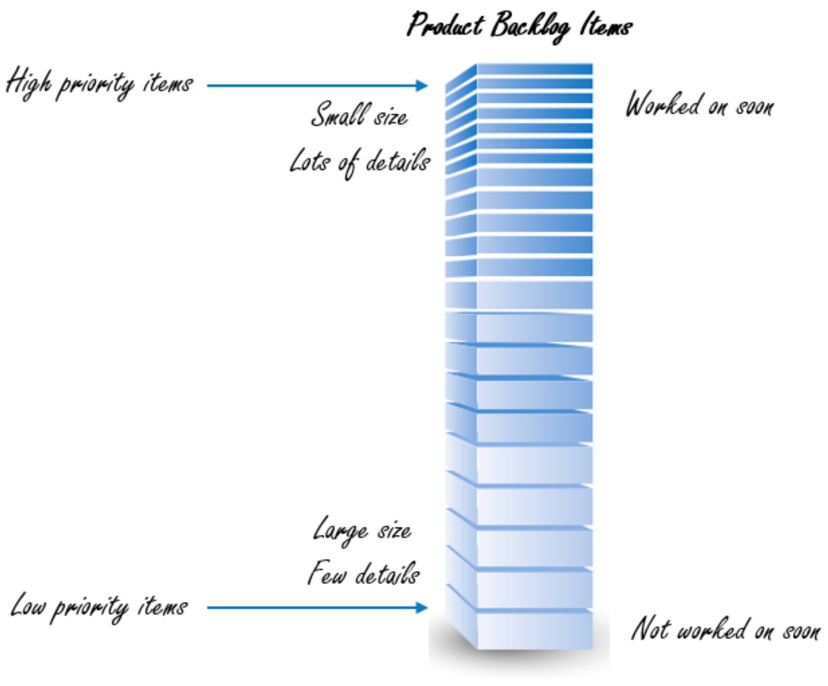


Figure 5 – Product backlog
Adapted from Rubin (2012, p. 102;104)

According to Cohn (2004, p. 17), a good US must be:

- Independent – dependencies can imply problems on planning, prioritization and estimation;
- Negotiable – a PBI should bring discussion, it is not a contract;

- Valuable – to deliver value to the stakeholders, and specially to the customer;
- Estimable – the developers should be able to estimate the PBIs;
- Small – small enough to fit in a sprint, otherwise they must be divided. On the other hand, if they are too small, the team can always combined them to form a bigger one;
- Tested – otherwise the team will never know if it was completed or not.

Usually, USs are written in a specific framework, in which the main purpose is to make its value visible for the user, translating his needs and what he wants. The Definition of Done (DoD) is what guarantees that the US was fulfilled. Many authors present their alternative frameworks to write user stories but all of them are aligned on what they should transmit.

An Epic is a very large US, regarding its complexity. It may also be a compound US. The compound ones are easier to deal with, once they just need to be split in more than one. The complex ones are more difficult because the disaggregation is not linear, so the team must find a way to adjust or split it, without losing the focus on the Epic. Yet, they can be very helpful, especially for the PO, once they allow to have the overview of the features, in a higher level of abstraction (M. Cohn, 2004).

2.3.3 Product Owner Role

The PO acts as a product manager in Scrum and that is why it is a full-time role, always focused on the business goals and the fulfilment of its needs. His primary responsibility is to maximize the Return on Investment (ROI), thus making the PO responsible for the product success or failure. He is the voice of the customer. He represents all the stakeholders and coordinates all their needs, acting like a buffer between them and the development team. Thus, communication skills are a must have for Product Owners, due to the need of transmitting different information to different individuals in different abstraction levels (Sutherland, 2010).

The PO needs to be the person who better understands the business and the product, ensuring the vision is maintained and guaranteeing it is understood by the development team. Since the PO represents the project, he is the owner of the release plan and roadmaps, creating them, and so with the responsibility of keeping them updated. Figure 6 summarizes the PO’s responsibilities.

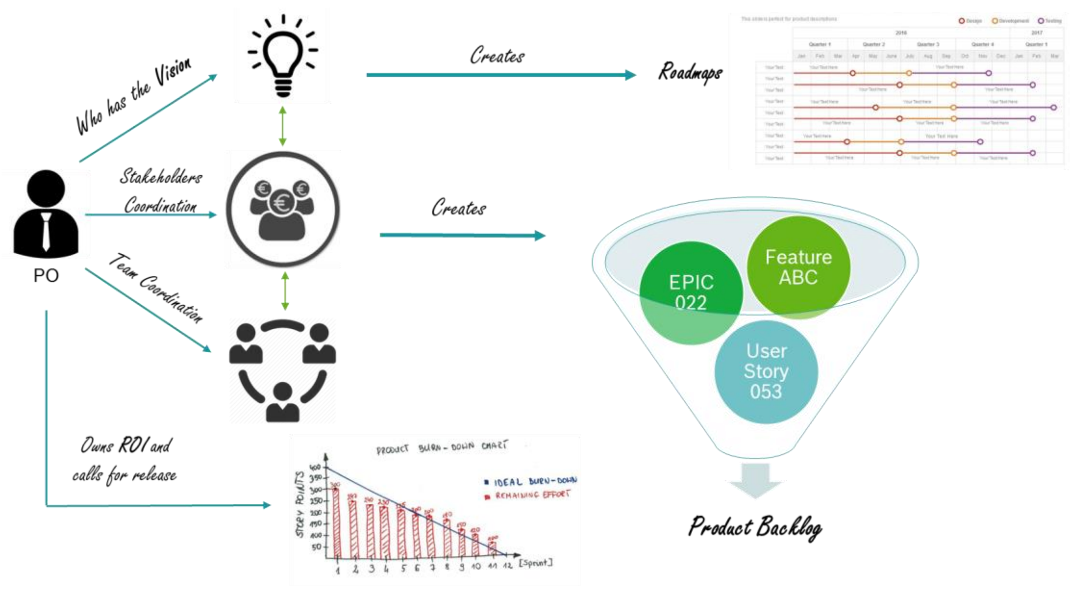


Figure 6 - Product Owner role
(Author's elaboration)

The PO is also the author and the owner of the backlog, which is created based on the requirements gathered from the stakeholders. Accordingly, he needs to breakdown the requirements and the vision into USs that are granular enough to enter in a sprint. This backlog refinement, done with the development team assistance, must be synchronized with the release plan. Therefore, the PO has authority to define what to build and when. In order to accomplish that, the backlog must be reviewed and updated whenever it is necessary, and prioritized according to the business value. Besides, the PO must guarantee the PB is visible, transparent and equally understood by all team members (Hundermark, 2015).

It is relevant to notice that PB can change up to 30%, which reflects requirements changes and consequently, scope changes. All these changes involve difficult decision making and the PO is the person responsible for them (Sverrisdottir et al., 2014). Nevertheless, when the PO shares with the team what to do in a sprint, he cannot forget to explain them why. This is very important so that they will feel connected and committed to achieve what he requests (Crisp & McKenna, 2016). The *how to do it* is defined by the development team, which is reviewed at the end of the sprint, on the review ceremony. At this point, the PO evaluates the work done and accepts or rejects the product increment delivered, taking into account the DoD and/or Acceptance Criteria previously defined (Kniberg, 2007). To conclude, the PO is the one who ensures the team is working on the right thing.

2.4 Scrum Tools

2.4.1 Planning Poker

During Sprint Planning 1, as previously referred, the team must commit with the SB. This commitment happens after the Scrum team and SM estimate the effort required to complete the USs. Some important tools can help the teams to estimate, such as Planning Poker, Fist of Five, Points for Cost Estimation and others.

> *"Planning Poker is the most used estimation technique in Scrum projects"* (Zahraoui & Janati Idrissi, 2015, p. 2).

The Planning Poker is a consensus-based technique, where the team allocates a number of Story Points that denote the effort required, to each US. Each team member has a set of 13 cards, with a predefined set of values that they choose after the discussion with the PO. The cards are revealed at the same time to guarantee the independence of the team members, avoiding influences that one can cause to another. If there is a large discrepancy between the values, then the team members discuss their vote, where the lowest and the highest share their reasons. This process is repeated until the team reaches a consensus in three rounds on average, which is supported by literature (Mike Cohn, 2017a; Moløkken-Østvold, Haugen, & Benestad, 2008; Tobergte & Curtis, 2013).

The planning poker allows the team to set a baseline, which helps them estimate by comparing with the defined baseline. It is also what turns Agile measurable and incentives the team to accomplish more story points.

2.4.2 Kanban Board

Kanban emerges with the lean concept, during the 50's in the Japanese industry, more precisely used in Toyota Production System (TPS). This tool has been increasing its popularity through many knowledge areas in the last years, gaining a position in software development (Ahmad, Markkula, & Oivo, 2013).

When applied to Scrum framework, in Agile software development environments, it can also be called Scrumban. By being a visual framework, it shows its relevance by representing a higher market share (7%) comparing to traditional Kanban (5%) (Muntean, Mircea, & Pop, 2016). Despite Scrum and Kanban have different principles they can co-exist and merge their benefits, complementing each other (Nedre, 2017).

"Scrum and Kanban aiming at increasing the overall understanding of the process transition and the methods to be used" (Nikitina, Kajko-Mattsson, & Stråle, 2012, p. 146).

The majority of Kanban boards in Scrum are represented by task boards, which aims at improving visibility and transparency, matching Agile principles (Sliger & Broderick, 2008). Having the prioritized SB and respective tasks, this enables the team members to see the progress of the work and have an overview of the tasks to be done (Lei, Ganjeizadeh, Jayachandran, & Ozcan, 2017). One example of a Kanban board for a Scrum team is illustrated below in Figure 7.

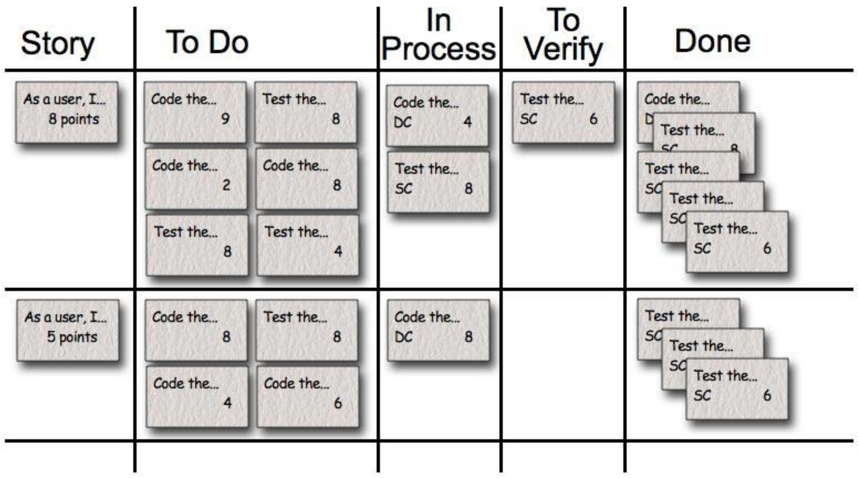


Figure 7 - Kanban board of a Scrum software team
(Mountain Goat Software, 2017a, p. 1)

2.5 Velocity: Measuring and Monitoring

2.5.1 Velocity

"Velocity is the amount of work completed each sprint" (Rubin, 2012, p. 133).

The team velocity can be translated as the rhythm the team delivers value to customers (Crisp & McKenna, 2016). When the team estimates the effort for a sprint with story points, the velocity is measured by basic math, being the total amount of story points per user stories done. However, the first estimation is uncertain, because teams always tend to use their "gut feeling", so the velocity is not always the most precise (Hundermark, 2015).

Only with time, and with an acceptable number of sprints, the measurement becomes more exact and precise, giving higher reliability to the team. This fact happens because not only the

teams increase experience and accuracy but also start to know their performance (Grapenthin, Poggel, Book, & Gruhn, 2015). Therefore, the velocity measurement comes as a baseline to help teams to predict how much work they will deliver on the next iterations.
It is also an important rate for the PO to manage release plans, build roadmaps, and forecast release scope (Hundermark, 2015).

2.5.2 Burn Chart

One of the most recognized values of Scrum framework is transparency and the burn chart is the most used to make visible the team's work. In this manner, a burn chart is a chart representation which aims to show how much work was completed during the sprint – burn up chart – and how much remains to be done – burn down chart. As velocity, if the team estimates their effort with story points, then the graphical representation will be the number of story points completed/remaining for burn up/down chart. Although the burn charts can represent work in both effort-hours or story points, the latter is the most used by the teams once the delivered value is the most important aspect (Rubin, 2012). Figure 8 is an example of a sprint burn down chart.

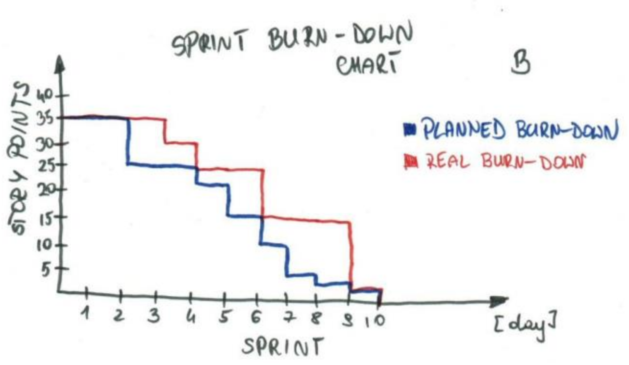


Figure 8 - Sprint burn-down chart
(STXNEXT, 2017, p. 1)

The chart should be interactive and updated by the team every day during the daily meeting, allowing the team to access if the performance of the team is in-line with expectations set at the beginning of the sprint (Scruminc., 2017). This way, it allows them to monitor progress and act upon whether performance is somehow below or above the planned, empathizing the Inspect and Adapt Agile cycle.

2.5.3 Impact on Release Plan

According to Crisp & McKenna (2016, p. 167), release plan can be defined as *"a document that shows what the team plans to deliver to customers and stakeholders"*, and the responsibility of management belongs to the PO. The main inputs are the team's velocity and the backlog already estimated (Sliger & Broderick, 2008). With these two dimensions, it is possible to calculate the number of iterations needed to deliver the increment to the stakeholders. In addition, having the iteration length, it is also possible to know the date to deliver it. Therefore, it is expectable that every time the team improves their velocity accuracy, the release plan should be updated. On the other hand, the need to update the release plan also derives from the continuous progress and change on the software requirements. For the PO to make his job right, it is essential for the SM to fulfil his part, recalculating the average velocity in every sprint, after the first three sprints at least (Crisp & McKenna, 2016). By being continuously updated, the release plan turns increasingly accurate.

2.6 Product Backlog Grooming / Refinement

The PB refinement, or grooming, is not a ceremony considered on the Scrum cycle but many authors as Rubin (2012) or Sutherland (2010), consider it essential and valuable for a good, updated and organized PB. The PB grooming is nothing more than a meeting between the PO and the Scrum team (development team and SM) to discuss the PB. This time-box meeting is an opportunity for the PO to share user stories and new features with the team. It aims to contribute and discuss future work in order to manage, organize and keep the PB updated. It is a collaborative effort, in which focus is on three main activities, according to Rubin (2012):

- Creating and refining PB items – which includes activities to add, remove, update or split PBIs;
- Estimating PBIs;
- Prioritizing PBIs.

This description is illustrated by the same authors below, in Figure 9.

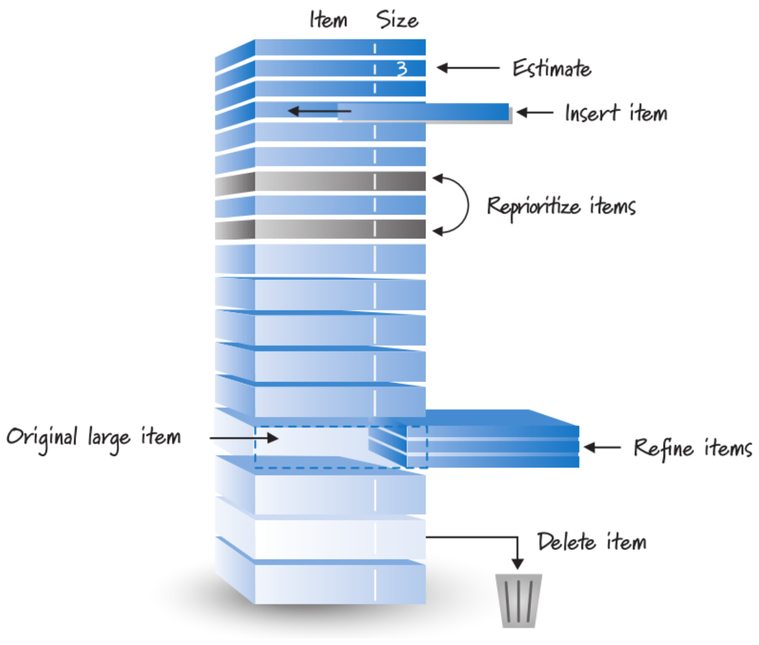


Figure 9 - Product backlog grooming activities
(Rubin, 2012, p. 105)

Considering the quick requirements change, it is really important that these meetings with the team occur continuously, so the PB could follow and reflect those changes. At this point, refining PB can include, not only changes in the user stories' scope, but also in their priority (Crisp & McKenna, 2016).

The capacity required is not a rule and differ between authors, having some of them defended the teams should invest at least 10% of each sprint capacity in grooming, like Rubin (2012), and others defending that it should happen every week, as Leffingwell (2011).

The main goal is to obtain USs ready to enter in a sprint, facilitating also the sprint planning, once it allows the team to have a better perception of what is supposed to do and its value. So, it contributes to transparency, team commitment (once they are involved), and decreasing of miscommunication and assumptions.

For the PO it is an essential tool to have an overview of what is coming, and to have a more reliable release plan, being the PB like a pipeline of requirements (Rubin, 2012).

3. METHODOLOGY

In this chapter, the methodology used in this dissertation is introduced. Also, an adaptation to this method is presented with the changes that have been made, in order to make the methodology adequate to the development environment.

3.1 Action Research

The methodology adopted in this dissertation project was Action Research, which assumes the action itself and not only the observation of actions that were already taken. For that reason, this type of research assumes the collaboration and involvement of the researcher with the organization in which he is inserted. This cooperation is crucial for the development and implementation of improvements in the current processes.
This research project approach was based on the five phases of the Action Research methodology described by Susman & Evered (1978) and later also defined by O'Brien (1998). The five phases are described in Figure 10. It should be noted that both authors recall this process to be iterative until it is possible to discover a final solution that adapts to the organization and solves the problems tackled.

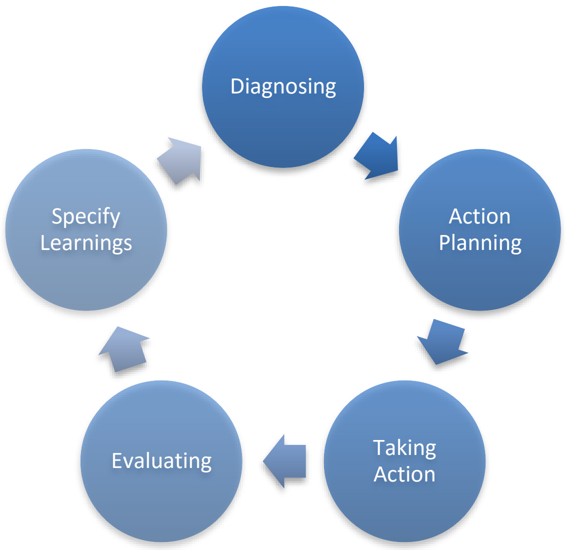


Figure 10 - Phases of Action Research Methodology
(Adapted from O'Brien, 1998)

The first phase is the Diagnosing, where observation and analysis are the main elements. At this point, the first direct contact with the teams happens and there is an interaction with them in order to be possible to understand the problems they are facing. This is the phase to collect important data, in order to find patterns and better plan what needs to be done.

