Gamification of Software Development to Raise Compliance with Scrum

Gonçalo Maia da Costa

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Miguel Leitão Bignolas Mira da Silva
Prof. Daniel Jorge Viegas Gonçalves

Examination Committee

Chairperson: Prof. José Luís Brinquete Borbinha
Supervisor: Prof. Miguel Leitão Bignolas Mira da Silva
Members of the Committee: Prof. Sandra Pereira Gama

October 2017

Acknowledgments

I would like to thank my supervisors Miguel Mira da Silva and Daniel Gonçalves for the time they had for me and for all the help they gave me throughout this year. I learned a lot with them.

Thanks a lot to my family, specially my parents, for being there for me during all these years and for giving me all the necessary support for me to reach this stage of my life. Without them, this would definitely be much harder to achieve.

Thanks to my university colleagues who shared with me the pain and hurdles that kept on showing up during this two years in Técnico.

Thanks to the gamification team, with which was an absolute pleasure to work with, and without whom this project would not be conceivable. A special acknowledge to Rita, who was tireless during this project and who still had the time to help me with all my endless questions and mistakes.

A big thank you to Jéssica for never giving up on me and being there for me during this whole process, even when I was impossible to put up with.

Thank you to my old-time friends for all the support and for being on my side for all this time, even when things were darker. Thanks to my friend Viven Rao for having time to listen to my problems even though he is on the other side of the world.

Thank you all.

Abstract

Even though agile methodologies have been widely adopted by practitioners, they still struggle to meet their goals in the scope of time, and budget (e.g. unresolved tasks, and failed deadlines). Resistance to follow agile practices such as estimating tasks, planning sprints, or write documentation has been pointed as problems in previous research. These challenges that organisations have been trying to solve can be explained, in part, by the lack of motivation from practitioners to adopt the necessary agile practices.

To tackle this problem, several researchers have used a technique called gamification i.e., the use of game elements, and mechanics, in non-game contexts to promote the adoption of certain behaviours. The research methodology used throughout this study is the Design Science Research Methodology (Design Science Research Methodology (DSRM)). This artefact represents an iteration in which the problem is validated with data from a real-world project, and a prototype is developed.

The solution presented in this artefact is focused on Scrum methodology, and materialised as a gamification app, with the goal of motivating practitioners to adopt Scrum practices. These are achieved by engaging them with feedback loops and game components, increasing their interest, and satisfaction at work. Finally, results from a real project were retrieved to allow the solution to tackle the most relevant problems.

Future work should include a study with multiple Scrum teams to better understand how to tailor the solution to different personal and motivational needs.

Keywords

gamification, software development, agile, scrum, motivation

Resumo

Apesar de as metodologias *Agile* serem frequentemente adoptadas por profissionais, existem ainda dificuldades no que toca a cumprir objectivos, metas temporais, e de custos (por exemplo, tarefas não resolvidas ou prazos falhados). A resistência à adopção de práticas agile (estimar tarefas, planear sprints, ou escrever documentação) tem sido apontada diversas vezes em investigações anteriores. Estes desafios que as organizações tentam resolver são explicados, em parte, pela falta de motivação dos profissionais para adoptar essas práticas.

Para combater este problema, vários investigadores usaram uma técnica chamada ”gamificação” ou seja, o uso de elementos, e mecânicas de jogo, em contextos da vida real para promover a adopção de certos comportamentos. A metodologia utilizada nesta investigação é o Design Science Research. Este documento representa uma iteração desta metodologia no qual o problema é validado com dados de um projecto real, e uma solução é apresentada.

A solução para o problema deste estudo é uma aplicação de gamificação que tem como objectivo motivar a adopção das práticas de *Scrum*. Este objectivo é atingido com a utilização de *feedback loops*, e dinâmicas de jogo, que envolvem os profissionais no seu trabalho e aumentam o seu interesse, e motivação por aquilo que fazem. Por fim, foram extraídos resultados de um projecto real de forma a perceber quais os problemas mais relevantes que a solução deveria atacar.

Como trabalho futuro desta investigação, deverá ser feito um estudo com várias equipas de *Scrum* para perceber melhor como adaptar a nossa solução às motivações, e necessidades dos profissionais.

Palavras Chave

gamificação, desenvolvimento de software, agile, scrum, motivação

Contents

List of Figures

List of Tables

Listings

Acronyms

ASDM	Agile Software Development Methodologies
CSV	Comma-separated Values
DSRM	Design Science Research Methodology
PBL	Points, Badges and Leaderboards
SDT	Self-Determination Theory
SP	Story Points
XP	Experience Points

1

Introduction

Software industry is nowadays full of challenges associated to both software and people involved in its development. Software development organisations have been adopting new tools, and methodologies [5] to cope with the increasingly dynamic business environments, fast technologies development, and increase of end user demands [6]. Some recent and popular approaches focus on Agile Software Development Methodologies (Agile Software Development Methodologies (ASDM)) combined with software tools. Agile methodologies, such as Scrum, promote self-organising and cross-functional teams that develop software in short iterations and work close with the customer. Scrum is the methodology with a higher rate of adoption [7], in which teams of practitioners plan and track their work iterations (called sprints) based on Scrum artefacts, and communicate in Scrum events [8].

While the benefits of agile methods (e.g. improved product quality, and customer satisfaction) have been demonstrated, agile teams are still facing challenges mostly associated to human factors [9]. Challenges faced by organisations include the motivation of teams to be more productive and to follow the employed methodologies. Factors as no work recognition, conflicting priorities [10], working context (e.g. multi-site development), and poor long-term planning [11] are problems associated to *Agile* methodologies. Also, lower motivation could result in missing deadlines for task completion, rejection of good practices for both software and knowledge management, and hence unfulfilled customers' needs.

Gamification has been used as a strategy to improve people motivation by combining game components, mechanics, and dynamics to engage, motivate and challenge teams throughout the process. This research intends to get developers to adopt Scrum practices, such as documenting tasks, resolving them, and collaborating with team members to get work done inside the planned time slot.

The solution proposed by this research is a gamification tool to promote behaviour change using game elements such as immediate feedback, and a diversified rewards system to attempt creating an amusing and challenging environment. The artefact focuses on the software development industry even though this topic could be applied to other contexts, such as learning.

Despite being the most popular agile methodology, also suffering from the lack of motivation problem, we are not aware of any proposal based on gamification that was evaluated with Scrum teams in practice.

The proposal was demonstrated in a Portuguese IT company that uses Jira Software to manage its software development projects. This consisted on the implementation of our tool in a Scrum project so it could be used by a team and provide feedback necessary for its evaluation.

Two steps were performed to evaluate the proposal. First, the analysis of data extracted from a project conducted by the selected company. Its goal was to create two scenarios that represent the time before, and the time after the use of our tool so we could perform a comparison. Then, the Scrum Master was interviewed to assess the solution acceptance in what regards simplicity, usefulness, and efficacy when compared against the goals defined to it.

The used research methodology was the DSRM which impacts this document structure and the way

this research is conducted. Hevner *et al.* [12] came up with seven guidelines to conduct a research, from which the most important is the production of an artefact to tackle a problem [12]. The artefact's design is meant to meet three objectives: providing a process for researchers who want to use design as a research mechanism, building upon prior literature while integrating DSRM principles into comprehensive methodology for conducting the research, and providing a mental model for the conduct, and presentation of the research.

This methodology is unfolded into six steps that must be accomplished throughout the research:

1. **Problem and motivation identification:** It justifies the value of a solution that the researcher and the audience are motivated to pursue;

2. **Define the objectives for a solution:** Objectives of a solution are deduced from the identified problem;

3. **Design and development:** The artefact is created after defining its functionality and architecture;

4. **Demonstration:** The artefact should be used to demonstrate a solution of the problem (or instance of it) being that a experimentation, simulation, case study or other;

5. **Evaluation:** The solution objectives are compared with the obtained results. Also, the researcher can choose to iterate back to "Design and Development" to improve the artefact effectiveness or to move to the next step;

6. **Communication:** Some aspects such as the problem, its importance, and the artefact's utility and effectiveness must be communicated.

Although this process has an ordered structure it is not required for researchers to follow a sequence order from the first to the last activity [1], hence the step to start the research with depends on the desired approach. Figure 1.1 represents the DSRM process mapped to this research.