Depending on the analysis made in the previous step, a plan is developed to address the problem and to take improvement actions. These actions can either be the implementation of new methods or the improvement of existing ones, if it seems suitable for the teams. In this phase (Action Planning), the development of new tools, processes, templates, and guidelines is expected, in order to increase the teams' efficiency.

It is in the Taking Action phases that all the planned ideas from the Action Planning are implemented in a defined time frame.

The fourth phase is the Evaluating of the consequences of the implemented actions, with the objective of comparing the current state with the initial state, evidencing whether or not the proposed objectives were initially achieved.

Finally, in Specify Learnings phase, the main, identified lessons learned are presented, more specifically the general findings related to the topic diagnosed and its improvements.

This Action Research cycle served as the basis to guide the research work of this dissertation, as mentioned above. However, some changes were made in order to adapt the cycle to the problems that were found, concerning primarily its implementation and its application environment.

3.2 Team

From the CM-CI2 software teams, it was necessary to choose a single one to implement the action research. Within the five teams that constitute the software section at Bosch Braga, there is only one from an innovation project – Cloud team. They are the only team that the PO is nearby, in the same development hub (Braga), which offers the best opportunity for intervention. In addition, an innovation project has a set of characteristics that distinguish it from other projects, being more appealing for an action research. Among them are the flexibility of the time plan, stability, and autonomy. With less rigid delivery dates and less "last-minute intervention" of the stakeholders, the team shows more openness to new ideas, methodologies, and tools. The fact that all the parts involved in the project are in the same city, is also contributing positively for the choice. This allows a higher engagement not only with the development team but also with all project stakeholders. For all these reasons, the Cloud team was the one selected for the action research.

3.3 Adapted Action Research

During the Diagnosing phase, more than one problem was identified, as will be presented in the next chapter. In order to have better control and tracking, each one of them was separately analysed, following their own action research cycle.

The majority of the problems only needed one iteration but there was one of them that needed continuous analysis, almost following the cycle inspect and adapt from Agile Methodology, for the same diagnose. In this sense, when following the action research cycle, the evaluation was the stage in which was noticed that the implemented process had problems that needed to be solved. At this point, it did not seem reasonable to develop either the learnings or a new Diagnosing phase, once a new iteration would be performed again. Thus, evaluating an iteration ended up by overlapping the diagnose step of the supposedly next cycle.

As a result, for the main problem that is presented in the next chapter, the entire action research cycle was only performed once. However, many iterations were done, following only the stages Action Planning, Taking Action and Evaluating. Figure 11 shows an adapted action research cycle for the current dissertation. All of the found problems fit on this approach, as they all follow the entire action research cycle once, regardless of the number of iterations.

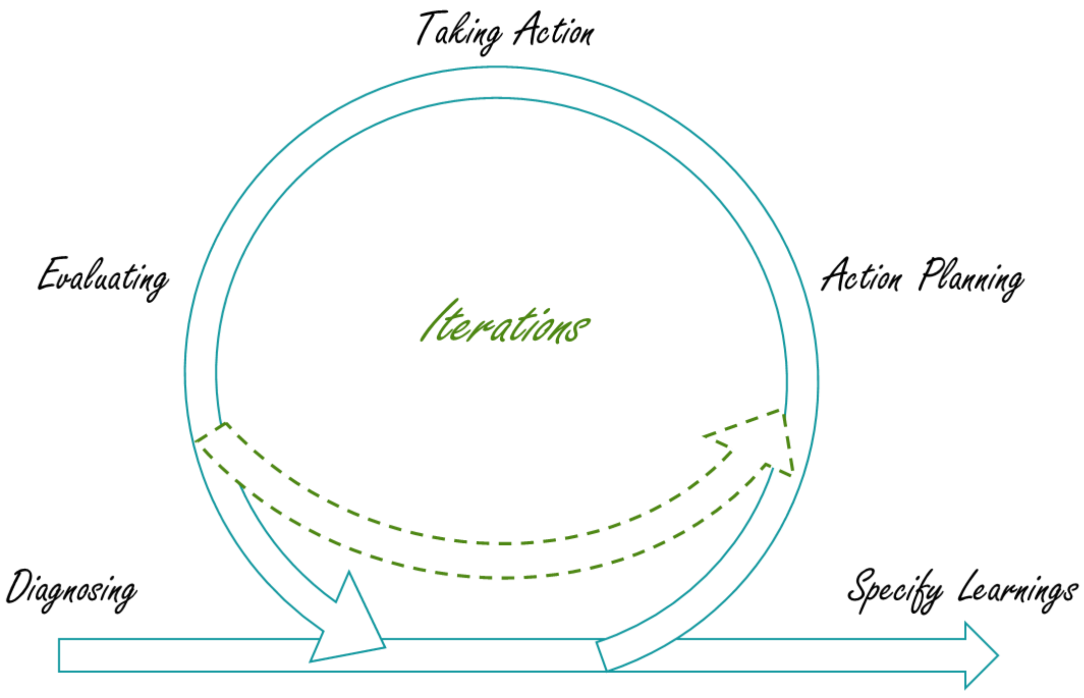


Figure 11 - Action research adapted cycle (Author's elaboration)

During the grooming process development, many evaluations were done, as will be explained in chapter 4.2 Grooming Process. Actually, each iteration had an informal evaluation, which is the reason to have another iteration. However, and even though the team gives continuous feedback, their opinion was not formally collected. Therefore, considering that at the end, only one cycle of the action research was followed, it was decided to collect the team’s feedback officially. After all, they are the final users of all interventions and improvements, and who benefit the most from it. Therefore, a summary of their feedback and learning is presented later on chapter 6,Evaluation and Learnings, and the complete feedback from each member can be found on Appendix I – Feedback from the Cloud team.

3.4 Researcher intervention on the team

Being this action research a participative collaboration, the best way for the researcher to integrate the Cloud team, once the project started in March 2016, was to start as their Scrum Master. In this way, not only some responsibility was removed from the team (the previous scrum master was a full-time developer), but also it was possible to help them to improve and turn the ceremonies more efficient.

One of the main difficulties when integrating a new team for intervening is to get their commitment. For this, since the very beginning, the main goal and intention of the intervention was shared with all the team members, reinforcing that they would be the first ones to be benefited. In order not to destabilize the team, with this sudden intervention, it was decided to make the transition smoothly, as it will be more detailed in the next chapter.

4. Grooming Process Development

The researcher's entrance as scrum master lasted two weeks, taking the effective role only after this period. On the first week, only observations were made, being part of their ceremonies and getting access to the project's files and status, performing minor interventions. On the second week, the researcher started to perform as the team's Scrum Master and to have the responsibilities the role demands but still with the guidance of the former Scrum Master. To conclude, these two weeks were the Diagnosing phase, where it was possible to observe their behaviour as a team, how they followed Scrum, its ceremonies and artefacts, and mainly to identify opportunities to improve. This diagnosis of Cloud team is described in the first section of this chapter (4.1).

The second part (4.2) focus on the main problem, which relates to the one that needed more iterations to achieve a structured solution. This section is also divided for each one of the iterations, which are structured following the phases: Action Planning, Taking Action and Evaluating, as previously explained on Adapted Action Research. At the end, on the last section (4.3) is the complementary work. Here are included other smaller problems the team faced. Note that these ones do not have less importance than the previous, even because they are related to it. Instead they are just smaller, not having more than one iteration, and that is why they were grouped.

4.1 Diagnosing

The Cloud's Product Owner is close to the team, as mentioned before, although that does not mean that the team is perfect and have no problems related to ceremonies or the backlog management. At the beginning, their Product Owner had other duties with other projects, and could not be 100% available for the team. At this point, the user stories for the planning were not very mature, relatively to the customer requirements, which led the team to spend more time in sprint planning meetings, as they always had a lot of questions for the Product Owner regarding the user stories presented by him. Later, during the sprint, the team had the need to contact the Product Owner to clarify some topics related to the user stories. Unfortunately, he was not always available when the team needed, which caused the team's work to stop, hindering them to proceed.

With the uncertain user stories, other problem the team faced was "How can we estimate something not knowing what it implies?". Estimation is a difficulty present in many teams, especially when they start, because the accuracy will be very low and can only be adjusted with time. So, it is possible to find many tools and methods to estimate complexity in literature that the teams can adopt. However, the difficulty and low accuracy at the beginning will always exist, disregarding the method used.

The Cloud team is not an exception. In addition, they did not monitor their velocity carefully. The software they use to manage their work – JIRA[1] – only registers the story points of the completed user stories in each sprint. This means that if the team does not finish a user story, their sprint's velocity is zero story points. Despite this fact being a Scrum rule – 0% or 100% delivery –, in fact, it does not translate the truth, neither the team's reality, as they were actually working.

In truth, this problem is a vicious cycle, more like a snow ball, and illustrated in Figure 12. Since they did not have precise values for team velocity, they did not know how much story points they could complete in a sprint. Because of this, they failed the estimation during planning, which increased the probability of overestimating and not allowing them to complete the sprint, resulting in other velocity data equal to zero story points.

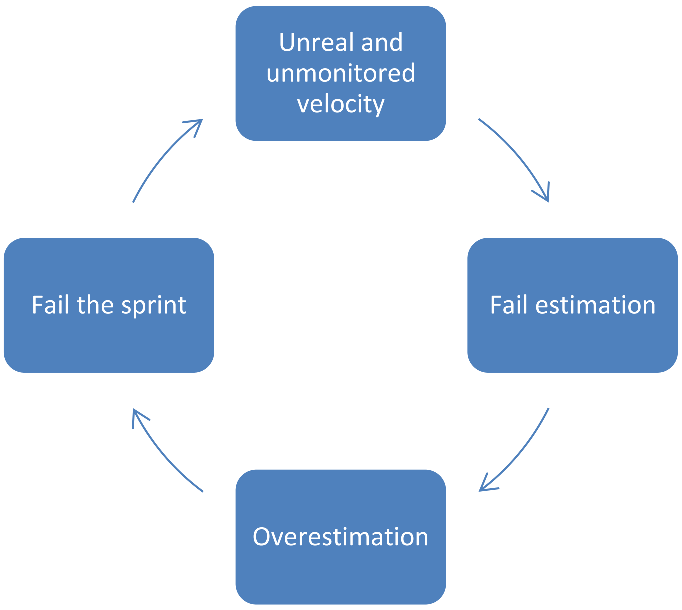


Figure 12 - Cloud team problems cycle

To conclude this velocity topic, the team did not feel comfortable, neither motivated, always having their velocity lower than the work they really performed. So, measures were taken in an attempt to solve the problem.

[1] JIRA – an agile project management tool that supports any agile methodology, including Scrum

The Diagnosing phase exposed more than one problem the team was facing, as listed above. For a better analysis, each of the problems found were separated, in order to develop an action research cycle individually. This approach allowed the focus to be put on one problem at a time, not confusing one with another, and avoiding forgetting or losing track of any problem.
As result of all the observation and analysis taken in the Diagnosing phase, the problems identified and selected as improvement opportunities were: the grooming process implementation, velocity monitoring, lack of management documentation and requirements workflow. The grooming process was considered the main problem, and thus, it was treated separately from the other topics; the remaining problems were grouped and presented as Complementary Work.

4.2 Grooming Process

The grooming process was developed and optimized during 8 sprints with the team. The best considered way to achieve a reliable and tested process, was running the action research iterations in each sprint, fulfilling the iterative characteristic of this method. The team observation started on their sprint 35 with the purpose of only planning an action on the next sprint (sprint 36), to be implemented on the grooming meetings. However, the team showed a high willingness to start the process as soon as possible, so the planning phase ended up by starting on the 35th sprint.
The time plan for the development of the grooming process is illustrated below in Figure 13, for a better visualization and time location of each iteration. It is also visible that the action research cycle was only completely followed once, despite counting with many iterations of the phases: Action Planning, Taking Action and Evaluating. The green marks represent the grooming meetings, when the implementation of the developed process was possible. So, those green marks in Figure 13 represent the Taking Action phase.

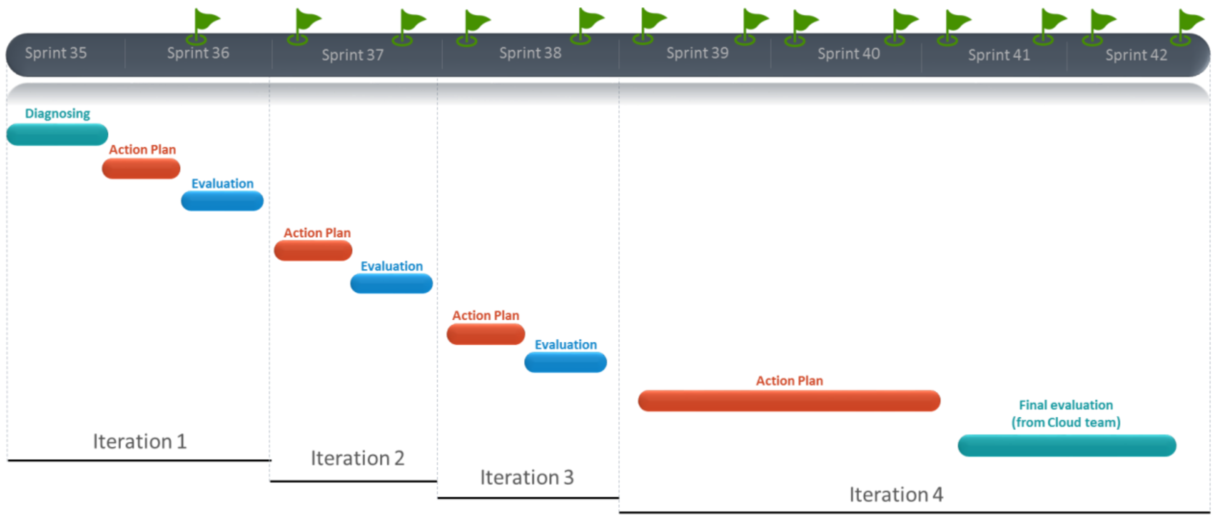


Figure 13 – Timeline of the grooming process development

The processes were modelled in Business Process Modelling Language (BPML), which is one of the languages offered by the non-profit consortium of the computer industry Object Management Group (OMG) (Lima, 2011). According to the Business Process Modeling Notation Specification Version 2.0, this language includes five basics categories of elements: Flow objects, Data, Connecting Objects, Swimlanes and Artefacts (von Rosing, von Scheel, & Scheer, 2011). The elements used to model the processes are described in detail on Appendix III – Business Process Model and Notation.

4.2.1 Iteration 1

Action planning

The first measure the team wanted to embrace for improvement was their involvement during the backlog refinement. Despite not being a ceremony on the Scrum cycle, many authors argue in favour of grooming's effectiveness, as mentioned before on section 2.6. At this point, it revealed to be a good measurement key, as it suits their needs.

The team had already run through the V-model, wherein each phase is a DoD for a user story, at least for the phases that can be included. This means that in each user story they define what is supposed to be done in order to fulfil each phase, including: Mock-up creation, Design/Architecture, Requirements, Implementation, Test and Release Notes. Taking this into account, the development of the mock-up and requirements during the grooming process seemed viable, once in the future the user story could become more *INVEST* when entering the sprint.

Before introducing it to the team, some time to think and develop a model on how the grooming meeting should take place was required, so that the meeting would occur the best way and to have the best outputs from it. The artefacts and the main activities were collected to define the first iteration of the process, as shown below in Figure 14.

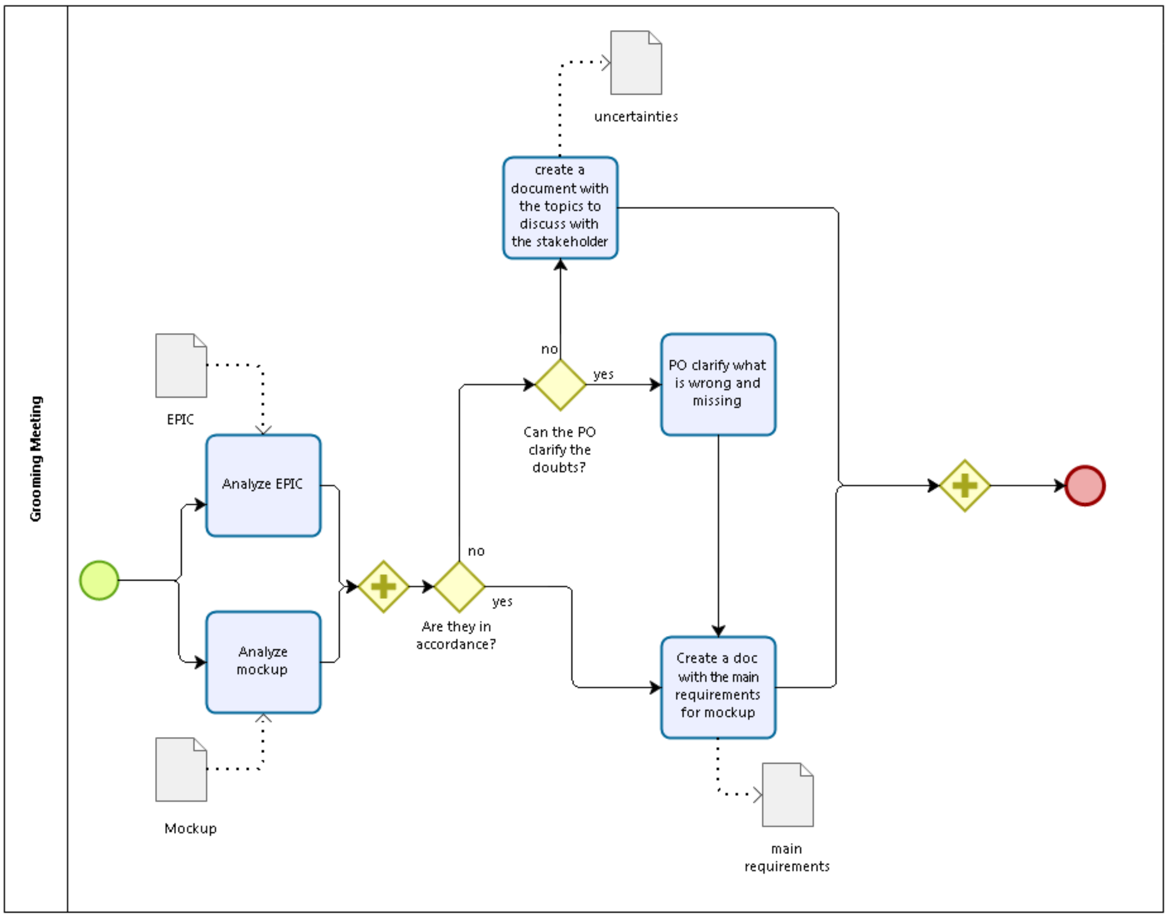

Grooming Meeting
uncertainties
create a document with the topics to discuss with the stakeholder
EPIC
Analyze EPIC
no
yes
Can the PO clarify the doubts?
PO clarify what is wrong and missing
no
yes
Are they in accordance?
Analyze mockup
Create a doc with the main requirements for mockup
Mockup
main requirements

Figure 14 - First Iteration on grooming process for Cloud team

In each sprint, the team should produce a mock-up according to an EPIC, given by the Product Owner, concerning the next feature that the team will develop. The main goal of the grooming meeting is to review the EPIC with the mock-up developed by the team, in order to obtain a valid mock-up that serves as the basis for the implementation. Often, doubts and topics to clarify with the Product Owner arise. However, he will not always be able to solve them all at that time, so it is important that the Product Owner takes notes about it in order to clarify them later with the stakeholders. The product requirements are also an important topic. They should be an output when developing the mock-up and reviewed after the mock-up approval.