This methodology provides an intuitive path beginning with understanding the problem to address, and what are the associated objectives, so it can be as much efficient as possible. Only then, with the previous knowledge in mind, should the research gain form, and become an instantiation (here, as a software). The idea behind this is to understand the scope of the studied topic, and to be able to define the most adequate features that comply with the defined goals.

In addition, the design and development of this instance could be in compliance with the research problem, and objectives, but still fail to provide the expected results. For that reason, it should be demonstrated, and then evaluated, for the instance to be tailored to possible constraints that were previously unconsidered. Finally, communication provides a contribution to the scientific community with results, and lessons learned, relevant to other researchers.

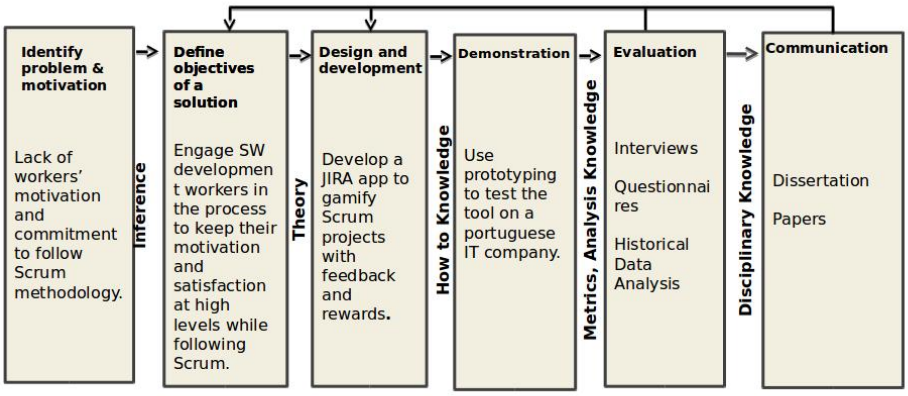


Figure 1.1: DSRM process model tailored to this research work (adapted from [1]).

Following this chapter, Chapter 2 exposes the defined research problem. Chapter 3 contains an analysis of important concepts, which is followed by previous research, and similar gamification tools, in Chapters 4. Chapter 5 comprises the proposal, in which the Gamification Design Framework is detailed. Chapter 6 depicts the implemented features, and the tool's demonstration is detailed in Chapter 7. Chapter 8 evaluates the research proposal which is then followed by a conclusion in Chapter 9.

2

Research Problem

Contents

Following the first step of DSRM, this research begins with the identification of its problem and motivation.

Practitioners and researchers have been studying failure in software development project for the past few years. The factors that lead to failure are still to be understood, and the way to increase projects' efficiency remains a challenge. In the scope of this research, failure regards requirements' incomprehension, the non-fulfilment of deadlines, and consequently, the inability to deliver the right product. Several causes can lead to software project failure, most of them based on human, and not technological factors [13]. An example is the practitioners' resistance to rigorously or uniformly following software development methodologies that are proven to improve overall success of software projects [14].

Many studies have attempted to understand the problems behind the poor implementation of software development methodologies, which include: lack of detail in requirements specification, inadequate planning, lack of progress tracking and feedback mechanisms, poor communication between practitioners, and lack of motivation to adopt the methodologies [15–19].

Even though agile methodologies are often more engaging [5], they still present specific challenges that must be addressed to achieve a successful outcome [20]. The following list presents the main challenges referenced in the literature:

- Documentation solely based on user stories and code is insufficient [21], and can lead to traceability issues [5] [22], loss of time [23], and loss of knowledge [21, 24];
- Communication can be affected by practitioners' lack of social skills and fear that transparency encouraged in agile values can emphasize their deficiencies [9];
- Effectiveness of some practices can be harmed in large teams, due to the high number of communication channels that have to be maintained [11, 23, 25];
- Ceremonies are frequently ineffective since they are not always adapted to reflect the project complexity [23];
- Daily meetings time is hard to setup due to practitioners different schedules, and people can get distracted if these ceremonies are not kept short [23];
- User stories are difficult to create and estimate [26], and lack acceptance criteria that could provide focus to the team [27];
- Unclear requirements make it difficult to developers to understand the customers' needs [23], and make quality assurance harder [26];
- Non-functional requirements, like security and performance, are neglected because practitioners focus on user stories, which define product features [22];

- Defining contracts for projects following agile principles is difficult [25], due to requirements' high susceptibility to changes, which make budget and time estimation a complex activity [22];
- Customer unavailability for requirements negotiation, clarification and feedback can result in increased rework and project delays [7] [22] [27];
- Knowledge sharing can be threatened by some practitioners' attitudes and personalities: people might not want to share, not be good at explaining, and be too shy to ask or provide help [28].
- Knowledge can be lost if practitioners do not understand, ignore, or forget what was discussed in a meeting [28].
- Cross-functionality is hard and expensive to achieve, since it relies on having practitioners competent in a broad range of skills [9]. Furthermore, it can be jeopardized when practitioners select simple and familiar tasks during self-assignment;
- Roles boundaries are more diffuse when compared to other methodologies [9];
- Convincing senior management to adopt agile methods might be difficult [27], and once they accept they might interfere too much on the process, impeding teams from self-organizing [6] [7].
- Progress tracking is a difficult activity, and it is not clear how to use results from tracking [6];
- Over commitment to tasks, deriving from the difficulty in performing estimation [6];
- Poor business knowledge can lead the customer to ignore practitioners technical strengths and losing confidence in their abilities [9];
- Neglecting design and architectural issues, which can lead to problems in later project stages and increases in cost [22, 29];
- Difficulty in implementing agile, rooted on poor process customization (which often means skipping steps) [26];
- Different interpretations of agile lead to different approaches [26];
- Focusing only on the agile practices, and ignoring agile principles [9];
- Difficulty in integrating non-development functions, like research [6];
- Implementation of personal performance rewarding, acting against team-centric agile thinking [9] [26];
- Losing discipline for not focusing too much on processes and tools [6];

- Forcing the methodology too much [6];
- Interdependencies between activities and artefacts [27], which can lead to delay and reworks;
- Resistance to adopt agile/Scrum methodologies [5, 7, 26, 30];
- Lack of Scrum knowledge [20];
- Lack of practitioners motivation to use agile methods [9]

Teams' productivity and software development process' efficiency can greatly influence the final product's quality. As claimed by Cho [23], bad documentation made it difficult for developers to understand the project requirements, and as they asked many questions this would result in loss of time. In addition, Mohammad *et al* [11] considered one of the main weaknesses of agile methodology as being the struggle to coordinate teams for large projects.

Challenges for particular contexts, like geographical dispersion of teams (known as global development agile [31, 32]), are outside the scope of this research. Still, in some cases, agile might not be able to provide the required support for projects [25].

The following sections present a deeper analysis and validate some of the identified challenges.

2.1 Interviews Results

In the scope of this research, a set of five interviews was conducted managers of five different companies with the intention of clarifying the issues leading to a poor software development process and to support the described problems. These interviews were semi-structured, allowing the five interviewees to provide insights not covered by the script. This section presents the main findings, but a more detailed analysis can be found in another publication [33].

All interviewees are managers responsible for one or more of their IT processes and belong to organisations in the fields of IT, and/or telecommunications, except for one that belongs to the gaming industry. At least three of the companies used Scrum with only one claiming to be compliant with all Scrum recommendations and the others adapting the methodology to their workflows.

Despite the interviewees' assertion that they follow these processes, they admit the existence of several failures:

- Lack of communication amongst departments or practitioners as result of poor distribution of responsibilities between those;
- Miss the testing phase, justified by either lack of time allocated for it or incorrect resolution of bugs;

- Lack of practitioners' commitment in performing their work correctly which results in poorly detailed requirements and documentation;
- Lack of cooperation.

Lack of commitment was justified with the feel of boredom associated with the assigned tasks and/or lack of time affecting the project performance and implementation times. Also, poor communication was mentioned as a possible producer of misconceptions which would negatively impact projects.

2.2 Survey Results

A survey was developed with the goal of obtaining more feedback from practitioners. Aside from working in the software development area, the targeted practitioners are also people who used Jira Software in their projects.