The implementation of the grooming process required dedicated effort from the team to the topic. This was not happening in the past but was perceived as an improvement to their effectiveness and efficiency, and also their autonomy.

Taking Action

The grooming process implementation was already planned, when, at sprint 36, the team could not decide if a user story should enter in this sprint, due to their velocity history problems. As an attempt to solve this situation, it was a good opportunity to introduce the grooming process, in order to have a better commitment from them and to capture their involvement, which is one of the main goals; how the implementation ran and its benefits are explained below.

Grooming benefits

Sharing the vision of the process and the benefits that it could bring for the team was the main objective when introducing the grooming process. The main advantage for the team that this process could bring is the improvement of user stories. Since the mock-up is already developed, and the requirements already discussed, the user story will be smaller once it will cover less DoD when matching with the V-model phases. With this better definition of the user stories, it is clearer what the product is supposed to do and not to do, decreasing the doubts during the sprint planning. This certainly improves the team estimation, also contributing in the future to establish and stabilize a real team velocity.

Grooming serves as a preparation for the next sprints, with the development of mock-ups and progress of requirements. Considering the fact that it consumes team's effort, and in order to formalize their commitment in developing those topics, every sprint should have a user story for grooming, where it is described, what, why and how they are supposed to deliver a product increment.

First grooming meeting

At the beginning, it was decided to only involve one team member on the process, besides the Scrum Master, once there was no need to spend too much resource capacity, and also in an attempt not to interfere with team performance.

The first grooming meeting went as expected, following all the process' activities. However, it started with some constraints. It was not so well prepared once the team member forgot to share the material in a folder that everyone had access to before, which delayed the meeting on trying

to solve the problem. During the discussion on what was presented, many topics arose for new user stories, mock-up changes, and possible requirements. However, this last topic was not specifically discussed and so did not demand an update of the requirements document.
As an output of the meeting, it was possible to have an alignment, between the PO and the team member, on how the next user stories should be created by the team member.

Evaluating

After the meeting, a report was made and shared with the team, fostering transparency, one of Scrum pillars, and leading by example. The process was then reviewed step by step in order to find possible improvements in activities, artefacts involved and their sequence, and the structure/design.
Regarding activities and artefacts, the first one – Analyse EPIC – needed changes, because not always are there epics to compare, but also because they can be customer requirements not yet matured enough by the Product Owner. One thing that is always an input for the process is the user story specially created for the current grooming. As mentioned, the requirements document was not created neither updated, but instead, topics for that were collected. This way, and closer to the meeting, the appropriate requirements should be developed and documented. It is also important to note that most of the times it will be necessary to update the mock-up, thus being important that notes are taken, in order not to miss significant information.
At last, the process showed some structure problems, once the activities should happen between parallel gateways, to force them both to be performed – Analyse EPIC and Analyse Mock-up – and concluded to proceed with the process, which did not reflect the reality.

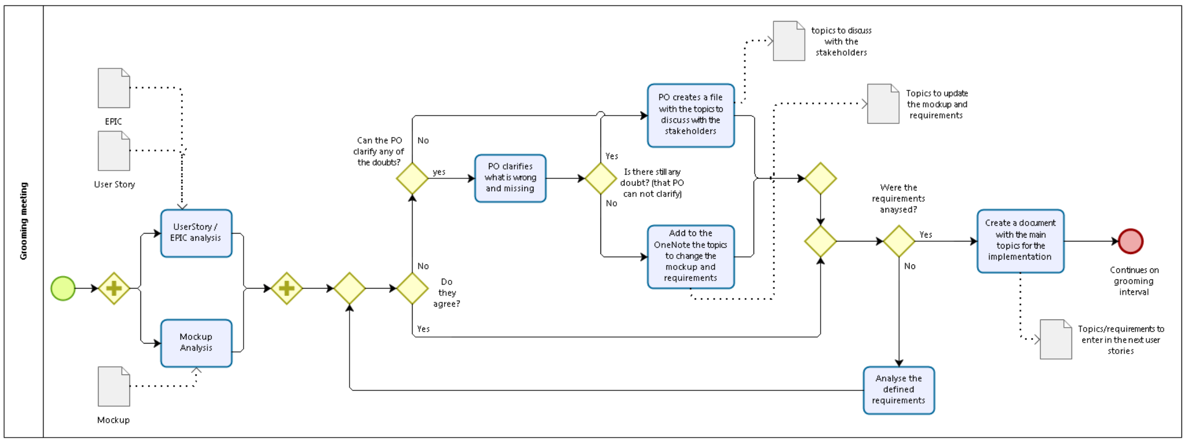
4.2.2 Iteration 2

Action Planning

The second iteration on the grooming process occurred after the evaluation of the previous one. The outputs of that phase were good inputs for a new iteration, as they provided specific topics to act upon. The topics regarding the structure were the easiest to change, because the parallel gateways were wrongly positioned.
The user story concerning the sprint grooming was added at the beginning of the process between the parallel gateways, forcing the comparison between the artefacts: the epic involved, the user story purposely created, and the mock-up developed.

The clarification of doubts is a cyclic process, and there might be some doubts that the PO clarifies, but others he does not, so it is important to have this described in the process. Other issue that was not included on the previous model is a document containing the description of topics related to changes on the mock-up, considering there are always inputs from the Product Owner to improve it. In this document, topics related to the requirements could also be included. If they were not discussed before, during the mock-up exposition, then they should be analysed to get further feedback from the Product Owner.

Finally, to conclude the meeting, a document or a list should be created with the main topics that will compose the next user stories for the backlog. This process is illustrated below in Figure 15.



Grooming meeting
EPIC
User Story
Mockup
UserStory / EPIC analysis
Mockup Analysis
Can the PO clarify any of the doubts?
No
yes
PO clarifies what is wrong and missing
Do they agree?
No
Yes
Yes
Is there still any doubt? (that PO can not clarify)
No
PO creates a file with the topics to discuss with the stakeholders
topics to discuss with the stakeholders
Add to the OneNote the topics to change the mockup and requirements
Topics to update the mockup and requirements
Were the requirements anaysed?
Yes
No
Create a document with the main topics for the implementation
Continues on grooming interval
Topics/requirements to enter in the next user stories
Analyse the defined requirements

Figure 15 - Second Iteration on grooming process for Cloud team

Taking Action

At sprint 37, there was an update on the process to be followed during the meeting. For this new iteration, it was decided to involve another team member, still ensuring that all team members participate directly in the process. Moreover, maintaining the first and inviting another one enables the sharing and guidance through the learning process. It also forces all the members to work with all the engineering processes that it involves: development of mock-ups and requirements, allowing them to have contact in different abstraction levels of software engineering.

First grooming meeting of sprint 37

The second overall grooming meeting, and the first on sprint 37, was better than the previous one. The model was taken for the meeting in order to conduct it through the planned activities. At the beginning, the mock-up was presented and compared to what was expected, as described in the grooming's user story. The team members also had prepared some requirements, which were analysed not only regarding their content but also the way the writing was implemented. At the end, the outputs were topics to change the mock-up, and also topics to build new user stories related to the current feature. At this point, it did not seem very helpful to end the sprint without ending the grooming process. To solve this situation, a new grooming meeting was scheduled, allowing the team to share the modifications done, based on the outputs of the first meeting. So, during this interval between the two meetings, the team was expected to update the mock-up and requirements, and to create new user stories to be evaluated and groomed on the next meeting.

Second grooming meeting of sprint 37

At the beginning of the meeting, the topics that the team committed to develop since the last meeting were shared. For each one of them, the work done was presented and as expected, other feedback and topics started to come up as well. Some of them were doubts and uncertainties, being the Product Owner responsible for taking notes. Moreover, it was important that he compromised in scheduling a meeting with the client, so that he could clarify those doubts and uncertainties.

In addition, it was also of great relevance to settle the next topic to be groomed, which usually is another feature related to the current one.

Evaluating

As it can be observed, many progressions and changes were made since the first iteration on the previous sprint. Another team member was included to participate and another meeting was performed in the sprint. The purpose of this phase is to really close the topic with user stories for the PB, ready to enter in a sprint. Having the mock-ups and requirements defined and described, the implementation in the future will be much facilitated. Once these updates are external to the model, a new process was not developed at this stage. Therefore, another sprint was held, still following this model, with the intention to identify lacks and improvements to develop a new process later.

First grooming meeting of sprint 38

For the current sprint, it was decided in the last grooming meeting to study a new feature that had not been discussed by then. For this new feature, the team developed three alternative models and presented them in the meeting to get feedback from the Product Owner. At this point, the most user-friendly model was chosen and, since the topic was still “raw”, modifications were pointed out to be updated later, in order to be useful in the future, by the time of implementation.

Second grooming meeting of sprint 38

Continuing with the previous feature, this meeting had the purpose to settle the mock-up. This time, the studied feature was very generic, which involved many other features. Those other features were then identified when evaluating the mock-up. Even from the functionalities already studied or implemented, new updates and doubts started being raised. To get around this situation, the team decided that in the next sprint they should spend their grooming effort in grooming the new features found, developing epics and user stories related to them and to the other feature already developed.

Evaluating

In general, the process seemed complete and its scope accomplished, which is to analyse the mock-up developed by the team to enlarge the backlog, as well as increasing their quality with more INVEST user stories. However, the models presented some problems on its structure that could be improved. At first, the trigger was not identified to start the process, more specifically, what was the start of the process. Moreover, the current model is forcing the analysis of requirements, and consequently its management, which not always happens. They can only be addressed, if those involved feel the need to, but it is not mandatory. Besides, the requirements do not have a good workflow, because topics for their change are included with the mock-up changes, but there is not a connection between the two when later analysis is held.

4.2.3 Iteration 3

Action Planning

Concerning the grooming meeting process, changes were made, starting by separating the mock-up review from the user story analysis. Because, in fact, at the beginning of the meeting, the team is free to present what they developed and only later it is verified if all acceptance criteria of the selected feature were fulfilled.

The requirements continue to be presented in the model, but this time, as not mandatory. On the contrary, they can be analysed if those present want to. But, if they do not, the process will not be blocked, allowing its normal flow. So, if the requirement development and monitoring does not occur, then there is no problem at all. Because, in fact, it is an engineering process, so they will be developed and matured during the sprint, at the feature implementation.

In the two last sprints, it was crucial to have two grooming meetings, to continue the topics addressed in the first one, in order to close them properly. It is coherent that if a mock-up is not completely settled on the first approach; it is necessary to review it once again before its closure. So, between the activities of developing the mock-up and creating user stories, the grooming meeting should occur as many times as the team feels the need to, until the mock-up is settled to create reliable user stories. All this description of the process can be visualized below in Figure 16.

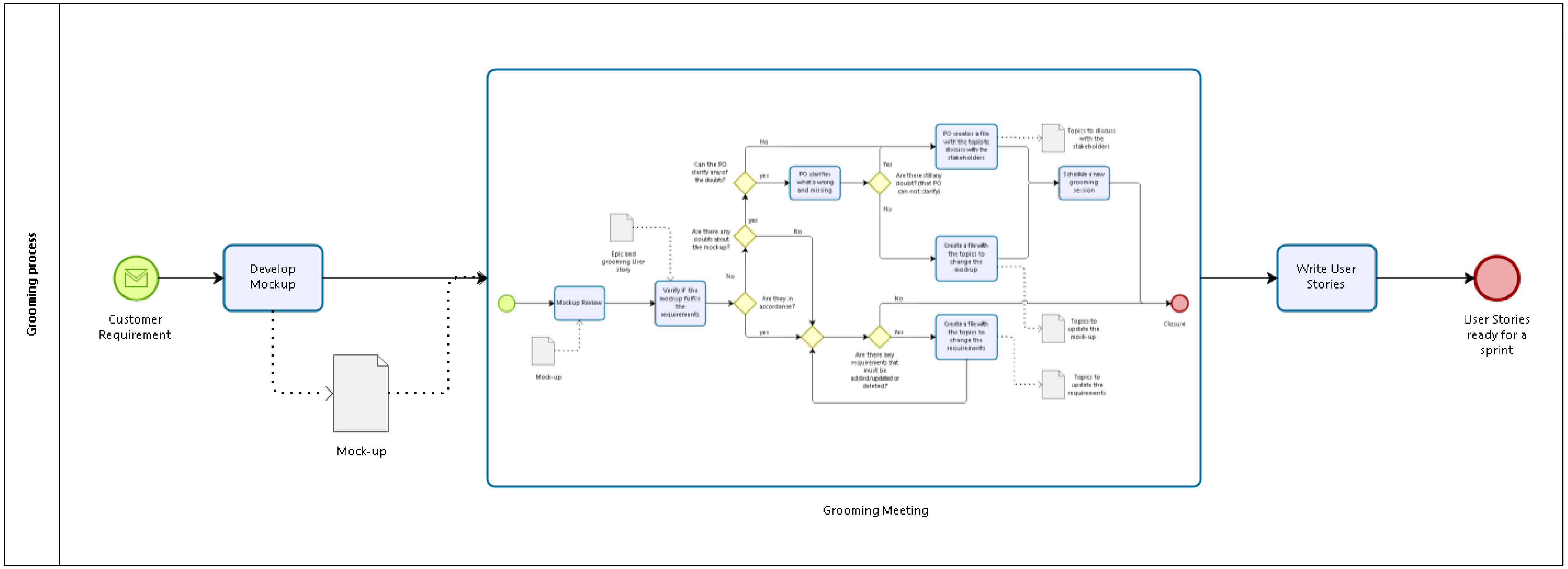

Grooming process
Customer Requirement
Develop Mockup
Mock-up
Mockup Review
Mock-up
Epic and grooming User story
Verify if the mockup fulfils the requirements
Are they in accordance?
No
yes
Are there any doubts about the mockup?
yes
No
Can the PO clarify any of the doubts?
yes
No
PO clarifies what is wrong and missing
Are there still any doubt? (that PO can not clarify)
Yes
No
PO creates a file with the topics to discuss with the stakeholders
Topics to discuss with the stakeholders
Schedule a new grooming session
Create a file with the topics to change the mockup
Topics to update the mock-up
Are there any requirements that must be added/updated or deleted?
Yes
No
Create a file with the topics to change the requirements
Topics to update the requirements
Closure
Grooming Meeting
Write User Stories
User Stories ready for a sprint

Figure 16 –Third iteration on grooming process for Cloud team

Taking Action

The purpose of the sprint, when it was decided to implement the process, was to enlarge the PB. Once on the last grooming, many features came up together with the mock-up, the decision of narrowing the work to the product backlog, and not on a new feature, was unanimous between the team members. So, the real grooming process was not completely implemented, from the beginning to the user stories, in a sprint, as planned in the first iteration. Quite the opposite, the grooming meetings were used to share and discuss the user stories created out of the meeting. This way, they could improve the quality of the user stories, which were being reviewed with the Product Owner.

First grooming meeting of sprint 39

In the first grooming meeting of the 39th sprint, the team members responsible for the grooming presented their work, more specifically the user stories they committed to develop from the previous sprint. Therefore, they shared the epics and the user stories related to the features and mock-up discussed on the previous sprint. As expected, many doubts were clarified and suggestions to change them were also made, as five user stories were presented. At this point, all the involved agreed that there should be one more grooming meeting to settle and confirm the modifications.

Evaluating

At this point of the process development, it is visible the necessity to change the abstraction for a higher level. In fact, this last iteration already addressed other activities outside the grooming meeting. Nevertheless, the grooming process as a whole, is not completely defined considering the main goal of this dissertation. So, in section 5.1, the last iteration of the grooming process will be presented in more detail.

4.3 Complementary Work

At the beginning of the chapter, in the Diagnosing phase, the sub-chapters organization was revealed. As mentioned there, the current Complementary Work contains the Velocity Monitoring, Requirements Workflow and Lack of Management Documentation issues. The action research cycle was run in all these problems but their description is briefer for simplicity.

4.3.1 Velocity Monitoring

The velocity monitoring was a problem mentioned by the team more than once. The fact that the user stories can only be delivered if completed, forces them to be 0 or 100% achieved. Once the team is junior, and is facing an innovation project, they were led to unpredictable tasks which had impact on their velocity. Many times, the team worked hard in a user story but they just could not complete it, contributing with a 0 for their velocity. Also, the software they use to monitor their work, only allows to deliver the user story if completely closed, i.e. with all the tasks done on their virtual Kanban board. So, it does not monitor a real velocity value, as sometimes the goal and the user's story scope were fulfilled, but smaller tasks were not completed. This situation contributed to the demotivation of the team, and also for a not so useful historical data of their velocity. Not only was it more difficult for the team to estimate, but also for the Product Owner to plan a release, becoming increasingly unpredictable.
In addition, during the sprint, the software used by the team only monitored the hours remaining to end the sprint and not the real story points that needed to be completed. This missing information was also a problem that the team shared, in order to find alternative solutions.

At sprint planning, and after they commit with the sprint backlog, the team defines which phases of the v-model they will develop in each committed user story. This way of describing the DoD allows them to specifically interact with the software processes engineering.
In order to improve their planning, it was unanimously decided to distribute the story points of a user story for its DoD. Thus, they were able to know how many story points they completed, even if the user story was not accomplished. Furthermore, the grooming process itself will improve the user stories, facilitating the planning for the team, as said before, and helping them to find their real and optimum velocity.
The Scrum Master role is essential at this point because, not only is he responsible for removing impediments but also, and even more important, responsible for improving the team's efficiency. Keeping relevant information like the team's velocity is vital, assuring the team's stability and, moreover, protecting them when the Product Owner demands more from them than they can afford.

4.3.2 Requirements Workflow

"Requirements are the basis for every project" (Hull, Jackson, & Dick, 2006), defining what the stakeholders want the new system to perform. There is an entire workflow since the requirements are delivered by the stakeholder until each of them is implemented and tested.

However, until now, the team did not have a defined requirements workflow to follow. The only stages that they followed were the two defined by default by JIRA: *To Do* and *Done*. With this, the team only knew if it was created – *To Do* –, even if it had not been defined, approved, implemented or tested – *Done*. All the stages between first addressing a requirement and its closure, were not defined, which made the work a lot more difficult for the team.

In addition, the team deals with requirements in some way, so without an accurate workflow and management, each team member will have different interpretations and perceptions of the requirements' status, taking the risk of implementing raw requirements that were not approved, for example. Also, the definition of the status *Done* might have different meanings for each member; for some it could be a requirement that was implemented, while for others it could be an already tested requirement. This situation leads them to avoid requirements management, which may create big problems in the project's future.

As Figure 17 demonstrates, a bad requirements' definition can have a huge impact on the projects failure, because the product delivered to the customer cannot fulfil what he requested at the beginning. Instead, it could be quite the opposite.

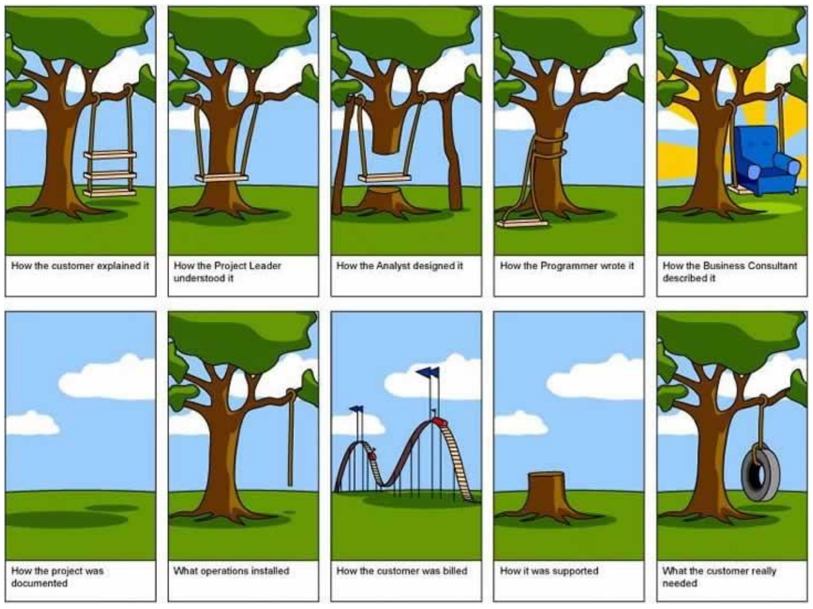

How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Figure 17 – When the requirements defined by the team do not meet the customer's
(Mifsud, 2013)

4.3.3 Lack of Management Documentation

As it was demonstrated, the lack of management documentation has many implications on the team's efficiency, as they did not have baselines to guide themselves. Furthermore, not only did they not have all the management documentation updated, but also, they had put it somewhere else and forgotten about it in some occasions. At this point, tracking becomes more difficult, which becomes an impediment to improve with the lessons learned.

The sprint retrospective was the main ceremony in which they did not have any tracking. Only one of the members, the Scrum Master, remembered where it was reported, but he also empathizes with the team, not having the curiosity to go there and check what problems they faced or whether they were solved. This means that even if the team had defined an improvement measure, at the end of a sprint retrospective, the evaluation of its application was never done.

5. RESULTS

At this chapter, the results and improvement measures obtained during the research and described in the previous section are exposed. This section is divided in two, each corresponding to the problems the team faced. The intention with this is to ease reading and remain aligned with the previous section, so that results could be more easily understood. The first section includes the last iteration on the grooming process and the main benefits while the second addresses the complementary work in the same format.

5.1 Grooming Process

At this point of the process's development, and considering that the last iteration did not reflect the complete process, it was a good opportunity to elevate its level. At this moment, it made sense to develop the entire grooming process and not only the meeting description. The grooming process was incomplete according as it only modelled how the grooming meeting should be performed. So, the underlying work that is developed outside the meeting was not fully visible, even after iteration 3 in Figure 16. This limitation, despite not showing the real work performed specifically, does not allow the team and the stakeholders to have the macro view of the process. Or, even if the team members had this view in their minds, they cannot follow it correctly, once it is not clearly defined nor modelled to be followed precisely. Therefore, the need to develop the whole grooming process emerged, from the breakdown of customer requirements until the PB, fulfilled with user stories ready to enter in a sprint.
Together with the Product Owner, and after a brainstorm, it was possible to define a process with all the knowledge collected at this point. The result is illustrated in the activities diagram of Figure 18.

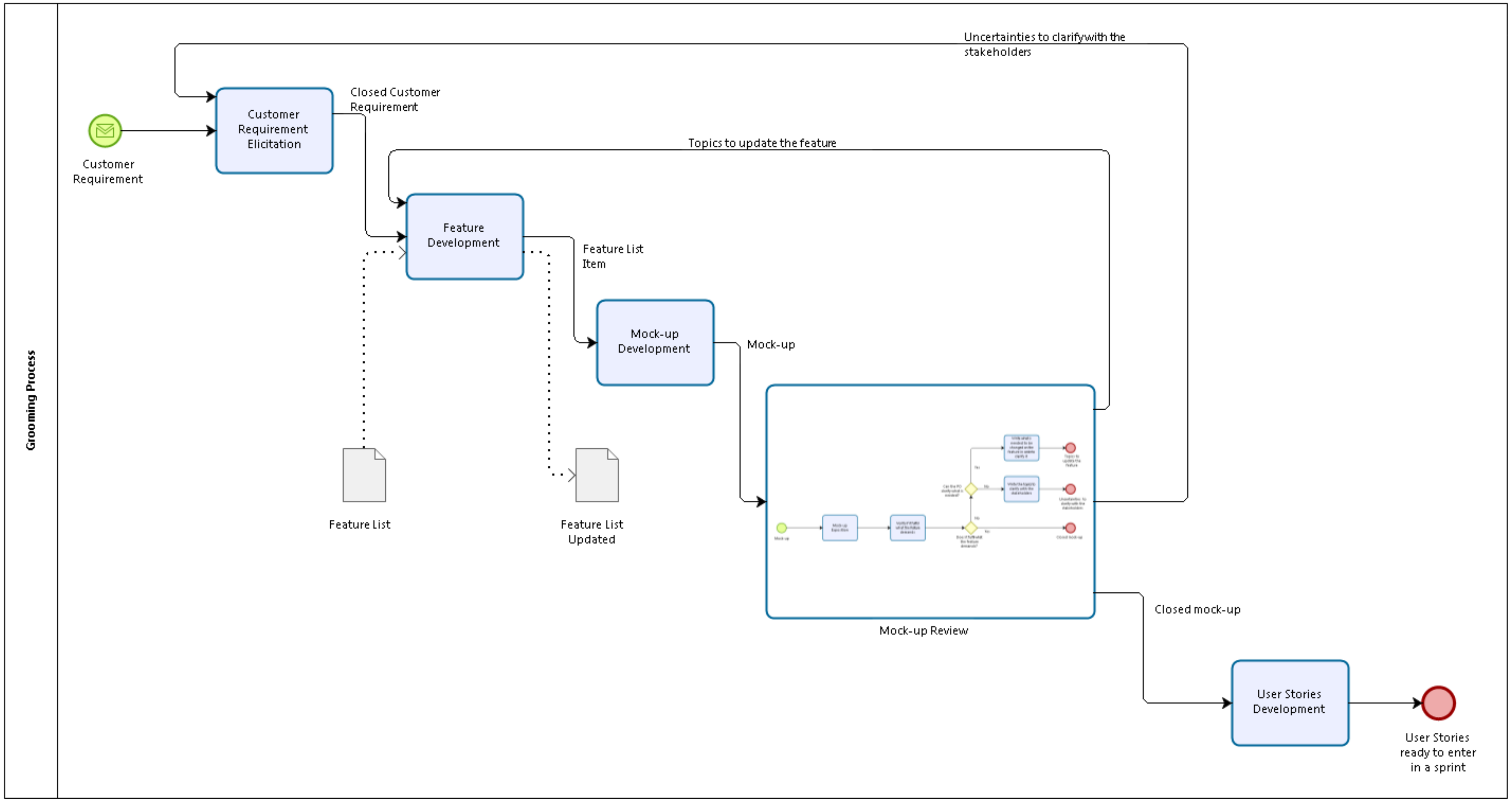


Figure 18 – Activities diagram of the grooming process

The *Customer Requirement* (CR) is the trigger to start the process, and it is usually sent as a message, e.g. by e-mail. The first stage is the Customer Requirement Elicitation, where the team and mainly the Product Owner discuss with the stakeholders, deepening the comprehension over their necessities. This is a decisive phase, where the expectations for both parts are aligned and managed. The output is the closed customer requirement, ready to continue the process and go to the Feature Development. Here, the *Closed Customer Requirement* is matured to be converted in features, specific functionalities that will be added, or updated, to the *Product Feature List*. Usually, the responsible for this activity is the Product Owner, but the team can also participate. One of the features - a *Feature List Item* - is the input; the team has to develop a mock-up, using Balsamic software. This mock-up is the input for the Mock-up Review, more precisely the grooming meeting discussed earlier, where the output is the mock-up reviewed, if agreed. However, and as mentioned before, it can go back for previous stages if it does not fulfil what was expected, so that a new mock-up is developed. Only when it achieves the goal of the features, and accommodates all demands, the mock-up proceeds as reviewed for the User Stories Development. Here, the user stories are created and matured, in order to fit in a sprint. The team already uses the JIRA software to manage their sprints, so the user stories are created there to avoid too much documentation.

The mock-up review, illustrated on Figure 19, is nothing more than the grooming meeting process described on the previous section 4.2 Grooming Process, but now more adapted to fit in the activities diagram that describe the whole grooming process. To be as compliant as possible, its exit points match exactly with the exit outputs, in arrow format, of the sub-process Mock-up Review of Figure 18. In the same way, the initial point of this sub-process is the same as the input illustrated in Figure 18. The flow of how this grooming meeting to review the mock-ups should be performed is represented below in Figure 19.

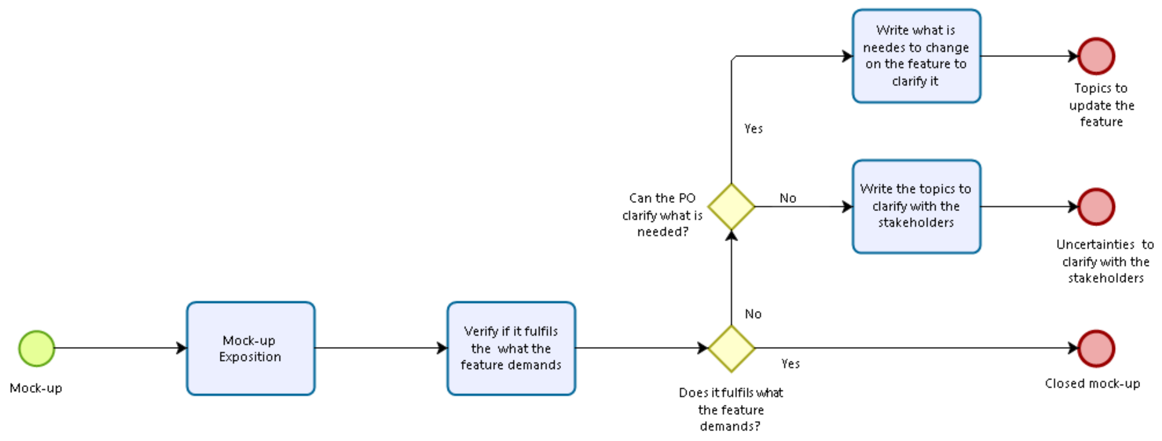

Write what is needes to change on the feature to clarify it
Topics to update the feature
Yes
Can the PO clarify what is needed?
No
Write the topics to clarify with the stakeholders
Uncertainties to clarify with the stakeholders
No
Mock-up Exposition
Verify if it fulfils the what the feature demands
Yes
Closed mock-up
Mock-up
Does it fulfils what the feature demands?

Figure 19 - Mock-up review meeting workflow

On the mock-up review meeting, three exits are possible. The ideal is to end the meeting with a *Closed Mock-up*, reviewed and settled to proceed with the process. However, and as mentioned most of the times, there are updates to be made or improved, which does not represent a problem. As Figure 19 shows, it is possible to have the Product Owner's feedback and just update the feature and the mock-up, returning then to the Feature Development to build more accurate and agreed features. But, even in the worst-case scenario, when this meeting is not sufficient, it is also possible to return to the Requirements Elicitation activity to discuss with the stakeholders what is needed. Thus, there is no limit for the frequency the mock-up should be iterated. The purpose is to have a reliable mock-up that fulfils what the feature demands, no matter the number of iterations required.

Benefits for the team

The benefits for the team are not less important. On the contrary, they could be significant to the project. As mentioned on the Diagnosing phase, the lack of commitment of the team was visible and this process came also as an attempt to solve it. Now, they could finally have a voice when a decision needs to be made, also being more involved on long-term planning. This gives them more autonomy, being visible that they became more committed and compromised with the project. Concerning to real work, this process is a huge help once it guarantees the team follows correctly an important part of the engineering process. Also, it gives them more awareness and knowledge about the functionalities before their implementation, allowing them to better predict effort and complexity at the sprint planning meetings.

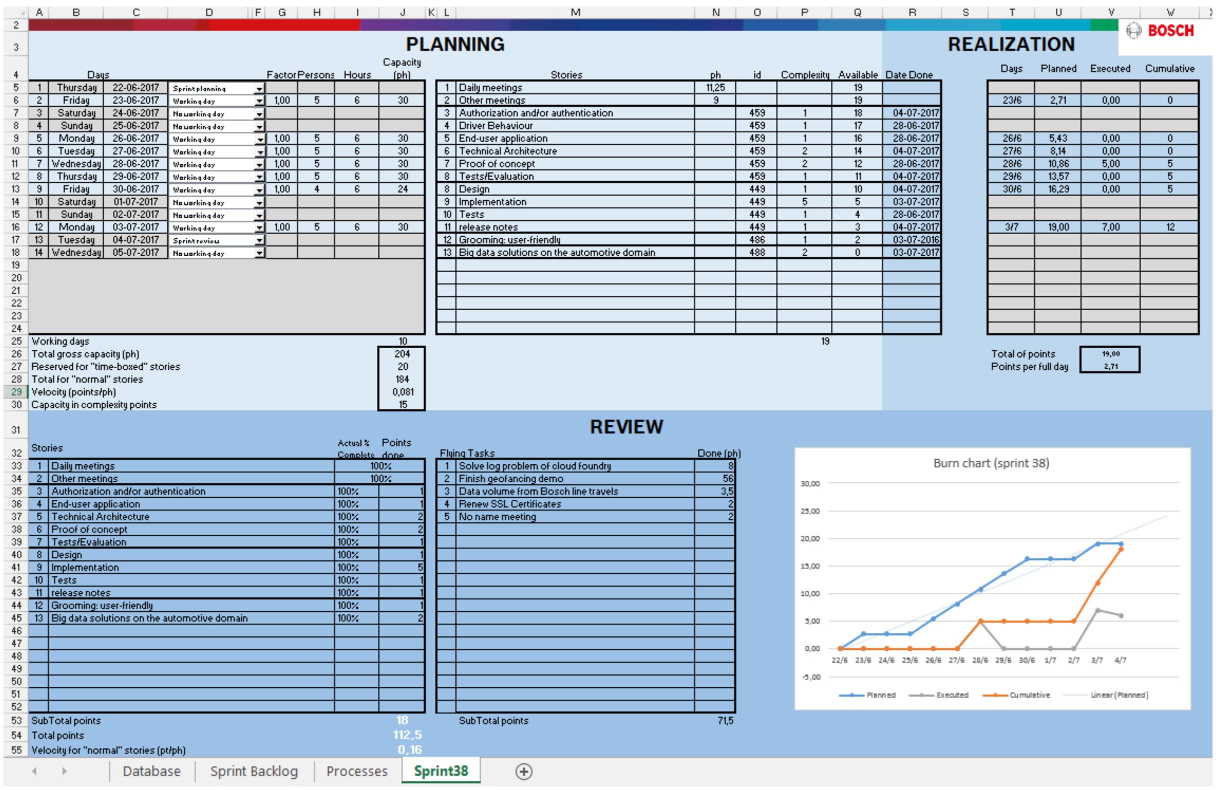
5.2 Complementary work

5.2.1 Velocity Monitoring

According to the diagnostic and planned actions discussed on the previous section, the team decided to distribute the user story's points for the phases that it includes, to vote more granularly. This distribution does not need to be equal for each phase. On the contrary, the team should attribute more story points to the phase which will demand more effort.

To have the perception on how the work is running during the sprint, they defined as team rule to have the burn-up chart, of the sprint, pinned on their Kanban board. The purpose is to have a baseline, and update the points completed every daily meeting. This way, it constitutes a trigger for the team, and mainly the SM, when they are not evolving, to know if the team is facing some problem that is blocking them to proceed their work.

With the purpose of improving all topics related to the team's velocity, it was developed an Excel template for the SM to monitor each sprint. This is distributed in 3 different areas, comprising Planning, Realization and Review as an example. Figure 20 shows the Excel spreadsheet for sprint 38.