397 answers were retrieved from respondents distributed in several organisations around 30 countries. With a response ratio of 0.20, an overwhelming majority of the respondents were Portuguese (88.6%) males (97.7%) working in the IT sector (90.9%). Also, approximately 30% of the practitioners are individual contributors. The main findings of this study include:

1. Teams' compliance with Scrum best practices (i.e., degree to which teams apply Scrum techniques in practice) was classified as either sufficient (46.9%) or good (43.8%);
2. Lack of Scrum knowledge (37.5%), excessive time consuming of Scrum events and activities (34.4%), and discomfort affecting people lacking communication skills (31.3%) were the main reasons affecting motivation in complying with Scrum best practices.
3. All Scrum events were considered important, being classified in the following order: Sprint planning; Daily meetings; Sprint reviews; and Sprint retrospectives.
4. Motivation to attend Scrum events was rated in the same order as before, suggesting that teams are more motivated to perform the Scrum events that they perceive as more important.
5. Scrum activities like creating user stories, creating sprint backlogs, prioritising tasks, and estimating task effort are perceived as being important, but respondents are not motivated to perform them.

2.3 Problem Statement

Multiple problems were common to the three conducted analysis. The literature analysis provided a set of general problems previously seen in Agile projects, which were then prioritised with the aid

of the interviews and surveys. From those, we considered the main challenges as the practitioners' commitment to their work, poor communication, and inability to comply with Scrum techniques such as work planning and estimation.

Considering the previous analysis we define the current problem on the software development process as **the practitioners' lack of motivation and commitment to adopt the Scrum methodology and its advocated practices.**

3

Theoretical Background

Contents

This chapter comprises all the concepts that are relevant for this study. We introduce agile methodologies, followed by Scrum as one of the most popular methodologies. Moreover, gamification is addressed as a solution for the research problem.

3.1 Agile Methodologies

Software development is a process to deliver products in a more rapidly, and cheaper way [11]. Although methodologies, such as the waterfall model, were highly used for large-scale projects because of their predictability, scalability, and high assurance, they ended up being too heavy and not adaptable to the continuous changes in business requirements. To embrace change, focus on people and customer's satisfaction, ASDMs were developed. It advocates development as based on iterative development since its requirements and solutions are changed through collaboration between organisations and teams [11].

Highsmith [34] used three characteristics to define agile development: a chaordic perspective, collaborative values and principles, and a barely sufficient methodology. The chaordic perspective (chaos and order) arises from the fact that the level of unpredictability in agile projects tends to increase as requirements and goals change throughout time. While traditional methodologies aim to standardise people to the process, agile aims to capitalise each individual's strengths. It focuses on collaboration and interaction between team members and the project's customers. The author still mentioned that agile was a barely sufficient methodology. He claims that there must be a balance between flexibility and structure, and that cost reduction is achieved by streamlining the process.

Moreover, with today's larger and distributed agile teams, the number of communication lines that need to be maintained could reduce the effectiveness of informal face-to-face communications and review meetings [25]. However, with the evolution of technology and decreasing costs during the past few years, distributed environments are easier to build considering that communication means are cheaper, and more available.

3.2 Scrum Methodology

As Scrum is one of the most popular agile methodologies, it was chosen as the scope of this research. Scrum is an agile process used to manage complex software and product development with iterative and incremental practices [23]. Schwaber *et al.* [35] defined Scrum as lightweight, simple to understand, and difficult to master. Its workflow is composed of Scrum Teams and their associated roles, events, and artefacts.

Scrum teams are self-organized and cross-functional, choosing how to best accomplish their work, rather than being directed by others [35]. The roles that a team member can represent include:

- **Product Owner**, responsible for managing the Product Backlog i.e., accepting ideas, prioritizing them, and inserting them into the Product Backlog (explained below);
- **Development Team**, responsible for implementing the work contained in the Sprint Backlog;
- **Scrum Master**, who guarantees that Scrum values, practices and rules are understood, and enforced.

Work is performed iteratively through sprints: time-boxes of four weeks (or less) during which a releasable product increment is created. Each sprint has a specific Sprint Goal and consists of several events with specific purposes:

- **Sprint Planning**, a meeting where the Product Owner and Team discuss the next sprint;
- **Daily Meeting**, a meeting to assess the project status since the last one, and decide work to do on the next 24 hours;
- **Sprint Review**, a meeting held at the end of the sprint to inspect the most recent instance and adapt the Product Backlog if needed;
- **Sprint Retrospective**, a meeting held at the end of the sprint to discuss it and define what can be changed and what can make the next sprint more productive.

People and ceremonies are indispensable elements of Scrum but, to manage knowledge and keep the work documented, artefacts are required. These can be of three types:

- **Product Backlog**, which gathers functional and non-functional requirements, sorted by priority and kept by the Product Owner;
- **Sprint Backlog**, which gathers the high priority items from the Product Backlog selected by team members, being those broken down into tasks;
- **Increment**, which is the sum of Product Backlog items that are completed during the sprint.

The Scrum process begins with the Product Owner creating the Product Backlog with all the product's requirements ordered by priority. Each sprint starts with the Sprint Planning meeting where requirements with higher priority are selected and compose the Sprint Backlog. Items are then broken down into tasks and assigned to members of the development team.

During the sprint, Daily Meetings are held to assess the project status and, when the sprint finishes, the Sprint Review meeting is held to inspect the new increment and adapt the Product Backlog if needed. In addition, the Sprint Retrospective meeting is conducted to discuss lessons learned, and improvements, for the following sprint.

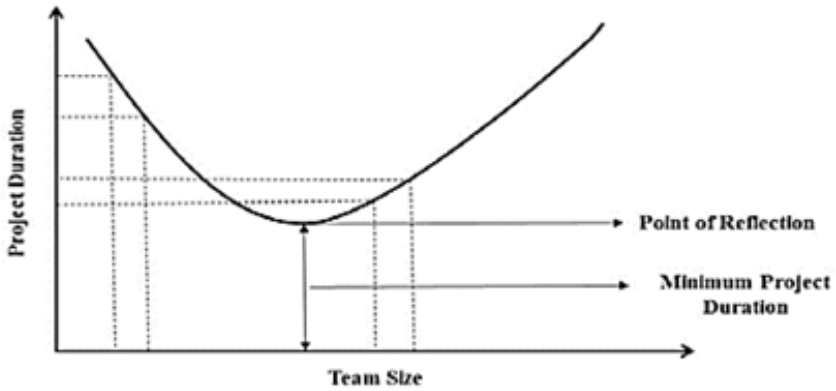


Figure 3.1: Impact of team size in project duration (adapted from [2]).

3.3 Metrics

The pure development of the product itself is not enough to achieve the best possible deliverable, and software metrics play an important role in good software engineering. This measurement is used to assess situations, track progress, and evaluate effectiveness [36].

In what regards software engineering, it is important to measure if requirements, design and code are in accordance to what was established for the project as those should meet the project requirements. The chosen measurements must be performed correctly, and rigorously, to retrieve the best knowledge on the current state of the project.

The definition, and measure of metrics, is relevant to keep track of projects' and individuals' performance. A major challenge faced by the project manager is to identify the key software metrics to monitor the project [2]. Mridul Bhardwaj *et al.* defined the key software metrics as being **size**, **effort**, **project duration**, and **productivity**.

Size is the measurement of software functionality that is being delivered or expected to be delivered [2]. Size estimation allows the project manager to compare the current project with previous completed ones in terms of complexity to predict the performance of the former.

Effort is usually measured in "person hours, person months or person days" [2] but "person hours" is defined as the most suitable considering that the number of standard working hours varies in different countries.

Also, project duration must be considered alongside effort because, and according to Bhardwaj *et al.* [2], a project manager must understand that the overall project duration can be reduced by adding more team members. However, at a certain point, increase in team size will not result in reducing the project duration. The reason for this is the fact that adding more members to a team increases integration efforts (shown in Figure 3.1), so a constant team size is preferable throughout the project.

Productivity is necessary for the project manager to predict the project's effort. Involved teams' expertise and their business processes understanding has impact on productivity considering that they are responsible for retrieving technical requirements from business requirements. It is also mandatory to assess teams that did not work together since integration efforts and collaboration issues will be higher, which can affect the overall productivity.