PLANNING

	Days			Factor	Persons	Hours	Capacity (ph)
1	Thursday	22-06-2017	Sprint planning				
2	Friday	23-06-2017	Working day	1,00	5	6	30
3	Saturday	24-06-2017	No working day				
4	Sunday	25-06-2017	No working day				
5	Monday	26-06-2017	Working day	1,00	5	6	30
6	Tuesday	27-06-2017	Working day	1,00	5	6	30
7	Wednesday	28-06-2017	Working day	1,00	5	6	30
8	Thursday	29-06-2017	Working day	1,00	5	6	30
9	Friday	30-06-2017	Working day	1,00	4	6	24
10	Saturday	01-07-2017	No working day				
11	Sunday	02-07-2017	No working day				
12	Monday	03-07-2017	Working day	1,00	5	6	30
13	Tuesday	04-07-2017	Sprint review				
14	Wednesday	05-07-2017	No working day				

Working days	10
Total gross capacity (ph)	204
Reserved for "time-boxed" stories	20
Total for "normal" stories	184
Velocity (points/ph)	0,081
Capacity in complexity points	15

	Stories	ph	id	Complexity	Available	Date Done
1	Daily meetings	11,25			19	
2	Other meetings	9			19	
3	Authorization and/or authentication		459	1	18	04-07-2017
4	Driver Behaviour		459	1	17	28-06-2017
5	End-user application		459	1	16	28-06-2017
6	Technical Architecture		459	2	14	04-07-2017
7	Proof of concept		459	2	12	28-06-2017
8	Tests/Evaluation		459	1	11	04-07-2017
8	Design		449	1	10	04-07-2017
9	Implementation		449	5	5	03-07-2017
10	Tests		449	1	4	28-06-2017
11	release notes		449	1	3	04-07-2017
12	Grooming: user-friendly		486	1	2	03-07-2016
13	Big data solutions on the automotive domain		488	2	0	03-07-2017
				19		

REALIZATION

Days	Planned	Executed	Cumulative
23/6	2,71	0,00	0
26/6	5,43	0,00	0
27/6	8,14	0,00	0
28/6	10,86	5,00	5
29/6	13,57	0,00	5
30/6	16,29	0,00	5
3/7	19,00	7,00	12

Total of points	19,00
Points per full day	2,71

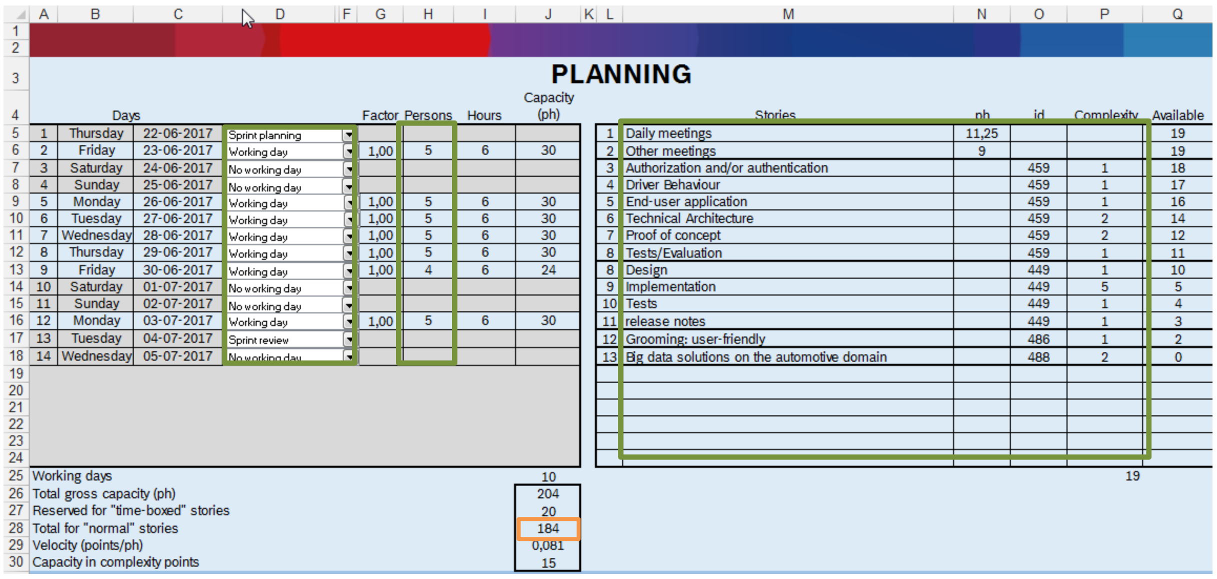
REVIEW

	Stories	Actual % Complete	Points done
1	Daily meetings	100%	
2	Other meetings	100%	
3	Authorization and/or authentication	100%	1
4	End-user application	100%	1
5	Technical Architecture	100%	2
6	Proof of concept	100%	2
7	Tests/Evaluation	100%	1
8	Design	100%	1
9	Implementation	100%	5
10	Tests	100%	1
11	release notes	100%	1
12	Grooming: user-friendly	100%	1
13	Big data solutions on the automotive domain	100%	2
	SubTotal points		18
	Total points		112,5
	Velocity for "normal" stories (pt/ph)		0,16

	Flying Tasks	Done (ph)
1	Solve log problem of cloud foundry	8
2	Finish geofancing demo	56
3	Data volume from Bosch line travels	3,5
4	Renew SSL Certificates	2
5	No name meeting	2
	SubTotal points	71,5

Figure 20- Monitoring template from Cloud' sprint 38

Going into more detail, Figure 21 shows a zoomed-in area of this spreadsheet. Cells in green are those the SM updates on sprint planning. On the left table, the SM updated the days the team would work and the absences to determine the team's availability. The DoD of each user story are included on the right table with each user story ID and story points. The user stories descriptions are on the *Sprint Backlog* Excel tab, and its DoD voted on *Process* Excel tab (once they concern to the software engineering processes), as visible on Figure 20.



PLANNING

	Days			Factor	Persons	Hours	Capacity (ph)
1	Thursday	22-06-2017	Sprint planning				
2	Friday	23-06-2017	Working day	1,00	5	6	30
3	Saturday	24-06-2017	No working day				
4	Sunday	25-06-2017	No working day				
5	Monday	26-06-2017	Working day	1,00	5	6	30
6	Tuesday	27-06-2017	Working day	1,00	5	6	30
7	Wednesday	28-06-2017	Working day	1,00	5	6	30
8	Thursday	29-06-2017	Working day	1,00	5	6	30
9	Friday	30-06-2017	Working day	1,00	4	6	24
10	Saturday	01-07-2017	No working day				
11	Sunday	02-07-2017	No working day				
12	Monday	03-07-2017	Working day	1,00	5	6	30
13	Tuesday	04-07-2017	Sprint review				
14	Wednesday	05-07-2017	No working day				

Working days	10
Total gross capacity (ph)	204
Reserved for "time-boxed" stories	20
Total for "normal" stories	184
Velocity (points/ph)	0,081
Capacity in complexity points	15

	Stories	ph	id	Complexity	Available
1	Daily meetings	11,25			19
2	Other meetings	9			19
3	Authorization and/or authentication		459	1	18
4	Driver Behaviour		459	1	17
5	End-user application		459	1	16
6	Technical Architecture		459	2	14
7	Proof of concept		459	2	12
8	Tests/Evaluation		459	1	11
8	Design		449	1	10
9	Implementation		449	5	5
10	Tests		449	1	4
11	release notes		449	1	3
12	Grooming: user-friendly		486	1	2
13	Big data solutions on the automotive domain		488	2	0
				19	

Figure 21 - Planning area from monitoring template, Cloud' sprint 38

With the previous velocity data, marked as orange on Figure 21, and with the total hours, it is possible to know how many story points they will be capable of delivering, as Equation (1) demonstrates. In sprint planning meetings, this should be the stop criterion, when the team stops committing with user stories.

$$\textit{Story points the team can commit } (\boldsymbol{sp}) = \textit{history velocity } (\boldsymbol{sp/h}) \times \textit{ total planned hours}(\boldsymbol{h}) \qquad (1)$$

With the team's capacity and the story points defined for the sprint, a burn-up chart is automatically built, as it is visible in the lower right corner of Figure 20.

At the end of the sprint the template should be completed, with the date in which each DoD was completed and with the flying tasks the team had. At this point, new velocity data, the *Real sprint velocity* is now calculated. This more accurate velocity is obtained with the story points completed and the time the team had to complete them. This duration is the capacity determined in sprint planning, for the sprint, less the time spent in flying tasks. So, velocity in story points per hour is obtained as Equation (2).

$$Real\ sprint\ velocity\ (sp/h) = \frac{story\ points\ done\ (sp)}{(total\ planned\ hours - total\ hours\ pent\ in\ flying\ tasks)\ (h)} \tag{2}$$

At the end of the sprint, the burn-up chart is also updated with the story points completed, in order to evaluate the team's velocity and performance during the sprint, obtaining a chart as Figure 22 below.

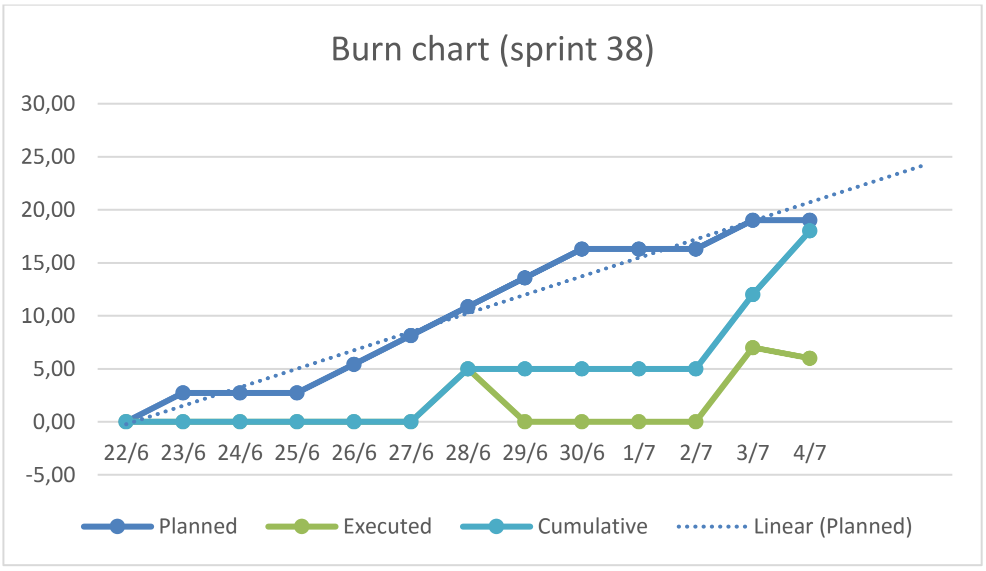


Figure 22 - Burn-chart results from Cloud team at sprint 38

The best alternative found, as mentioned before, to allow the team to monitor their work was to create a burn up chart and pin it on their Kanban board, as illustrated on Figure 23. Its purpose is for the team to update it, in each daily meeting, with the completed story points concerning to the finished DoD. It has the baseline drawn in it based on the data inserted on the Scrum Master's Monitoring Excel, so they could compare with reality.

This visual tool works as a trigger for the team once they have a visual control on the work progress, easily having the perception if at the middle of the sprint the team already completed half of their work.

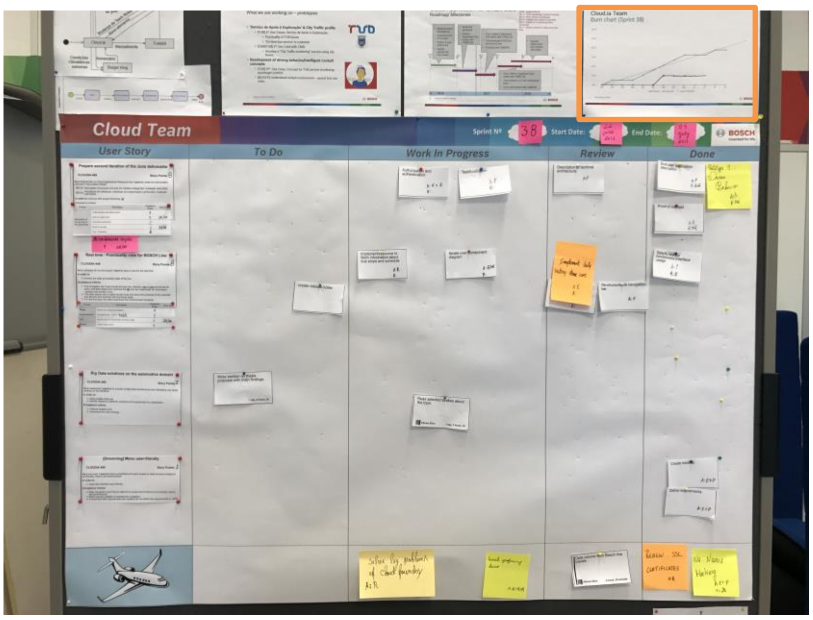


Figure 23 - Cloud's Kanban board during the print 38

For the SM to have the overview of the team's performance, the main data related to their velocity and efficiency were included in a new Excel file, as demonstrated on Table 1. At the end of each sprint, it is expectable that the SM will update the file. The Average Velocity for each sprint is determined with all the previous velocity data, so this value includes more historical data to be increasingly accurate and realistic. This is the value that will be inserted on each sprint planning on the orange cell marked on Figure 21.

Table 1- Sprints performance of Cloud team.

Sprint	Planning			Review				Average Velocity (Story points /
	Planned working hours	Planned Story points	Planned Velocity in story points/ hour	Flying tasks hours	Real working hours	Real Story Points	Real Velocity story points/ hour	
35	236	21	0,089	5,5	230,5	21	0,091	**0,091**
36	196	20	0,102	33	163	11	0,067	**0,079**
37	216	21	0,097	21,3	194,7	17	0,087	**0,082**
38	220	19	0,086	50	170	19	0,112	**0,089**
39	366	21	0,057	43	323	19	0,059	**0,083**
40	205,5	20	0,097	0	205,5	15	0,073	**0,082**
41	180	16	0,089	70	110	16	0,145	**0,091**
42	209	21	0,100	35	174	18	0,103	**0,092**

As a result, this tool can be more readable when converted to a graph format, as Figure 24 illustrates, working as a trigger for the SM. At the end of eight sprints that were monitored, the team velocity is approximately 0,092 story points per hour. It can be observed on Figure 24 the decrease of their performance on sprint 39, which was due to vacations of some team elements. Because of these, the team decided to extend the sprint length, but still their velocity did not progress as expected. After this situation, when all the team members were back, their velocity started to increase suddenly, and finally returned to a value that was more similar to their normal rhythm.

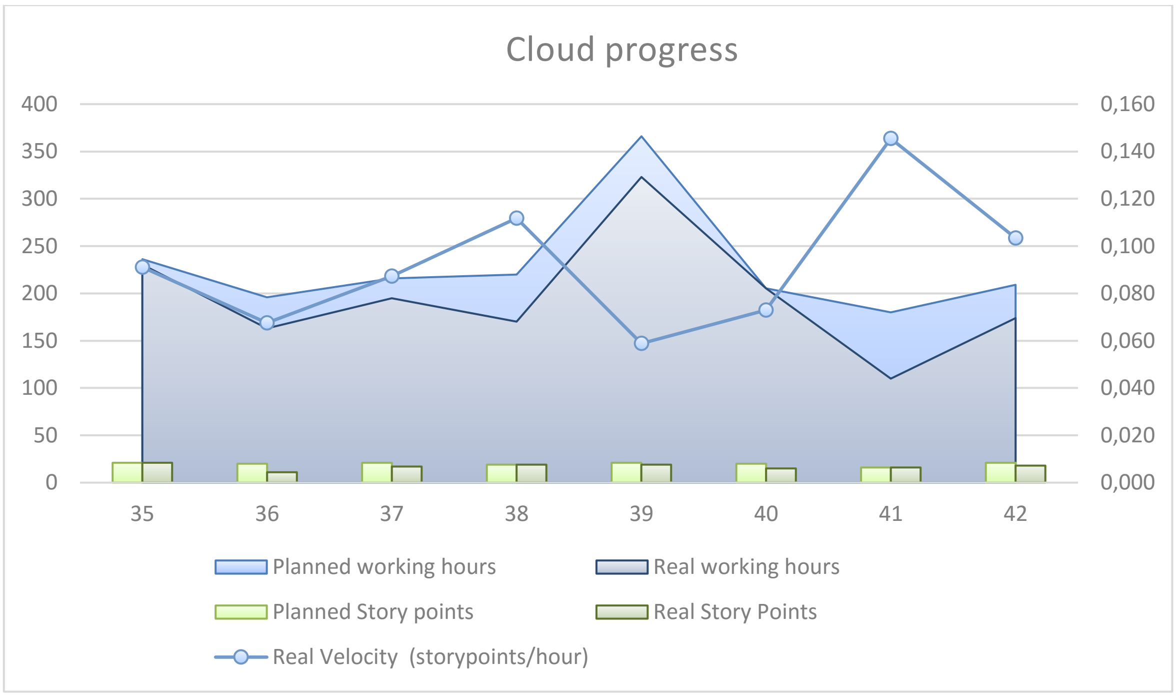


Figure 24 - Cloud team Progress during eight sprints

For the SM, it is really important to keep these values monitored, because they can be a good tool to track problems the team faces. After all, it is his role to defend the team and to remove impediments. So, if the team starts to get worse, either abruptly or continuously, he will observe big differences in their performance.

5.2.2 Requirements workflow

To try to solve the problem related to the requirements, and the decrease the risk of failure of the project, a requirements workflow was defined with the team. They must follow it and commit with it in order to guarantee the project's success. Or, at least, to guarantee that the project fulfils what was asked, improving the team's involvement on the requirements management. This way, all of them will be aligned and it will be possible for everyone, from the team or not, to know and to understand what the state of a requirement is.

The defined workflow of the requirements status is illustrated in Figure 25. The first stage is *Open*, which is the first approach to a requirement and so the rawest one. The next status is *To Be Groomed*, which represents a better-defined requirement. Only the requirements approved by the team and stakeholders will pass to the status *To Implement*, which concerns to the requirements that the development team should pass through, when the time for code development arrives. At last, and after the implementation, the requirement migrates to the state *To Be Tested*, where one of the team members will run the test in order to know if it was fulfilled.

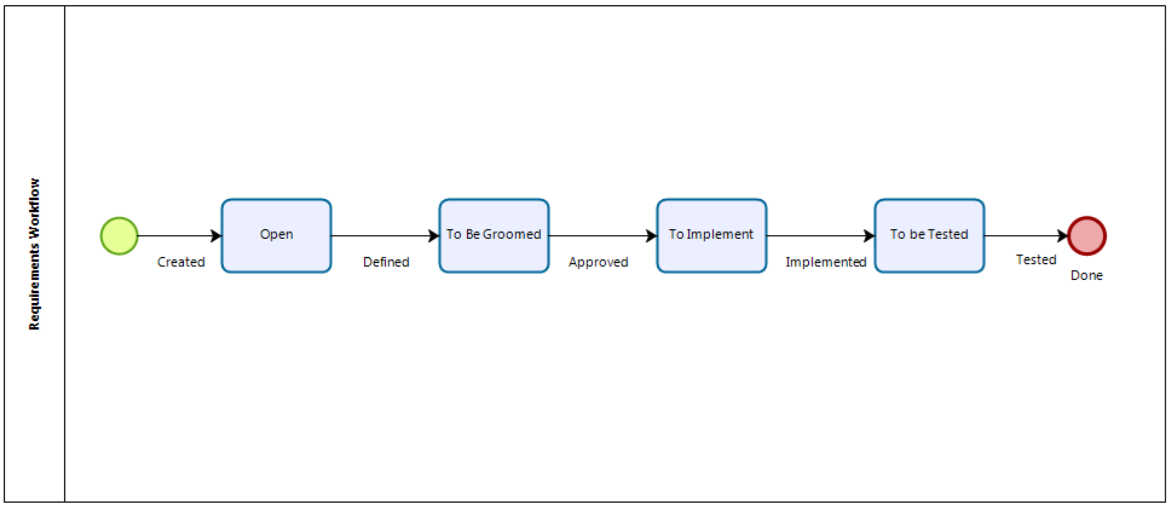


Figure 25 - Cloud team's requirements workflow

With the purpose of guaranteeing that the team follows the workflow, the first step was to make it visible on their Kanban board, as it can be seen on the upper left corner of Figure 23. Thus, every time they are working with a requirement, they can easily visualize which status it fits in, as which is the next status for a requirement already generated. Once the team managed and organized the project's requirements on a JIRA's plug-in, it was very important to create the workflow there, and mainly to review all the requirements in order to assign each one with their correct status.

5.2.3 Lack of Management Documentation

To solve the lack of management documentation problem it was decided, with the team involvement, that the Scrum Master should take a photo at the end of the sprint retrospective and if possible to report it where everyone could have easy access. Since the team already used the One Note tool to share information, it was decided to create there a section called “Retrospectives” to include all of them, in which each sprint retrospective is reported in its associated page.

To monitor if the improvement measures were fulfilled by the team, it was decided that not only should they be highlighted on the sprint retrospective report, in the One Note, but also that the last improvement measure should be a topic to discuss on the next sprint retrospective.

This measure proved to be essential for the grooming process, since meetings’ reports also began to be included there. This way, every team member has access to the folder and there is no information missed, especially when a member continues the work of another colleague. Moreover, it could also be worked as a small database where the Product Owner, and the team, save questions to clarify in the next grooming meetings or in the next meetings with the stakeholders, and of course to save the report of those meetings.

6. Evaluation and Learnings

This organization has a common problem to solve for all software teams. However, before implementing practices and perform changes on organizational concepts in all teams, it was necessary to focus in only one team, as mentioned at the beginning of the dissertation. The cloud team acted as an experimental team to solve this general problem. Despite the results presented before, it seemed viable and noticeably better to be the team members to give a realistic feedback, once they are the ones that most benefited from it. So, this is the more truthful way to know the impact of the improvement measures on the team. This feedback is really important as it defines if the improvement measures should or should not be implemented in the remaining teams. These remaining teams work every day with colleagues from other development centres, so the impact of applying new practices and improvement measures will be huge. That is why the feedback from the Cloud team is so important.

On the last retrospective the researcher participated, the contribution was already addressed informally, and the feedback was positive. However, in order to have a more structured and completed feedback, considering all the implemented measures, they were openly and directly asked about the actual contributions and consequently their impacts. Their answers were sent by e-mail, as all of them asked for more time to think, meditate and to structure their opinions on how the work was developed and their influences on the team's performance.

This chapter follows the same structure as the Grooming Process Development and Results, being divided for each topic addressed: Grooming and Complementary Work. The complete feedback from each team member is located on Appendix I – Feedback from the Cloud team.

6.1 Grooming

During this collaborative participation with the team many findings were collected, in order to improve each iteration of the grooming process. Despite the findings have not been so explicit and emphasised, at the end, some conclusions, learnings and advantages can be considered and highlighted.

One of the main learnings was that timing is not a rule. During the grooming process evolution, the process started with only one meeting for each sprint, with the purpose of creating the mock-up, review it and develop the user stories on the same sprint. However, and as observed, this cannot always be done. On the contrary, it is really unlikely that all of these activities can be held on the same sprint without consuming a lot of time from the team. In addition, and as occurred in sprint 38, other features and user stories from others functionalities can arise from the mock-up development of another functionality, and as such, it is also important to dedicate more time to their development to enrich the PB, without neglecting the quality of the user stories, once it can lead to the uncertainty the team initially faced.

To conclude, the timing, when the grooming should occur, is not a rule, as referred on chapter 2.6 Product Backlog Grooming. The "when" and "how often" it should happen is not the most important on grooming. Instead, guaranteeing it occurs and that the team is committed with it is. Because, in fact, the team should adapt to their rhythm. They just need to assure it does not compromise the PB, guaranteeing that it is never empty. This constitutes a huge step for the Product Owner, since the process allows to enrich the backlog with the user stories already matured, and ready to enter in a sprint. With a good backlog size, in user stories, epics and features, the Product Owner can have a vision and a defined time plan for a longer period than he had before.

The major conclusion, and as expected since the beginning, is the impact that grooming meetings have on the macro level of software engineering. Of course, this can bring repercussions until the code implementation, and thus dictating the project's success or failure. So, as soon as it is possible to assure the engineering process for the higher abstraction level, the risk at the end of the project will be minor.

> *"Each sprint, the grooming process makes it easier to understand user's needs and define user stories to keep the backlog consistent and aligned with the customer needs." – Team member E*

Before the grooming process' implementation, some benefits were studied and presented to the team in order to explain to them the "why". However, the main advantages are the ones the team now shares, after the implementation and continuously working in that way. The understanding of the customer's needs is now more clear, leading the team to identify themselves the goal of each user story. With this, the requirements are more visible now and the decrease of the planning meeting duration for almost a half was noticeable avoiding the constant interruptions with doubts. Finally, the researcher, as their SM, believes the increase of the team's motivation, commitment and mostly their autonomy were huge, as now they participate and have a voice on the long-term planning.

"Among the plentiful outputs from such activities, I would like to highlight: an increase of user stories with mock-ups and test cases attached; increased openness to talk about the customers' motivations and needs; and the constant refinement (or definition) of requirements." – Team member N

6.2 Complementary work

6.2.1 Velocity Monitoring

Before the intervention on the Cloud team, they only used to confer the sprint velocity at the Review. Despite many authors argue in favour of this approach, in young teams it is not always a good attitude once they are learning and trying to find their stable average velocity. By measuring it at the end, it does not allow them to change their path and adapt, during the work cycle, depending on what is going on during the sprint.

> *"In the worst scenario, we could decide to do tasks that will maximize the product increment for the current sprint, this thanks to the process of splitting points of all DoD present in User Stories." – Team member R*

Splitting the story points of a user story for each DoD allows the team, not only to have the notion of the effort that each one of them requires, but also to iterate the burn up chart more often. Having the burn up chart pined on their Kanban board allows them to update it every day, helping them to track when the work is blocked, and adapting their work to benefit more.

> *"The distribution of story points by the DoDs, as well as the burn chart, and the refinement of the tasks […] were very useful for evaluating sprint performance, resulting in greater consistency in assigning story points to user stories" – Team member A*

All the mentioned benefits also have positive repercussions on the planning. Over time, they will discover the real effort associated to the value that they use as a basis for comparison (usually eight story points). Also, they will learn and realize which DoD and engineering process demands more effort. These learnings will help the team to better estimate on each user story and their sprint velocity will stabilize. At this point, with the team's real average velocity more stable, the Product Owner can predict better, improving the release plans done by him and also making the project timelines more realistic.

> *[Burn chart] "the impact in morale was noticeable." – Team member N*

Seeing the burn chart daily grow up, the team becomes more motivated and encouraged once they see their work being done, improving their moral and willingness to finish their job.

6.2.2 Requirements workflow

Having a defined and well-structured workflow to follow the requirements is the basis to have a project's requirements list updated and organized, facilitating its management. It also allows the team to stay aligned in the process knowledge, avoiding different understandings by different members. Now, everyone is able to update a requirement created by another colleague, because the status of the requirements are well defined and equally understood by all. The sooner this workflow is implemented, the better, preventing a massive effort in the future, when much less effort is put for shorter periods of time.

> *"All the processes implemented on the team, also including the requirements workflow, are really useful and complement team's engineering process." – Team member E*

6.2.3 Management Documentation

The opinion that the team shared was that it is good to have all the management documentation organized, structured and controlled, but also that they are not the ones that benefit the most. Instead, it shows to be very important and useful for the Product Owner and Scrum Master. Firstly, because it keeps all team's performance data saved, so that the Product Owner could easily access it to create and improve time and release plans. And secondly, because it allows the Scrum Master to see and follow what is going on with the team along sprints. One example is to monitor if any negative topic or obstacle is tackled more than once in reviews and retrospectives, in order to define measures to overcome it. Another one is to track if the improvement measures defined by the team to solve a problem are being implemented. In this situation, the Scrum Master can put this improvement measure as a topic for the next retrospective.

> *"Finally, standardizing a place to store the team's procedures and ceremony report helped mitigate the team's lack of common knowledge of the processes due to the scarcity of localizations where these were previously." – Team member P*

7. RELATED WORK – PRODUCT OWNER COACHING

The main goal of this dissertation was related to the PO role and the team's involvement. Taking the cloud team as experimental, the improvements to the team were very visible, based on their feedback presented in the previous chapter. At this point, it seemed reasonable to start involving other teams in implementing this methodology, and perceiving how they deal with Scrum.
Some coaching sessions with the Product Owners were already initiated in order to improve their role, both in soft and hard skills. These sessions are moments for them to share the problems they are facing, not only related to communication and interpersonal relationships among the team, but also related to doubts about their responsibilities and work. These coaching sessions are given bi-weekly, by the Agile team. Besides the communications problems, addressed in the section 1.1.2, that continue to be the main topic discussed, some doubts arose about the tools they can use.

7.1.1 ROI

One main difficulty, common to all Product Owners during one of the coaching sessions, was: "How to prioritize work taking into account the business value and complexity?". So, the first question in fact is "Since we already know how to estimate complexity, measured by story points, how can we estimate business value?". Only after being able to answer this, it is possible to go back to the question proposed by the POs.
Starting from the beginning, we should always deliver the maximum value to the customer. After all, this is one of the reasons why Agile exists. So, the PB should always be prioritized by Return on Investment (ROI) (Hundermark, 2015), being ROI the quotient between the value and cost; we can associate this to Scrum following the equation (3).

$$ROI = \frac{Business\ Value}{Story\ points} \tag{3}$$

The story points, as mentioned, are related to the complexity in performing the user story, while the business value concerns to the value a user story brings to the customer, representing how much the user or customer needs that user story or functionality. There are many techniques that can help to determine this value, and, actually, in this situation is not the real value that matters. Instead, it is only necessary to obtain a relative value once it will be translated to a ratio. What is really important, at this point, is that a chosen user story brings more or less value compared to another one, as happens with the story points. So, in this situation, the group leader, who has the last voice, does not force the POs to use a specific scale or monetary value. Because, as mentioned, the focus is to keep coherency and consistency with the data used.

Returning to the ROI, as the PBIs must be prioritized following this ratio, it is possible to conclude that the first user stories to enter in a sprint, will be the ones that have high business value and low complexity. For reinforcing this idea, Botton (2012) presents a visible way to prioritize PBIs, as shows Figure 26 below. Following this figure, the user stories in the left upper corner will be implemented first and the ones in the bottom right corner will be the last to be implemented.

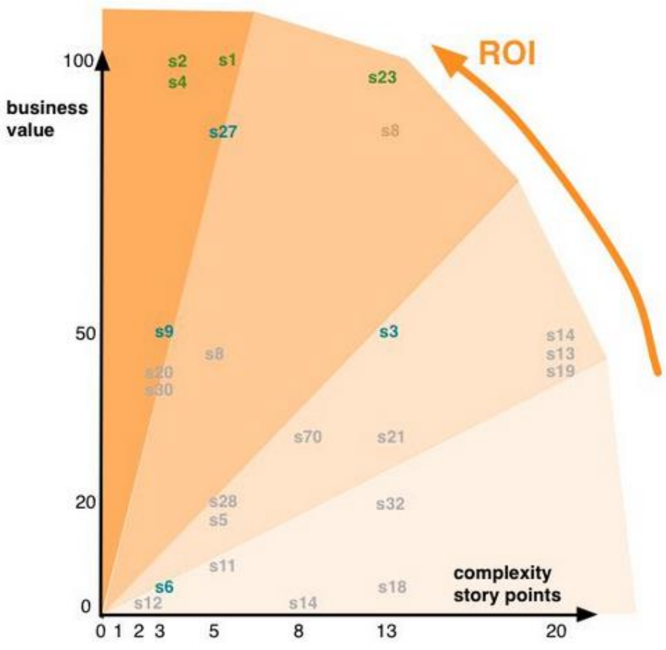
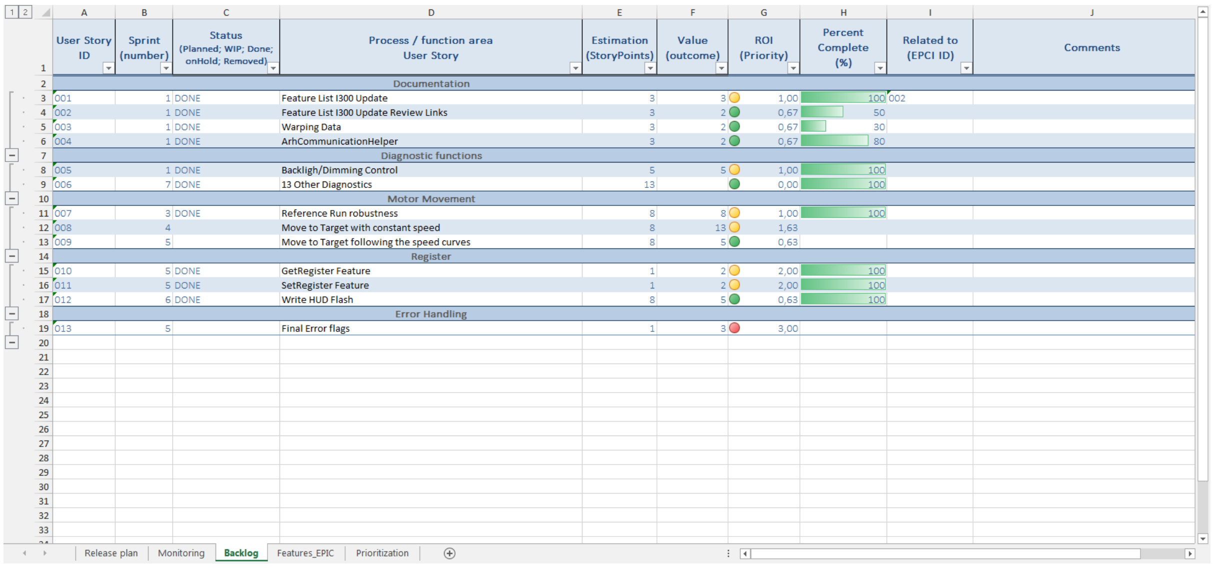


Figure 26 – User stories' ROI positions
(Botton, 2012, p. 1)

7.1.2 Complete Backlog

The ROI topic will be continuously tracked, but in the meantime the team's POs continue to have some uncertainties and to need help. This time, the difficulty is on how to compact all the information they need to manage in a single document to avoid excess of documentation, which many times can lead to them being lost and losing track. For this, it was developed an Excel Backlog, which includes more information than the expected. Usually, the backlogs already include the PBIs and historical velocity data of the team, but now, it was also included a sheet that represents the release plans, another sheet that allows to visualize the ROI in order to perform the prioritization, and of course, its calculation.

Figure 27 shows, the *Backlog* has the user stories allocated to epics (that are horizontally grouped) to improve organization. It includes each user story identification (ID), the sprint when it was or will be developed, the estimation in story points and the value. With this, the ROI is calculated and a colour is added according to the priority, being red the higher priority and green the lower, as observed in column G of Figure 27. It is also possible to insert the complete percentage, showing as a bar on the cell to be more visible.



User Story ID	Sprint (number)	Status (Planned; WIP; Done; onHold; Removed)	Process / function area User Story	Estimation (StoryPoints)	Value (outcome)	ROI (Priority)	Percent Complete (%)	Related to (EPCI ID)	Comments
			Documentation						
001	1	DONE	Feature List I300 Update	3	3 🟡	1,00	100	002	
002	1	DONE	Feature List I300 Update Review Links	3	2 🟢	0,67	50		
003	1	DONE	Warping Data	3	2 🟢	0,67	30		
004	1	DONE	ArhCommunicationHelper	3	2 🟢	0,67	80		
			Diagnostic functions						
005	1	DONE	Backligh/Dimming Control	5	5 🟡	1,00	100		
006	7	DONE	13 Other Diagnostics	13	🟢	0,00	100		
			Motor Movement						
007	3	DONE	Reference Run robustness	8	8 🟡	1,00	100		
008	4		Move to Target with constant speed	8	13 🟡	1,63			
009	5		Move to Target following the speed curves	8	5 🟢	0,63			
			Register						
010	5	DONE	GetRegister Feature	1	2 🟡	2,00	100		
011	5	DONE	SetRegister Feature	1	2 🟡	2,00	100		
012	6	DONE	Write HUD Flash	8	5 🟢	0,63	100		
			Error Handling						
013	5		Final Error flags	1	3 🔴	3,00			

Figure 27 - Product Owners' backlog

At the same time, on the Excel sheet *Prioritization* the graph that represents ROI is updated with the localization of the user stories on its story points and business value, presented on Figure 28.

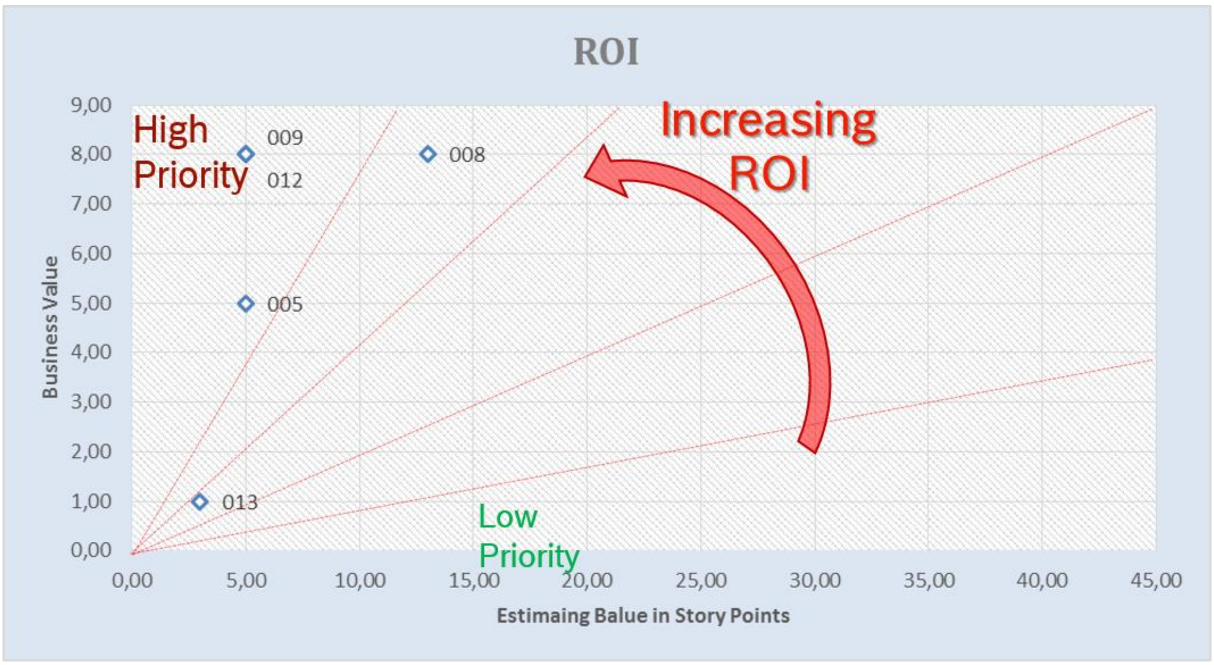


Figure 28 - Distribution of the user stories per ROI

Now, and with the division of the graph, as shown in Figure 28, it is more perceptible which user stories should be done first and last. If any doubts prevail, the Product Owner can also see the user stories sorted by ROI, by decreasing order, as shown in Appendix II – Product Owner Backlog. It is important to note that these figures only concern to a small example, from a team that was not allowed to give their complete backlog. However, it is expected that the teams keep their backlog more filled, so it can be possible to build a release plan.

Another important topic is the estimation. Although the Product Owner estimates the complexity of the user stories for the prioritization, it is the team who dictates the final vote on the sprint planning meeting. Nevertheless, the Product Owner should involve the team on the grooming phase, not only for the reasons that were given before on the implementation with Cloud team, but also to have their opinion concerning the estimation. This way, even if the value changes on the sprint planning meeting, it will be more precise, avoiding big differences between estimations and so avoiding minor changes on the release plan.

8. CONCLUSION AND FUTURE RESEARCH

The ultimate purpose of this dissertation was to develop a complete refinement process: the whole breakdown since a customer requirement to a small work package that can fit in a work cycle. The main limitation associated with this goal was the lack of communication between the parties involved in a project, which did not allow the teams' involvement in management. To accomplish that goal, and to solve that issue, it became necessary to focus on one team only. Having just one team, it was easier to understand how they managed their work, all the interfaces, processes and roles they had. With all the context well defined, the process modelling became more realistic.