Kupiainen *et al.* performed a literature review to understand metrics in the context of Agile projects and to find out the metrics' benefits and effects on Agile teams [37]. The conducted study has shown that nowadays not only is important to plan and track projects and its sprints, but also to retrieve knowledge on improving quality, fixing problems and motivating people. A list of high influence metrics was defined and sorted by occurrences and importance.

Progress is also an important factor to be measured, considering the velocity (productivity is termed as "Velocity" [2], and it measures the delivered features *per* iteration) and effort estimate metrics with increased occurrences and importance (shown in Figure 3.2). Even tough projects that were considered successful measured customer satisfaction [37], the reduced number of occurrences of this metric is a sign that companies are not considering customers as much as they should.

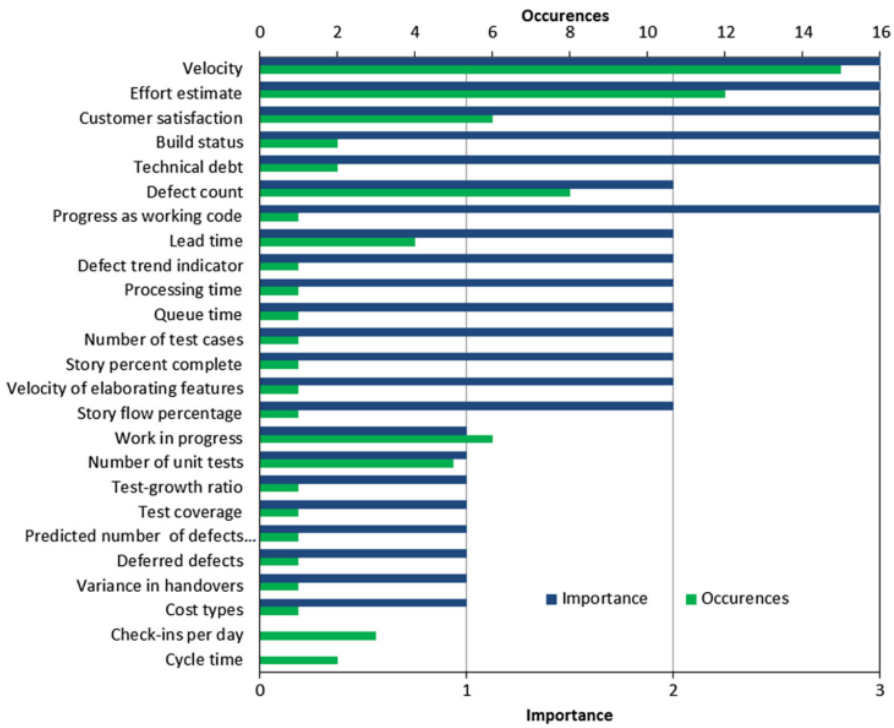


Figure 3.2: List of metrics with values for importance and occurence (adapted from [2]).

3.4 Gamification

Improving practitioners' motivation, and, consequently, their satisfaction and well-being is a challenge that researchers have been focusing on. One of the strategies to tackle this challenge is the use of gamification.

3.4.1 Definition

Gamification is the use of game mechanics and design elements in non-game contexts. It uses elements of games for purposes other than their normal expected use as part of an entertainment game [38].

Gamification focuses on the "game" part instead of focusing on the "play", as the former provides a set of choices and the latter is about "freedom", and being creative. These choices must include the option of being part (or not) of a specific gamification initiative.

Although it has been used in several areas, (e.g. persuasive technology) video games and game aspects have been studied as potential means to shape user behaviour in directions intended by the system designer [39]. The motivation issue is tackled by figuring out what lures the user, how to engage him/her, and provide appropriate feedback in between, increasing the level of engagement to finally achieve what is a victory for the user [40].

In their research, Werbach and Hunter [3] defined three gamification elements:

- **Dynamics**, which represent the highest level of abstraction of a gamified system (e.g. appealing to users emotions);
- **Mechanics**, which represent processes that make use of components and engage the users (e.g. giving positive messages for specific behaviours);
- **Components**, which consist of items that are used to complement the previous elements (e.g. points and badges awarded for specific behaviours).

Usually reward systems and scoreboards are used when implementing such ideology. As an example, Nike gamified everyday activites with their FuelBand to keep users active and exercising. They created levels and missions to be played against the clock, and users' progress would be tracked by NikeFuel points. The social element is important here, as users are now able to share their runs and compete against others for specific milestones.

Essentially, to gamify is to listen to what games have to teach us, and use that knowledge to improve further backgrounds.

3.4.2 Motivation

Although gamification has many benefits, its implementation must be performed correctly for it to be efficient. For that purpose, we must understand the way practitioners behave and how to measure these behaviours.

3.4.2.A Changing Behaviour

According to Herranz *et al.*, in change environments all the people in the organisation are required to change their attitude, acquire and practise new behaviours and skills aimed at improvement and better performance [30]. People's attitude is of major importance when implementing any initiative and that is why resistance to change is so difficult to overcome when managing large organisations. The authors claim that gamification leads to motivation improvement and makes it easier to accept change.

However, adopting gamification is not just "throwing" badges and rewards at the users, it requires knowledge on people's behaviour and how to influence it. Behaviourism emphasises the role of environment factors in influencing behaviour [41] i.e., that the environment shapes an individual's behaviour. It states that people learn new behaviours through classical or operant conditioning, but only the latter is considered for this study as the classical conditioning involves reflexive/involuntary responses, such as phobia-triggered behaviours. Operant Conditioning is a type of learning in which the strength of a behaviour is modified by its consequences. It represents roughly change of behaviour by the use of reinforcement which is given after the desired response [42]. Operant Conditioning can be applied with three types of elements:

- **Neutral Operants:** Responses from the environment that don't change the probability of a behaviour being repeated;
- **Reinforcers:** Responses from the environment that influence a person to repeat the behaviour;
- **Punishers:** Responses from the environment that influence a person to not repeat the behaviour.

Using a punishment strategy is not the right way to promote change because it creates fear and pressure among the users [42], so reinforcers should be used instead. However, to understand their strengths, the difference between intrinsic and extrinsic motivation must be clear. Werbach and Hunter state that wanting to do something is called intrinsic motivation because it lies inside the activity [3]. On the other side, feeling that you need to do something involves extrinsic motivation, because the motivation lies outside [3]. To fuel this extrinsic motivation, Points, Badges and Leaderboards (PBL) are often used in multiple solutions.

The Self-determination Theory (SDT) endorses the power of intrinsic motivation as the strongest. It suggests that human beings are inherently proactive, with a strong internal desire for growth, but that

Table 3.1: Categories in which a person's needs are split, according to SDT [3].

Category	Description
Competence	Effectiveness in dealing with the external environment.
Relatedness	The universal need to interact and be connected with others [43].
Autonomy	The innate need to feel in command of our life.

the external environment must support it [3], focusing on what people need to boost their inner growth. Table 3.1 contains the three categories in which those needs are split.

Usually many companies focus too much on points' systems (a phenomenon called *pointification*) to motivate employees, which creates a sense that there is no value on doing tasks just for the "sake of it". This is the moment where employees stop enjoying what they are doing unless a reward is given, causing extrinsic motivation to overlap the intrinsic. For that reason, an efficient gamification strategy must present a balance between points, badges and leaderboards along with game dynamics to keep users engaged.

3.4.2.B Fogg's Behaviour Model

Fogg's behaviour model states that behaviour is a product of three factors: motivation, ability and triggers [4]. Failing to understand those factors increases the chances of failing to design a persuasive system. The author claims that for behaviour to be changed those three factors must be present before its occurrence.

A person with high motivation and ability will be more likely to perform the target behaviour. However, if the same person has low ability and/or low motivation it will be less likely for the target behaviour to be achieved due to the lack of one (or both) the factors. Although a person high on both factors is more likely to achieve the target behaviour, in fact, without an appropriate trigger, behavior will not occur even if both motivation and ability are high [4]. Fogg defined three core motivators:

- **Pleasure/Pain:** This motivator is immediate and people respond to what is happening in the moment;
- **Hope/Fear:** This is related to anticipation of an outcome. The former is the anticipation of something good while the latter is the anticipation of something negative;
- **Social Acceptance/Rejection:** People want to do things to win acceptance or even to avoid being rejected.