During this collaborative participation, several other processes and work practices were addressed and improved. Also, the fact that the researcher had been the team's scrum master facilitated all the changes, being the improvement measures smoothly inserted. This way, the team was really committed and it was easier to adapt their work to the new practices, without spending excessive effort and without abrupt fluctuations on their performance.

As stated in chapter 6, the feedback given by the team was really positive. Not only did the improved measures have an impact on their work and organization, but also in their autonomy and openness to the work.

> *"Overall, I felt that all the processes implemented by our Scrum Master had a positive effect on the team. The fact that we kept following those processes – even after Francisca transitioned to another team – is the greatest proof of that influence." – Team member N*

Knowing that the processes implemented are still followed through, is a real proof that the effort was worth it. Also, the team's maturity was visibly improved, both in terms of engineering processes and management, as in responsibility and commitment with the team and the project.

> *"We are gaining maturity in the way we work as a team and the ceremonies have been effective. We've just won! Thank you Francisca!" – Team member A*

The grooming process showed good results to the team, as it was possible to conclude considering the feedback from the team, who most benefits from it. So, the next step is to take this approach to other software teams on Car-Multimedia. An agreement with the Product Owner should be made, so that all can agree on their role. This way, the teams get their commitment on participating and empowering the grooming process. This closeness will improve the communication among the team, allowing them to get more involved on planning for the long-term and participating on decision making with the Product Owners.

However, it is important to highlight that the implementation should be participative as it was with the Cloud team. Only this way, it will be possible to help them follow the process and also to help them in parallel topics, as these can have a huge impact on the team and in the process. In fact, the process worked well on the Cloud team due to other problems the team had at the beginning, that were improved too; so, it worked as a whole. Therefore, the grooming process implementation in future teams should be smoothly inserted and adaptable to them.

References

Ahmad, M. O., Markkula, J., & Oivo, M. (2013). Kanban in software development: A systematic literature review. *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, 9–16. https://doi.org/10.1109/SEAA.2013.28

Balaji, S. (2012). Waterfall vs v-model vs agile : A comparative study on SDLC. *Waterfall vs V-Model vs Agile: A Comparative Study on SDLC*, *2*(1), 26–30. https://doi.org/10.1.1.695.9278

Baptista, A., Santos, F., Páscoa, J., & Sändig, N. (2016). Project Management Methodologies as Main Tool for Current Challenges in Global Economy Driving Historical Changes. *Journal of Advanced Management Science*, *4*(2), 146–151. https://doi.org/10.12720/joams.4.2.146-151

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). Agile Manifesto. *Software Development*. Retrieved from http://agilemanifesto.org/

Bosch. (2015). Bosch Nova. *Comunicação Institucional Da Bosch Em Portugal*, 21.

Bosch. (2016a). About Bosch - Mission Statement "We are Bosch." *Bosch Global Network (Internal Information)*. Bosch Car Multimédia Portugal, S.A. Braga.

Bosch. (2016b). *Manual de acolhimento Bosch* (Departamen). Bosch Car Multimédia Portugal, S.A. Braga.

Bosch. (2017). BrgP Organization Chart. *Bosch Global Network (Internal Information)*. Bosch Car Multimédia Portugal, S.A. Braga.

Botton, T. (2012). Key Dimensions of User Stories. Retrieved from https://www.scrumalliance.org/community/articles/2012/january/key-dimensions-of-user-stories

Christoffer. (2015). Introducing: Agile. Retrieved April 19, 2017, from https://retrospektivet.wordpress.com/2015/03/09/introducing-agile/

Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional.

Cohn, M. (2006). *Agile Estimating and Planning*. Pearson Education.

Cohn, M. (2017a). Playing Poker. Retrieved April 19, 2017, from https://www.mountaingoatsoftware.com/agile/planning-poker

Cohn, M. (2017b). Scum. Retrieved April 19, 2017, from https://www.mountaingoatsoftware.com/agile/scrum

Crisp, C., & McKenna, D. (2016). *The Art of Scrum: How Scrum Masters Bind Dev Teams and Unleash Agility* (Vol. 4). https://doi.org/10.1007/978-1-4842-2277-5_10

Dean, B. J., Pessanha, B. G., Langfeldt, N., Pritchard, S., & Stanger, J. (2006). *Agile Project Management with Scrum*.

Denning, S. (2016a). Explaining Agile. *Forbes*, 1–14. Retrieved from http://www.forbes.com/sites/stevedenning/2016/09/08/explaining-agile/#4aa5f0c82ef7

Denning, S. (2016b). HBR's Embrace of Agile. Retrieved December 18, 2016, from http://www.forbes.com/sites/stevedenning/2016/04/21/hbrs-embrace-of-agile/#2c15bfca27fe

Fondahl, J. (1987). The history of modern project management. *Project Management Journal*, (2), 1–7.

GlobalNet, B. (2016a). About BrgP/ENG - Braga Development.

GlobalNet, B. (2016b). Bosch in Portugal.

Grapenthin, S., Poggel, S., Book, M., & Gruhn, V. (2015). Improving task breakdown comprehensiveness in agile projects with an Interaction Room. *Information and Software Technology, 67*, 254–264. https://doi.org/10.1016/j.infsof.2015.07.008

Hull, E., Jackson, K., & Dick, J. (2006). *Requirements Engineering*. (K. J. Jeremy Dick, Elizabeth Hull, Ed.), *Requirements Engineering*. Springer. https://doi.org/10.1145/336512.336523

Hundermark, P. (2015). *Do-Better-Scrum*. agile 42 - The Agile Coaching Company.

Kniberg, H. (2007). *Scrum and XP from the Trenches - How we do Scrum*. C4Media, Publisher of InfoQ.com.

Kwak, Y. H. (2005). A brief history of Project Management. *The Story of Managing Projects: An Interdisciplinary Approach*, (1916), 1–10.

Lawrence, K. (2013). Developing Leaders in a Business. *UNC Executive Development*, 1–15.

Leffingwell, D. (2011). *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. (I. Pearson Education, Ed.) (Kindle Edi).

Lei, H., Ganjeizadeh, F., Jayachandran, P. K., & Ozcan, P. (2017). A statistical analysis of the effects of Scrum and Kanban on software development projects. *Robotics and Computer-Integrated Manufacturing*, *43*, 59–67. https://doi.org/10.1016/j.rcim.2015.12.001

Lima, R. M. (2011). *GIP - GESTÃO INTEGRADA DA PRODUÇÃO - Texto de apoio*.

Mahnič, V., & Hovelja, T. (2012). On using planning poker for estimating user stories. *Journal of Systems and Software, 85*(9), 2086–2095. https://doi.org/10.1016/j.jss.2012.04.005

Mifsud, J. (2013). Requirements Gathering: A Step By Step Approach for a Better User Experience (Part 1). Retrieved July 25, 2017, from http://usabilitygeek.com/requirements-gathering-user-experience-pt1/

Moløkken-Østvold, K., Haugen, N. C., & Benestad, H. C. (2008). Using planning poker for combining expert estimates in software projects. *Journal of Systems and Software*, *81*(12), 2106–2117. https://doi.org/10.1016/j.jss.2008.03.058

Mountain Goat Software. (2017a). Scrum Task Board. Retrieved October 4, 2017, from https://www.mountaingoatsoftware.com/agile/scrum/scrum-tools/task-boards

Mountain Goat Software. (2017b). What is Agile Project Management? Retrieved May 26, 2017, from https://www.mountaingoatsoftware.com/agile/agile-project-management

Muntean, M. I., Mircea, G., & Pop, A. (2016). Analyzing Agile Development – from Waterfall Style to Scrumban. *Informatica Economica*, *20*(4/2016), 15–25. https://doi.org/10.12948/issn14531305/20.4.2016.02

Nedre, N. (2017). How to choose between Agile and Lean, Scrum and Kanban — which methodology is the best? Retrieved June 21, 2017, from https://realtimeboard.com/blog/choose-between-agile-lean-scrum-kanban/#.WUqCj3Zwz7w

Nikitina, N., Kajko-Mattsson, M., & Stråle, M. (2012). From scrum to scrumban: A case study of a process transition. *2012 International Conference on Software and System Process, ICSSP 2012 - Proceedings*, 140–149. https://doi.org/10.1109/ICSSP.2012.6225959

O'Brien, R. (1998). An Overview of the Methodological Approach of Action Research. *University of Toronto*.

Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., & Still, J. (2008). The impact of agile practices on communication in software development. *Empirical Software Engineering, 13*(3), 303–337. https://doi.org/10.1007/s10664-008-9065-9

PMI. (2013). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) Fifth Edition*. *Project Management Institute*. https://doi.org/10.1002/pmj.21345

Rubin, K. S. (2012). *Essential Scrum*. Pearson Education, Inc.

Schön, E., Thomaschewski, J., & Escalona, M. J. (2016). Agile Requirements Engineering: A Systematic Literature Review. *Computer Standards & Interfaces*, *49*. https://doi.org/http://dx.doi.org/10.1016/j.csi.2016.08.011

Schwaber, K., & Sutherland, J. (2016). The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game. *Scrum. Org and ScrumInc.*, *2*, 17. https://doi.org/10.1053/j.jrn.2009.08.012

Scruminc. (2017). Sprint Burndown Chart. Retrieved June 13, 2017, from https://www.scruminc.com/sprint-burndown-chart/

Sliger, M. (2006). Bridging the Gap: Agile Projects in the Waterfall Enterprise. *Better Software Magazine*, (August), 26–31. Retrieved from http://programmeiiba.programmedevelopment.com/public/uploads/files/agile_-_bridging_the_gap.pdf

Sliger, M., & Broderick, S. (2008). *The Software Project Manager's Bridge to Agility*. Addison Wesley Professional.

Špundak, M. (2014). Mixed Agile/Traditional Project Management Methodology – Reality or Illusion? *Procedia - Social and Behavioral Sciences*, *119*, 939–948. https://doi.org/10.1016/j.sbspro.2014.03.105

Stretton, A. (2007). A Short History of Modern Project Management. *PM World Today*, *IX*(X), 1–18.

STXNEXT. (2017). How to use Burndown Charts for transparent and predictable development results. Retrieved October 4, 2017, from https://stxnext.com/blog/2017/05/17/all-about-burn-down-charts/

Susman, G. I., & Evered, R. D. (1978). An assessment of the scientific merits of action research. *Administrative Science Quarterly*, *23*(4), 582–603. https://doi.org/10.2307/2392581

Sutherland, J. (2010). *Scrum handbook. Scrum Training Institute*. Scrum Trining Intitute Press.

Sverrisdottir, H. S., Ingason, H. T., & Jonasson, H. I. (2014). The Role of the Product Owner in Scrum-comparison between Theory and Practices. *Procedia - Social and Behavioral Sciences*, *119*, 257–267. https://doi.org/10.1016/j.sbspro.2014.03.030

Tobergte, D. R., & Curtis, S. (2013). *A Guide to the Scrum Body of Knowledge*. *Journal of Chemical Information and Modeling* (Vol. 53). https://doi.org/10.1017/CBO9781107415324.004

Torrecilla-Salinas, C. J., Sedeño, J., Escalona, M. J., & Mejías, M. (2016). Agile, Web Engineering and Capability Maturity Model Integration: A systematic literature review. *Information and Software Technology*, *71*, 92–107. https://doi.org/10.1016/j.infsof.2015.11.002

Verner, J. M., Brereton, O. P., Kitchenham, B. a, Turner, M., & Niazi, M. (2012). Systematic literature reviews in global software development: A tertiary study. *Proceedings of 16th International Conference on Evaluation Assessment in Software Engineering - EASE 2012*, *85*, 2–11. https://doi.org/10.1049/ic.2012.0001

von Rosing, M., von Scheel, H., & Scheer, A.-W. (2011). Business Process Model and Notation. *The Complete Business Process Handbook*. https://doi.org/10.1007/978-3-642-25160-3

Zahraoui, H., & Janati Idrissi, M. A. (2015). Adjusting story points calculation in scrum effort & time estimation. *2015 10th International Conference on Intelligent Systems: Theories and Applications, SITA 2015*. https://doi.org/10.1109/SITA.2015.7358400

APPENDIX I – FEEDBACK FROM THE CLOUD TEAM

Member A

“There was a noticeable improvement in planning.
The definition of DoDs allowed us to focus on more concrete goals throughout the sprint. The distribution of story points by the DoDs, as well as the burn chart, and the refinement of the tasks helped us to have a sense of the evolution of the work compared to what we committed in the planning. These processes also were very useful for evaluating sprint performance, resulting in greater consistency in assigning story points to user stories (and apportioning them when necessary).
The introduction of requirements and mock-ups grooming was one of the biggest assets. It clearly facilitated the construction of user stories, the clarification of the requirements to be fulfilled and the identification of points to be clarified with the client.
The processes introduced by Francisca continue to be implemented, being well managed by the new Scrum Master. We are gaining maturity in the way we work as a team and the ceremonies have been effective. We’ve just won! Thank you Francisca! “

Member E

“There are good points that highlight the work carried out by Francisca with the development team and they must be mentioned.
Initially, throughout the sprint, the velocity chart didn't show the reality of the work done because we only obtained the points when we closed the user story. The story points split into Definition of Done was a good strategy to keep team's motivation along the sprint as we update the burn chart every daily meeting.
Another good tool implemented is the Grooming process. Each sprint, the grooming process makes it easier to understand user's needs and define user stories to keep the backlog consistent and aligned with the customer needs. This helps on the requirements clarification and definition between the Product Owner and the Development team.
To help the Scrum Master on the ceremonies like Planning, Review, and Retrospective, the files created to report each ceremony are really profitable and increase the productivity of Scrum Master on his tasks.

All the processes implemented on the team, including the requirements workflow, are really useful and complement team's engineering process.”

Member N

“In addition to its most obvious benefit – having a better idea of what you’re going to do next –, the Grooming Process led the team to invest time and effort on important activities previously neglected. Among the plentiful outputs from such activities, I would like to highlight: an increase of user stories with mock-ups and test cases attached; increased openness to talk about the customers’ motivations and needs; and the constant refinement (or definition) of requirements.

The velocity monitoring brought a better perspective of the work in progress, due to its small feedback loop. Instead of having to wait several days for a user story to be completed and seeing improvements in the burn down chart, the team was able to watch the points go up daily – the impact in morale was noticeable. It is also worth noting that this process allowed the team to have better estimates of the available capacity, even when a user story was not finished.

Overall, I felt that all the processes implemented by our Scrum Master had a positive effect on the team. The fact that we kept following those processes – even after Francisca transitioned to another team – is the greatest proof of that influence.”

Member P

“Previously, our velocity tracking was very reductive, as the story points were only considered if the whole user story was completed and, in most cases, these were only achieved near the end of the sprint.

The visualization of the team's velocity was disregarded as it provided no knowledge to the team, being only presented on the Sprint Review. To tackle this, I believe that splitting the user story points over the existing processes, accordingly with their complexity was an important step to better convey our progress along the sprint. Another important measure was to exhibit the velocity chart to keep track of the development team's day-to-day activities.

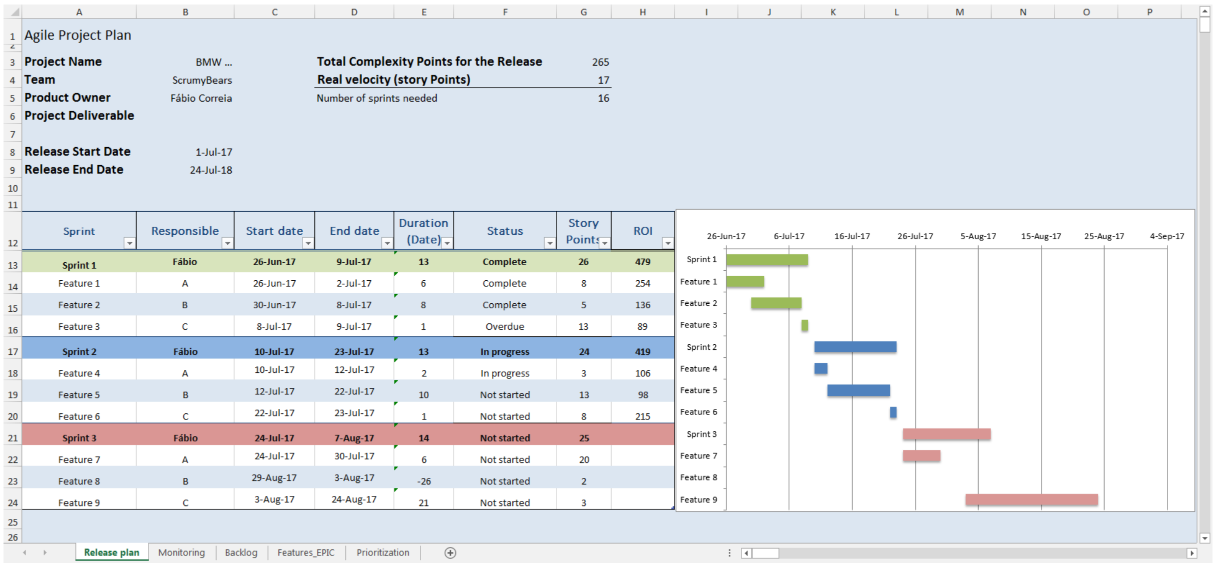
Finally, standardizing a place to store the team's procedures and ceremony report helped mitigate the team's lack of common knowledge of the processes due to the sparsity of localizations where these were previously.”

Member R

“The velocity monitoring helped the team to give more attention to all targets of one sprint. Over the time, the burn chart gives a precious information about how the sprint is going, and whether it will be possible to do everything or not. In the worst scenario, we could decide to do tasks that will maximize the product increment for the current sprint, this thanks to the process of splitting points of all DoD present in User Stories.
Other important process brought was the grooming process. It improved the Planning ceremony a lot, because the goals and new User Stories were already prepared before the beginning of the sprint, therefore, on the planning ceremony only details are decided for each user story, shortening dramatically the planning phase.”

APPENDIX II – PRODUCT OWNER BACKLOG

Release plan excel sheet from the Product Owner backlog.