Although pain and fear are not recommended to be used, simplicity is an additional way to motivate people considering that, if something is easier, a person will be more likely to do it. People like simplicity and making a behaviour simpler can be a strong motivator. Despite the power of simplicity being

something for which Fogg stands, for an individual's behaviour to happen, (s)he must be triggered with simple elements. Triggers are used to persuade someone to follow a desired behaviour, and timing is very important considering that not every moment is the most appropriate. These triggers can be of three types:

- **Spark:** If someone does not feel motivated to perform a behaviour, then a trigger should be combined with a motivational element (core motivators) and must be presented at a moment in which is possible to act;
- **Facilitator:** When users have high motivation but low ability, an easier behaviour should be triggered;
- **Signal:** When someone is at the top of motivation and ability, (s)he just needs a reminder to perform the target behaviour.

3.4.3 Gamification Design Framework

To better design and develop a solution, it is important to understand how to promote behaviour change in a positive way. Fogg's Behavioural Model [4] can be used to understand how to persuade practitioners to follow particular behaviours, and to adopt certain techniques.

Learning the level of engagement, and the way practitioners are behaving, requires metrics that can be defined to measure those behaviours, and to track their productivity. With that information we can understand more effectively which game components, and dynamics to fit to our system. Also, it is essential to learn what works, and what should be changed so the users are kept satisfied and engaged.

Different approaches, based on a set of steps or guidelines, can be used to design a gamification system. For this study, we used the Gamification Design Framework [3] which consists on 6 steps to break down the design process of a gamified system:

1. **Define Business Objectives:** Why is this being done and how will it benefit the business?
2. **Delineate Target Behaviours:** What is meant for the players to do and what measures will be used to measure it?
3. **Describe Your Players:** Who are the people involved in the system?
4. **Devise Your Activity Loops:** How will users be motivated in what regards engagement and progression loops?
5. **Don't Forget the Fun:** Fun must be ensured to keep players engaged;
6. **Deploy the Appropriate Tools:** An idea of a possible system and what elements will it use.

The Gamification Design Framework was chosen because it not only provides the steps to work on features and pick elements, but also to understand the goals and target users behind the tool. This framework begins by understanding the business, which is crucial to learn what are the objectives for the gamification system, and what problems is it meant to tackle. As stated before, a gamification strategy is not exclusively about points, and rewards, so we should further define the behaviours we want our users to follow, and learn their features, and needs. Only then we can decide how to motivate, and engage them for the goals to be achieved.

4

Related Work

Contents

There are several studies exposing gamification implementation cases. Learning the lessons from those examples is a way to comprehend what benefits can be retrieved from specific contexts and what are the downsides of some actions.

Hamari *et al.* performed a study on the success of gamification by conducting a literature review [44]. They mostly tested PBL components and noticed that, although gamification shows potential, it may not be long-term but instead could be caused due to a novelty effect. Removing gamification might also be a wrong choice due to the sense of losing everything employees earned before (e.g. points, badges, etc.).

A negative is example is the one from Disneyland hotels in California, where large flat-panel monitors were installed in laundry rooms to show employees how quickly they fulfil their tasks and how their speed compares to co-workers [3]. The outcome was a tense work environment with employees skipping bathroom to raise their scores and afraid that the lower-positioned ones would lose their jobs. Workers' performance became more quantitative and less qualitative, which would eventually backfire on the hotels' management.

This is an example of exploitationware[1] – situation where organisations ask for loyalty, but they reciprocate that loyalty with shams, counterfeit incentives that neither provide value nor require investment. It shows that PBL components can be powerful, and the wrong use of them can create negative effects.

4.1 Gamification in Software Development Education

Some studies on gamifying learning experiences were also performed and that knowledge can be meaningful for this study. Domínguez *et al.* [45] claimed that to create a gamification system that increases student motivation it is necessary to focus on the fundamental elements that make games appealing to their players.

Understanding what are those elements and how they influence success and failure of the players is important to prevent them from feeling frustration when failing to perform some task. Also, having multiple players interacting with each other through a game encourages cooperation to achieve a common goal. This is a similar situation to one with team members cooperating in a software development project.

Assessing this circumstances in the scope of assigning new developers to projects might be relevant for those who are shifting to a new methodology and thus need to learn new methods. To test the gamification influence on learning, the authors designed a gamified educative experience that included a system of rules in which each student had to obtain skills and a virtual reward system, based on achievements and badges, to influence motivation.

[1] Exploitationware - http://bogost.com/writing/exploitationware/ (Accessed in 12/05/2017

Results have shown that although rewards and leaderboards served as motivation, that did not happen for every student. Another problem was task evaluation as many students did not complete gamified exercises because they thought that it was a waste of time to capture and upload screenshots of their work. Also, not providing immediate feedback failed to give them the *fiero* moment (what we feel after we triumph over adversity [46]), impacting one's motivation positively.

Dubois and Tamburrelli [47] tested a strategy with engineering students by assigning them to a software development project in which the goal was to acquire ability to design, implement, document, and test software. Using a score system, the obtained results shown an increase in the quality of software artefacts because students wanted to keep their rules compliance score. This happened because a code checker software – Sonar – would give them reports for every code submission. By allowing half of the teams to check the metrics of all the others they concluded that competition had a positive impact on those teams, which presented better compliance with the metrics.

However, the authors were not able to show how competition could affect specific aspects of a project, and they concluded that integrating gamification in the software development process was not the difficult task. Developing a gamification method and predicting its effect is much more difficult [47] and increasing software quality is a natural goal of this kind of gamification strategy despite the challenges inherent to it.

4.2 Gamification in Software Development Industry

Snipes *et al* [48] implemented a tool focused on motivating the adoption of good practices, called Blaze. The tool is installed in Visual Studio, and collects data during a week to provide feedback on what users have been developing. According to Snipes *et al.*, when discussing gamification, concerns are raised about replacing an activity like software development, which is intrinsically motivated, with extrinsic motivation provided by points and achievements [48]. Blaze did not considered tasks that are fixed requirements of the job because they would reduce intrinsic motivation.

The obtained results after the use of Blaze showed that two out of six developers responded positively when starting to receive points, which was not enough to allow the implementation of such method. The authors found that developers would appreciate more formats of feedback such as charts and graphs, meaning that users need diversity and choice. With this, there is a necessity to balance the types of feedback with game dynamics which is important to kill boredom and keep users engaged.

Herranz *et al.* developed Gamiware, a gamification platform that is intended to increase both motivation, and employee engagement in software initiatives [49]. Its purpose was to assist the implementation of another framework named by the authors as "Gamification Framework" [30]. Gamiware was designed to be project and process independent [49], and it was meant to tackle high implementation costs.

Upon a full framework iteration, Gamiware comes into play to support the implementation phase by applying game elements which, along with mechanics and dynamics, leverage users' engagement in the process. For this gamification platform, three types of roles were defined:

- **Admin:** Its main task is to perform initial configurations based on business goals, software goals, and main tasks;
- **Gamification Master:** His/her task is preparing the gamification project inside the organisation and the specific project. Also, he/she is given the task of configuring game mechanics appropriate to that specific project, and to assign tasks to users;
- **Users:** General users are assigned with tasks and each has access to a customised dashboard that displays information dependent on his/her role. Their function is to use gamification techniques and present challenges associated to assigned tasks, initiating a competition with another user.

Participants in this study found that their motivation had increased and most of them felt satisfied with their performance. However, results also pointed lack of relaxation and fun in performing the assigned tasks while competing against others. This situation created stress upon the users although the authors claim that motivation was not impacted, and that collaboration was an effective solution to those problems.

4.3 Applications

There are many gamification solutions developed as commercial tools.

Jiraffe[2] is a gamification app that claims to make boring tasks funnier and, hence, more likely to be performed. To get the users to achieve a higher state of motivation it transforms projects into games, naming them as "adventures". For each adventure, several characters with different statuses (e.g. developer and tester) exist and each of them can be assigned with points, badges and enter the leaderboard to compare their skills.

An advantage presented by this application is the configurable feature that allows an admin to adapt it to any project necessities. Also, rules can be created and edited to assign different rewards for different kinds of tasks which, again, reveals a certain adaptability to different projects.

GetBadges[3] which aims at improving productivity, and engagement, while providing fun. Players are provided with its own profile and avatar with an experience points' system presented with difficult levels through which the users must run.

[2]Jiraffe - Available at: https://marketplace.atlassian.com/plugins/com.bugpotion.jiraffe/server/overview (Accessed in 11/10/2017)

[3]GetBadges - Available at: https://marketplace.atlassian.com/plugins/io.getbadges.plugins.jira/cloud/overview (Accessed in 11/10/2017)

A graphic dashboard is available with all the necessary information including an activity feed, team members, and their badges, being those available in more detail for other users to check. Competition is introduced with a leaderboard and every user can choose a guild (e.g. developers or testers), inducing a sense of relatedness and fuelling a collaboration environment between players.

GetBadges' advantages against Jiraffe consist of more appealing visuals and it seems to be a better developed Jira App. Although both allow a certain level of configuration, the former enables integration with tools as Bitbucket[4] and HipChat[5] which could extend its features beyond the predefined ones.

Regarding Scrum documentation, a challenge identified in Section 2 involves the lack of detailed tasks, and documents. Karma for Confluence[6] aims at motivating employees with points, and badges so they feel a higher predisposition to keep the necessary documents updated. Included, is the possibility for colleagues to award medals to others which promote knowledge exchange, and raises the team's spirit.

4.4 Discussion

Previous research has proved the benefits of gamification as this is an efficient approach to increase teams' motivation and satisfaction. However, to harvest its potential is not a task that many can easily accomplish. Although some of the developed applications have shown positive results, there are still constraints in what regards stress, extreme competition and novelty effects. Moreover, cases of gamification applied to software development students shown that some youngsters demonstrated lack of interest and saw it as extra work to be performed.

Most of the previous research lack a proper validation. Even though some of the cases are tested with software development students, they cannot present results comparable to what practitioners would show. Moreover, previous cases make use of simpler gamification elements and don't consider all the users' needs. When developing a gamification strategy, important factors exposed in previous research should be considered. These factors and lessons to retain from chapters 2 and 4 are presented in Table 4.1.

Finally, a good start for developing a gamification strategy would be to first conduct a clear analysis of the implementation context and avoid making the same mistakes that lead to failed strategies.

[4]Bitbucket - Available at: https://bitbucket.org/ (Accessed in 11/10/2017)
[5]HipChat - Available at: https://www.hipchat.com/ (Accessed in 11/10/2017)
[6]Karma for Confluence - Available at: https://marketplace.atlassian.com/plugins/de.communardo.confluence.plugins.karma/server/overview (Accessed in 11/10/2017)

Table 4.1: Factors identified in the previous gamification contexts.

Factors	Interpretation
Competition & Leaderboards	We should be careful when implementing competition because it can put too much pressure on teams. The intention must be clearly exposed to those involved and consequences should be taken into account to avoid killing performance.
Points & Badges	Although they are used many times, they do not always deliver the expected results. Each people has different motivation needs and the abuse of points and badges can result in excessive extrinsic motivation. A consequence of this is having people only working for the prizes which will not happen when those are removed.
Extra Work	If extra work is necessary to accomplish the gamification strategy then it won't be well accepted by most people.
Feedback	Feedback must be given immediately after an accomplished action. For example, if a person finishes a difficult task, receiving a "good work!" message 15 minutes after that won't have the same effect as receiving it at the moment.
Gamification Removal	Sometimes gamification strategies might not work as expected. However, removing them and taking all the given prizes away could make the players feel like they worked for nothing. They could feel cheated and, consequently, less motivated.

5

Proposal

Contents

5.1 Objectives

Considering the research problem and previous studies on gamification, **this research proposes the development of a software tool to gamify Scrum projects, with the goal of raising teams' engagement, and motivation to adopt the Scrum methodology.**

There is a need for development teams to not only be more productive, but also to adopt Scrum techniques, and good practices. The main objective of this proposal is to increase practitioners motivation to adopt those techniques, and comply with the Scrum methodology.

The mentioned objectives are based on the most relevant problems that were analysed in the previous chapters. Moreover, the gamification strategy, based on the Gamification Design Framework, is detailed in the following section.

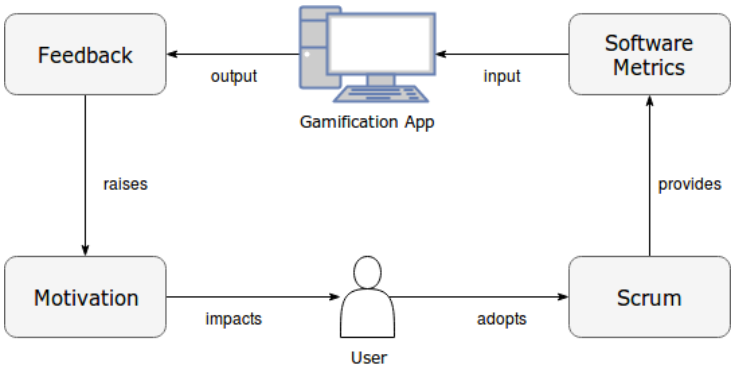


Figure 5.1: Input-process-output diagram exposing the cycle in which the practitioner's motivation is fed by the gamification system, leading to adoption of Scrum techniques.

5.2 Strategy

The idea behind this research's proposal is to implement a gamification strategy (detailed in Section 5.2) as a software tool. When a user of the system performs an action it is measured by specific metrics that are used as input to the software. These metrics are processed and the output is given in the format of feedback (e.g. congratulations message), and/or rewards (Experience Points (XP) and badges). Balance between positive feedback and the use of rewards for specific actions (or collections of them) aims at keeping the user engaged as his/her intrinsic and extrinsic motivation are fairly sustained. As seen before, a user who feels more motivated will feel a higher predisposition to keep and/or adopt a new behaviour. From then on, the cycle can restart and keep fuelling the user motivation. Figure 5.1 displays the proposal with an input-process-output diagram.

5.2.1 Define Business Objectives

As part of building an efficient gamification system, the goals we want it to achieve must be clearly defined. The main objective for our system is to increase practitioners' motivation to adopt the Scrum methodology, and its practices.

Based on the analysis of the research problem, and related work, we defined a set of objectives for our gamification system to achieve:

- Improve the quality of tasks' specification, including general information (e.g. type, description, etc.). Meeting this objective would provide valuable knowledge, and reduce tasks' ambiguity;
- Reduce rate of unassigned work;
- Increase the rate of estimated tasks, important for practitioners to plan their schedules;
- Increase the rate of completed tasks per sprint;
- Improve communication, and team cooperation using collective goals, and rewards.
- Increase meetings' attendance;
- Track projects' performance, and keep users so informed about their productivity.

To accomplish the defined objectives, our system's users must follow specific behaviours.

5.2.2 Delineate Target Behaviours

Based on the defined objectives, a set of target behaviours must be defined. This behaviours will be supported, and measured, with a set of metrics that translate them into quantifiable results. To profound the metrics scope, we defined two contexts to which they can belong:

- **Sprint:** It provides knowledge on how behaviours change through sprints, which depicts a global view of the project;
- **Assignee:** It focuses on work performed by each practitioner, providing individual results, and knowledge on how motivated is each of them.

Table 5.1 maps all the objectives with behaviours, and metrics.

To improve tasks' specification quality, two behaviours are needed: to **complete more fields in tasks' specification**, and to **provide clear descriptions**. Both behaviours would improve knowledge sharing among practitioners, and provide more complete documentation which might be later necessary (e.g. understand why some old task was done in a particular way). We measure this behaviours by

calculating the **average of completed tasks specification's fields**, but a future improvement would be to perform text analysis of descriptions to evaluate them.

When specifying a task, a practitioner should also **assign it to someone**, and **provide an estimation**. Not only this will improve a project's organisation but also allow practitioners to plan their work considering the effort they need to put in each task. We can measure both behaviours by retrieving the **rate of assigned/unassigned tasks**, **estimated effort (in days)**, and the **rate of estimated/not estimated tasks**.

With better specified tasks, sprints can be better planned, and tasks' requirements be better comprehended. As consequence, the number of completed tasks per sprint is expected to increase as practitioners **resolve all tasks by the end of the sprint**. However, projects have problems, and barriers, so we aim to measure this by retrieving the **number of tasks (specified by total, resolved/unresolved, reopened, and persistent)**, **number of completed sprints**, and **sprint velocity**. Persistent tasks are the ones that go from a sprint to another without being resolved.