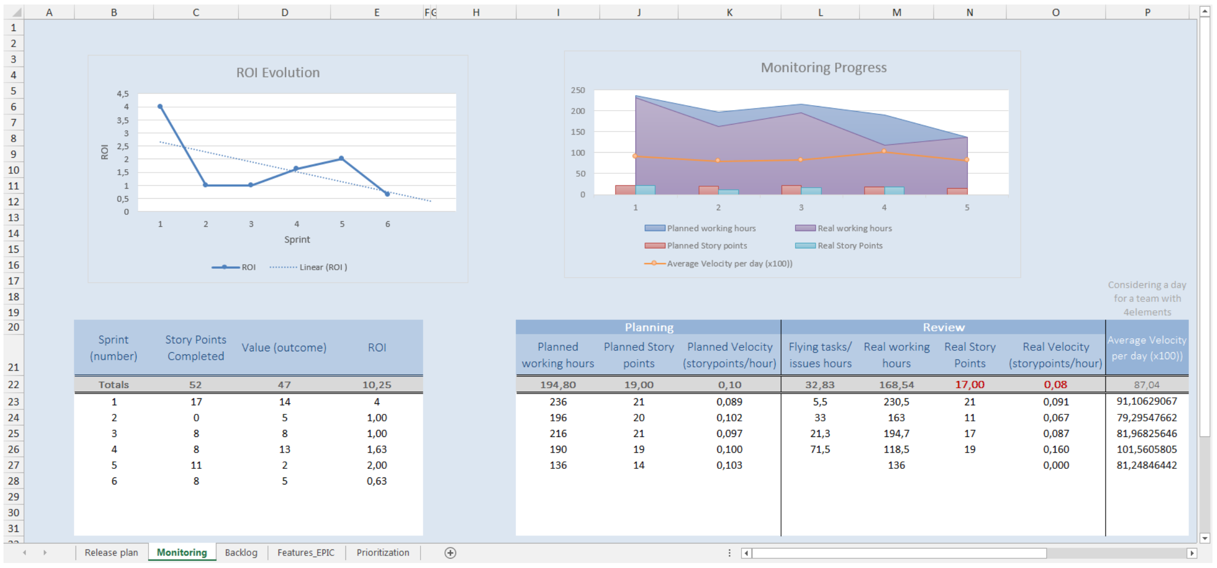


Agile Project Plan

Project Name	BMW ...	**Total Complexity Points for the Release**	265
Team	ScrumyBears	**Real velocity (story Points)**	17
Product Owner	Fábio Correia	Number of sprints needed	16
Project Deliverable			
Release Start Date	1-Jul-17		
Release End Date	24-Jul-18		

Sprint	Responsible	Start date	End date	Duration (Date)	Status	Story Points	ROI
Sprint 1	**Fábio**	**26-Jun-17**	**9-Jul-17**	**13**	**Complete**	**26**	**479**
Feature 1	A	26-Jun-17	2-Jul-17	6	Complete	8	254
Feature 2	B	30-Jun-17	8-Jul-17	8	Complete	5	136
Feature 3	C	8-Jul-17	9-Jul-17	1	Overdue	13	89
Sprint 2	**Fábio**	**10-Jul-17**	**23-Jul-17**	**13**	**In progress**	**24**	**419**
Feature 4	A	10-Jul-17	12-Jul-17	2	In progress	3	106
Feature 5	B	12-Jul-17	22-Jul-17	10	Not started	13	98
Feature 6	C	22-Jul-17	23-Jul-17	1	Not started	8	215
Sprint 3	**Fábio**	**24-Jul-17**	**7-Aug-17**	**14**	**Not started**	**25**	
Feature 7	A	24-Jul-17	30-Jul-17	6	Not started	20	
Feature 8	B	29-Aug-17	3-Aug-17	-26	Not started	2	
Feature 9	C	3-Aug-17	24-Aug-17	21	Not started	3	

Monitoring excel sheet from the Product Owner backlog.

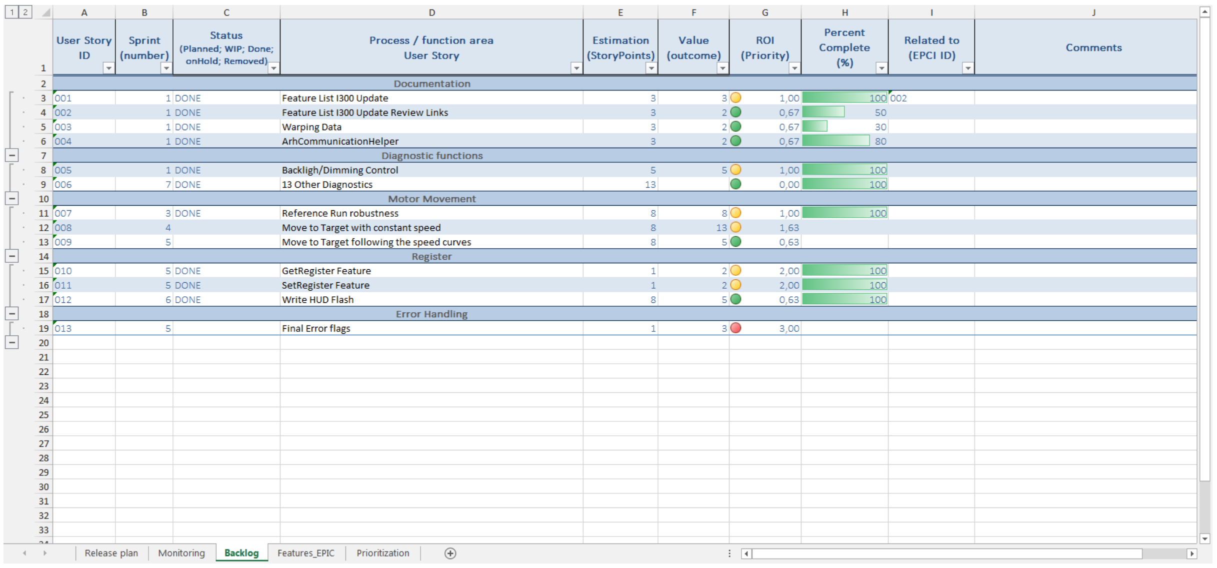


Sprint (number)	Story Points Completed	Value (outcome)	ROI
Totals	52	47	10,25
1	17	14	4
2	0	5	1,00
3	8	8	1,00
4	8	13	1,63
5	11	2	2,00
6	8	5	0,63

Considering a day for a team with 4elements

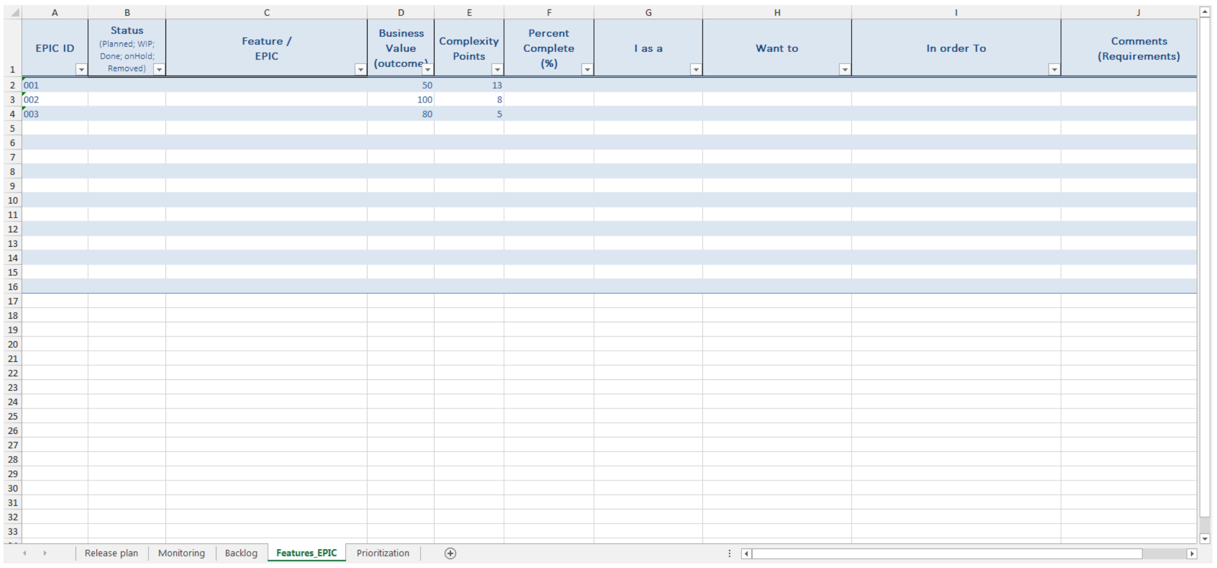
Planning			Review				
Planned working hours	Planned Story points	Planned Velocity (storypoints/hour)	Flying tasks/ issues hours	Real working hours	Real Story Points	Real Velocity (storypoints/hour)	Average Velocity per day (x100))
194,80	19,00	0,10	32,83	168,54	17,00	0,08	87,04
236	21	0,089	5,5	230,5	21	0,091	91,10629067
196	20	0,102	33	163	11	0,067	79,29547662
216	21	0,097	21,3	194,7	17	0,087	81,96825646
190	19	0,100	71,5	118,5	19	0,160	101,5605805
136	14	0,103		136		0,000	81,24846442

Backlog excel sheet from the Product Owner backlog.



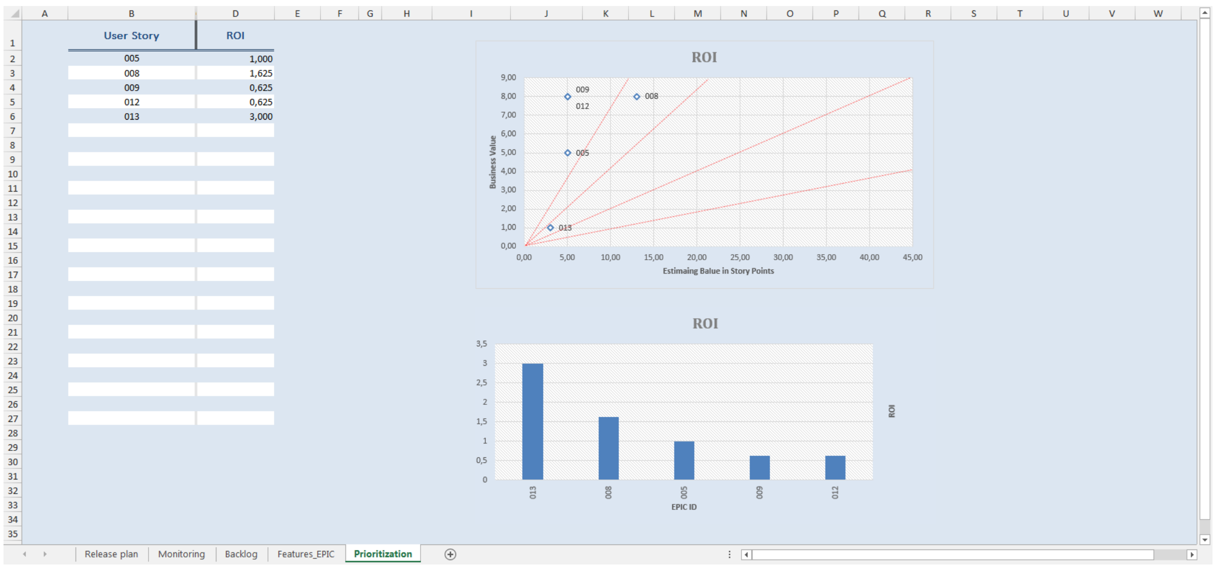
User Story ID	Sprint (number)	Status (Planned; WIP; Done; onHold; Removed)	Process / function area User Story	Estimation (StoryPoints)	Value (outcome)	ROI (Priority)	Percent Complete (%)	Related to (EPCI ID)	Comments
			Documentation						
001	1	DONE	Feature List I300 Update	3	3 🟡	1,00	100	002	
002	1	DONE	Feature List I300 Update Review Links	3	2 🟢	0,67	50		
003	1	DONE	Warping Data	3	2 🟢	0,67	30		
004	1	DONE	ArhCommunicationHelper	3	2 🟢	0,67	80		
			Diagnostic functions						
005	1	DONE	Backligh/Dimming Control	5	5 🟡	1,00	100		
006	7	DONE	13 Other Diagnostics	13	🟢	0,00	100		
			Motor Movement						
007	3	DONE	Reference Run robustness	8	8 🟡	1,00	100		
008	4		Move to Target with constant speed	8	13 🟡	1,63			
009	5		Move to Target following the speed curves	8	5 🟢	0,63			
			Register						
010	5	DONE	GetRegister Feature	1	2 🟡	2,00	100		
011	5	DONE	SetRegister Feature	1	2 🟡	2,00	100		
012	6	DONE	Write HUD Flash	8	5 🟢	0,63	100		
			Error Handling						
013	5		Final Error flags	1	3 🔴	3,00			

Feature_EPIC excel sheet from the Product Owner backlog.



EPIC ID	Status (Planned; WIP; Done; onHold; Removed)	Feature / EPIC	Business Value (outcome)	Complexity Points	Percent Complete (%)	I as a	Want to	In order To	Comments (Requirements)
001			50	13					
002			100	8					
003			80	5					

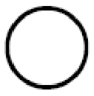
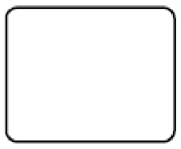
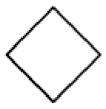
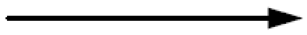
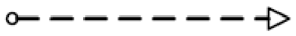
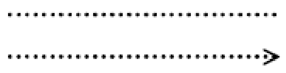
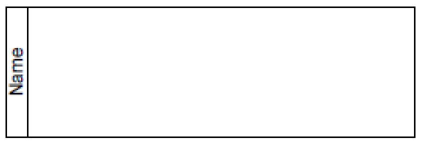
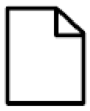
Prioritization excel sheet from the Product Owner backlog.



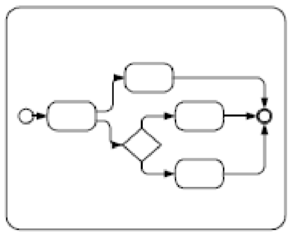
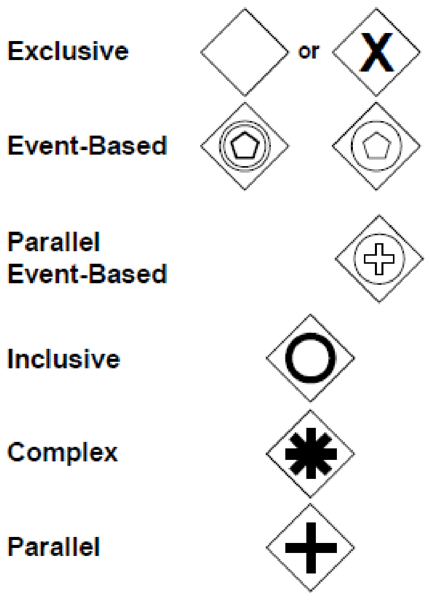
User Story	ROI
005	1,000
008	1,625
009	0,625
012	0,625
013	3,000

APPENDIX III – BUSINESS PROCESS MODEL AND NOTATION

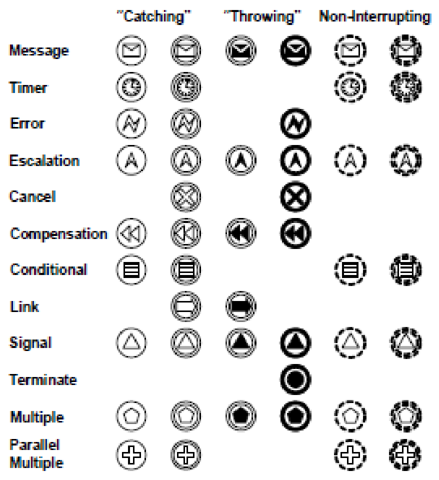
Basic Modelling Elements (von Rosing et al., 2011, p. 29;30)

Element	Description	Notation
Event	An Event is something that "happens" during the course of a Process (see page 238) or a Choreography (see page 339). These Events affect the flow of the model and usually have a cause (*trigger*) or an impact (*result*). Events are circles with open centers to allow internal markers to differentiate different *triggers* or *results*. There are three types of Events, based on when they affect the flow: Start, Intermediate, and End.	
Activity	An Activity is a generic term for work that company performs (see page 151) in a Process. An Activity can be atomic or non-atomic (compound). The types of Activities that are a part of a Process Model are: Sub-Process and Task, which are rounded rectangles. Activities are used in both standard Processes and in Choreographies.	
Gateway	A Gateway is used to control the divergence and convergence of Sequence Flows in a Process (see page 145) and in a Choreography (see page 344). Thus, it will determine branching, forking, merging, and joining of paths. Internal markers will indicate the type of behavior control.	
Sequence Flow	A Sequence Flow is used to show the order that Activities will be performed in a Process (see page 97) and in a Choreography (see page 320).	
Message Flow	A Message Flow is used to show the flow of Messages between two *Participants* that are prepared to send and receive them (see page 120). In BPMN, two separate Pools in a Collaboration Diagram will represent the two *Participants* (e.g., PartnerEntities and/or PartnerRoles).	
Association	An Association is used to link information and Artifacts with BPMN graphical elements (see page 67). Text Annotations (see page 71) and other Artifacts (see page 66) can be Associated with the graphical elements. An arrowhead on the Association indicates a direction of flow (e.g., data), when appropriate.	
Pool	A Pool is the graphical representation of a *Participant* in a Collaboration (see page 112). It also acts as a "swimlane" and a graphical container for partitioning a set of Activities from other Pools, usually in the context of B2B situations. A Pool MAY have internal details, in the form of the Process that will be executed. Or a Pool MAY have no internal details, i.e., it can be a "black box."	
Data Object	Data Objects provide information about what Activities require to be performed and/or what they produce (see page 205), Data Objects can represent a singular object or a collection of objects. Data Input and Data Output provide the same information for Processes.	

BPMN Extended Modelling Elements (von Rosing et al., 2011, p. 31;33)

Flow Dimension (e.g., Start, Intermediate, End) Start Intermediate End	As the name implies, the Start Event indicates where a particular Process (see page 238) or Choreography (see page 339) will start. Intermediate Events occur between a Start Event and an End Event. They will affect the flow of the Process (see page 249) or Choreography (see page 341), but will not start or (directly) terminate the Process. As the name implies, the End Event indicates where a Process (see page 246) or Choreography (see page 343) will end.	Start ○ Intermediate ◎ End ⭘
Expanded Sub-Process	The boundary of the Sub-Process is expanded and the details (a Process) are visible within its boundary (see page 173). Note that Sequence Flows cannot cross the boundary of a Sub-Process.	
Gateway Control Types	Icons within the diamond shape of the Gateway will indicate the type of flow control behavior. The types of control include: • Exclusive decision and merging. Both Exclusive (see page 290) and Event-Based (see page 297) perform exclusive decisions and merging Exclusive can be shown with or without the "X" marker. • Event-Based and Parallel Event-based gateways can start a new instance of the Process. • Inclusive Gateway decision and merging (see page 292). • Complex Gateway -- complex conditions and situations (e.g., 3 out of 5; page 295). • Parallel Gateway forking and joining (see page 293). Each type of control affects both the incoming and outgoing flow.	 Exclusive or X Event-Based Parallel Event-Based Inclusive Complex Parallel

BPMN Extended Modelling Elements (von Rosing et al., 2011, p. 31;33)

Type Dimension (e.g., None, Message, Timer, Error, Cancel, Compensation, Conditional, Link, Signal, Multiple, Terminate.)	The Start and some Intermediate Events have "triggers" that define the cause for the Event (see section entitled "Start Event" on page 238 and section entitled "Intermediate Event" on page 249). There are multiple ways that these events can be triggered. End Events MAY define a "result" that is a consequence of a Sequence Flow path ending. Start Events can only react to ("catch") a *trigger*. End Events can only create ("throw") a *result*. Intermediate Events can catch or throw *triggers*. For the Events, *triggers* that catch, the markers are unfilled, and for *triggers* and *results* that throw, the markers are filled. Additionally, some Events, which were used to interrupt Activities in BPMN 1.1, can now be used in a mode that does not interrupt. The boundary of these Events is dashed (see figure to the right).	 "Catching" "Throwing" Non-Interrupting Message Timer Error Escalation Cancel Compensation Conditional Link Signal Terminate Multiple Parallel Multiple