Scrum projects contain multiple meetings throughout its lifetime. From planning to evaluating, meetings are performed with practitioners to discuss relevant subjects, and understand the how well the project is going. For that purpose, our gamification system should allow its users to **check-in to meetings**, and we could measure attendance by calculating the **rate of attended meetings**.

The remaining objectives, related to team cooperation, and project tracking, are not associated with behaviours, and metrics, so their degree of achievement will be assessed with other methods, such as interviews or questionnaires.

5.2.3 Describe Your Players

In this step we should get to know our target users and hence understand how can we motivate them.

Practitioners enjoy deciding how to conduct their tasks and being able to make useful contributions [50]. They feel motivated with meaningful, and challenging work, while being able to focus on quality and receiving individual feedback based on productivity [13].

To get users' attention, and increase engagement, there should be challenges, and goals for them to achieve [51]. In addition, doing this collectively with a good team is more likely to improve their motivation, and consequently, enhance communication.

Our players will appreciate recognition for their effort, and, for that purpose, immediate feedback is provided upon a completed task, and rewards might be awarded in case a specific milestone is met.

Table 5.1: Map of proposal's objectives, behaviours and metrics. Metrics are classified as being defined in a sprint (S), assignee (A), or sprint and assignee (S + A) contexts.

OBJECTIVES	BEHAVIOURS	METRICS
Improving tasks specification quality, which can in turn increase the sharing of knowledge among practitioners and provide documentation to support communication.	Complete more fields in the task specification and elaborate on the task description, providing useful information that clarifies its purpose	Average of task specification's fields completed (S + A)
Reducing user stories ambiguity	Elaborate on the user story description, providing useful information that clarifies its purpose	—
Reducing the percentage of unassigned tasks	Assign more tasks to a user	Number of assigned tasks (S + A); Number of unassigned tasks (S + A); Estimated effort (in days) (A)
Increasing the number of tasks completed per sprint	Resolve all tasks by the end of each sprint	Number of tasks (S); Number of resolved tasks (S + A); Number of unresolved tasks (S + A); Number of completed sprints (S + A); Sprint velocity (S); Number of reopened tasks (S); Number of persistent tasks (S)
Increasing the number of effort estimated tasks	Estimate the effort necessary to complete a task, in all tasks	Number of estimated tasks (S + A); Number of not estimated tasks (S + A)
Increasing participation in meetings	Check-in to Scrum meetings	Rate of attended Scrum meetings
Increase team cooperation with team-centric goals and rewards	—	—
Implementing project tracking by providing continuous feedback based on relevant metrics	—	—

5.2.4 Devise Your Activity Loops

This is a major step in designing a gamified system as it focuses on user engagement. Along with the defined target behaviours, the implementation of the appropriate triggers is a way to engage users and boost their predisposition to follow those behaviours. It is also relevant to consider the three types of trigger described in Section 3.4.2.B, which have different effects in users' motivation. The list of defined triggers is available at Table 5.2.

To keep a user motivated and more likely to perform the target behaviours, a feedback mechanism is used. This positive feedback is given in the format of notification messages that, for example, are given when the user finishes a task. Notifications can be of three types:

- **Reminder:** Launched when a relevant event is close to occur;
- **Achievement:** Launched immediately after an event occur (e.g. when a user attends all meetings in a sprint);
- **Feedback:** Launched in specific contexts (e.g. when a user resolves a non-estimated issue).

Triggers can then be associated with activity loops that engage users in the process [3]. Based on the trigger types proposed in Fogg's Behaviour Model (see Section 3.4.2.A). There are two types of activity loops to be considered:

1. **Engagement Loops:** They keep the user engaged by giving feedback after an action, which creates motivation that leads to more action (action creates feedback which creates motivation leading to more action, and so on);
2. **Progression Loops:** They allow players to start from the beginning (defined as "onboarding") and keep on moving by completing balanced steps while becoming more experienced.

All triggers, except the last three, can be associated with engagement loops as they are implemented to feed users' motivation. The introduction of encouraging pop-up messages as immediate feedback for specific behaviours is an example of this.

The last pair of triggers can be associated with progression loops. Their purpose is to make the user feel that (s)he is in journey where (s)he must complete small but simple tasks, or a difficult one to consolidate his/her skills. One of the examples is the use of levels to make a user feel that (s)he is progressing, and evolving their skills.

5.2.5 Don't Forget the Fun

The app design should be simple, visually appealing and present easily understandable elements. Simple icons and balanced colours along with a clean design could be complemented with:

Table 5.2: Description of the triggers used in this proposal and their type, based on Fogg's Behaviour Model [4].

TRIGGERS	
Description	**Type**
Immediate feedback, in the form of pop-up messages, appearing after specific behaviours (e.g. resolving a task) or events (e.g. unlocking a badge)	Spark
Display reminders, in the form of pop-up messages, when a relevant event is close to occur (e.g., upcoming meeting)	Signal
Display reminders, in the form of pop-up messages, when the user is close to reach an achievement (e.g., having almost all tasks resolved) or reward (e.g. close two more tasks for receiving a badge)	Signal
Give gems (virtual currency) to a user or to each member of a team for performing specific behaviours (e.g. a team resolves all tasks/user stories in a sprint)	Spark
Give XP to a user or to each member of a team for performing specific behaviours (e.g. attending a meeting)	Spark
Provide a level system based on XP gained by the user	Spark
Provide a dashboard displaying the project(s) information (e.g., tasks remaining to resolve, sprint progress)	Facilitator

- A dashboard with experience bar, featured badges, and activity feed;
- Profile configuration section with global activity feed, featured projects with general statistics, general metrics related to an user's history throughout the projects;
- XP, and badges to be given for specific behaviours, such as completing ten tasks or attending five meetings in a row;
- Gems, as virtual currency, awarded for the most complex tasks (e.g. closing all issues in a sprint). Users could later use them to buy special prizes created by the company they work at;
- Levels that provide to the user a sense of evolution or progress.

Badges, and XP are useful to boost users' extrinsic motivation, thus by awarding these rewards to them after accomplishing target behaviours could increase their predisposition to keep performing them. Also, using a virtual currency for special rewards that can be personalised, provides a sense of exclusiveness hence leading the user to feel desire for the hardest realisations. Badges can be of two types:

1. **Static:** kept by the user after a certain milestone is accomplished;
2. **Dynamic:** Can be taken away, and given to other user that surpassed the former (e.g. the best user for each sprint keeps the badge).

In addition to these, a badge can still be individual or collective. **Individual badges** are awarded to a single user, and **Cooperation badges** are awarded to all team members when completing a milestone, hence promoting a cooperation environment.

Levels can also be of two types: global, and local. Each level is mapped to a XP interval that is gradually increased for every new level. The goal is to increase the amount of XP required and, consequently, the difficulty to reach the next level. The difference between both types is the scope i.e., local levels will only apply to single projects, and global levels will apply to the user itself. To exemplify, a user can be an ”Apprendice” of a project but a ”Hero” in general because (s)he has just began that project but as worked in other projects that allowed him/her to gain experience. This choice will provide practitioners with a sense of progressing, not only as a person, but also in each project they work at.

Lists regarding the defined game components are provided in Appendix A.

Finally, users are not familiar with the features int the first time using the app. When using it for the first time, a scaffolding mechanism - mechanism that guides the user through some basic and fundamental steps - should be activated.

5.2.6 Deploy the Appropriate Tools

The game components and the implementation of features is further described in Chapter 6.

6

Design & Development

Contents

This section provides an enlightenment on what, and why, certain design and development choices were taken. The development environment is described and the tool's features are showcased along with mockups to illustrate them.

6.1 Using Jira Software

There are several software applications to support software development in many ways. Our solution was developed as a Jira App to gamify the development process.

When using JIRA Software, the Scrum Master can create new issues in the backlog that are later associated to a specific Sprint. Issues represent a problem that needs to be resolved. They can have different types, to distinguish its purpose, and the problem they aim to solve (e.g. bug fixing). Programmers can be assigned to issues that they have to complete, which helps planning, and distributing work across the team. Also, graphs and reports can be created using project data to support estimations and evaluations in order to understand not only the project but also the people involved in it.

The Atlassian Marketplace[1] provides access to Jira apps which enhance or brings new features into Jira Software. The decision to develop for Jira is not only associated to its features, and stability. The fact that it also has builded a strong user base had a strong influence on the decision. Allegedly, there are over 89.000 customers using Atlassian products and services[2].

From now on, the term "issue" will be used instead of "task".

6.2 App Features

As seen before, the idea behind the gamification strategy is to get users to feel a higher predisposition to promote adoption of good practices among teams.

Each user will receive a specific amount of **XP** for specific actions (s)he performs. However, XP requires meaning to cause impact (e.g. representing the user progression inside projects). For that reason, users can collect **XP** to achieve new **levels**, which get harder to achieve as they become more experienced. Moreover, competition encourages users to work better so they can achieve higher **levels** than their colleagues and be perceived as the most experienced.

Jira Software provides different estimation factors that must accounted for. **Story Points** (Story Points (SP)) rate the relative effort of work with the Fibonacci sequence, while **Original Estimate** represents the time (e.g. hours, days, etc.) an issue will take to be resolved. **Business Value** is a relative estimation of how much an issue is valuable to the business, and **Issue Count** is based on the number of issues in a sprint.

[1] Atlassian Marketplace - Available at: https://marketplace.atlassian.com/ (Accessed in 12/10/2017)
[2] https://www.atlassian.com/customers?page=1 (Accessed in 12/10/2017)

Table 6.1: Conversion measurements for the four default types of estimation available in Jira Software.

ESTIMATION FACTOR	
Estimation Type (1 Unit)	**Experience Points (XP)**
SP	**16**
Original Estimate (Hour)	**2**
Business Value	**1**
Issue (Count)	**10**

XP are given based on each task estimation i.e., for each unit of estimation a specific amount of **XP** would be given. The objective was to keep a fair distribution of points having the different types of estimation into consideration. Business Values was the one with more value for less effort while **Story Points!** (**Story Points!**) are relative to the other issues' effort. Conversion measurements are available in table 6.1.

Special achievements and specific milestones require a different emphasis. For example, resolving all issues before the end of a Sprint is a moment worth to be recognized. For that purpose, **rewards** are awarded for defined milestones that a user completes, hence being able to showcase them, and be aware of those achievements.

The use of rewards for specific actions encourages users to pursue specific milestones that become important to them, and have relevance inside the project. It is also possible for them to achieve the same **rewards** in different projects, which can later be used to trade for special rewards. The latter case is implemented to maintain the engagement for each new project in which the user works.

PBL components impact users' extrinsic motivation but people are not supposed to accomplish certain actions only for the rewards. This type of motivation is balanced with immediate feedback, consisting of positive, and encouraging pop-up messages displayed to users for specific moments or actions. The idea is to give users a "thumbs up", and tell them that they are doing a good job. Also, the implementation of tips in pop-up format, is a way to remind or instruct the user to perform certain important actions.

All components are meant to have customizable content, so every company gets its own personalization and adapts the app to specific needs.

6.3 App Sections

There are multiple sections for users to access information, and to rapidly understand their allocated work, their previously performed tasks, their evolution throughout time (in the project or individual scope), and their projects' global view (e.g. team members, and sprint metrics). Further on, the implemented app sections will be described.

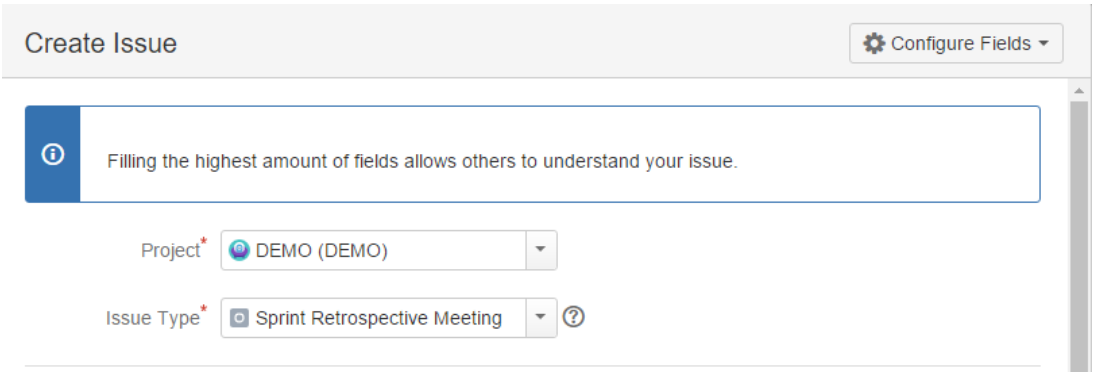


Figure 6.1: "Create Issue" form with example of a tip.

6.3.1 Project Dashboard

This section displays all project-related information, and individual stats for that project. The user has access to his/her own card that includes a personal picture, name, project role, current level, 4 featured badges, featured projects, and the amount of XP and gems. This card provides short, and rapid information regarding the user itself. Below, in the activity section, all project-related activity is shown. On the right, a burndown chart, is included with information on performed work (resolved issues), and work to perform (open issues). To impact user motivation, rewards', and statistics' sections are implemented. The latter contains the badges from which the user is closer to win. The former contains sprint-related, and user-related statistics, including:

1. **Sprint Progress:** Given by the reason between resolved and open issues;

2. **Effort:** Given by the reason between the SP a user already resolved, and the SP assigned to him/her in a sprint;

3. **Productivity:** Given by the reason between the user current performance, and his/her estimated velocity for a Sprint;

4. **Contribution:** Given by the number of issues resolved by the user and the total number of issues resolved in a project.

While the sprint progress metric provides an idea of how advanced is the current sprint in what regards performed work (by all team members); effort, productivity, and contribution serve the user with knowledge on his/her own information (see the needs identified in Table 3.1). The awareness of knowing his/her contribution is a way to understand his/her relevance among the team. Figure 6.2 represents the Project Dashboard.

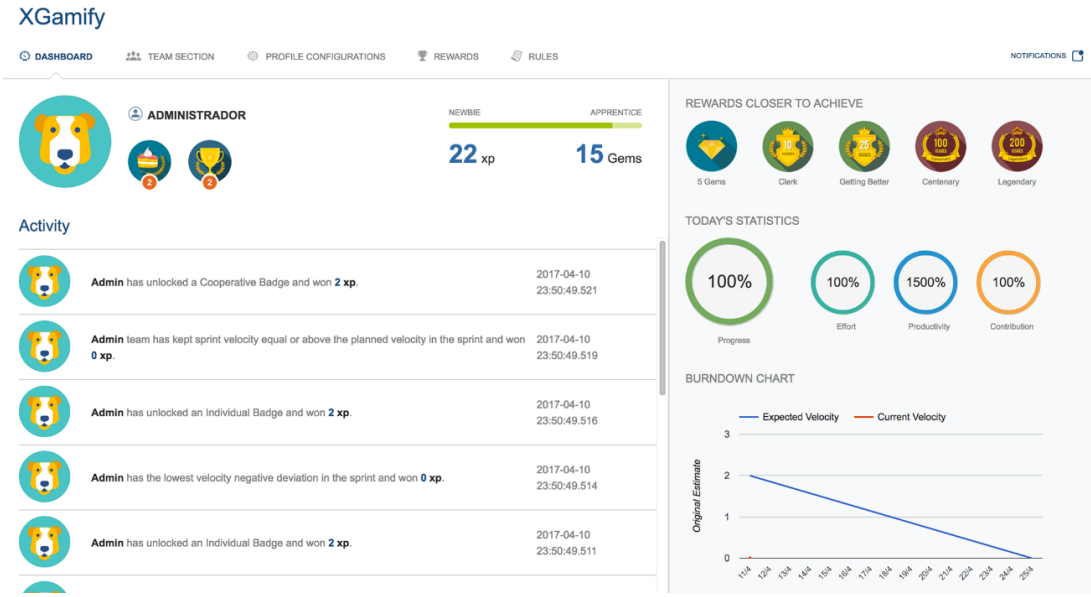


Figure 6.2: Project Dashboard, including the closest badges, activity feed, and project metrics.

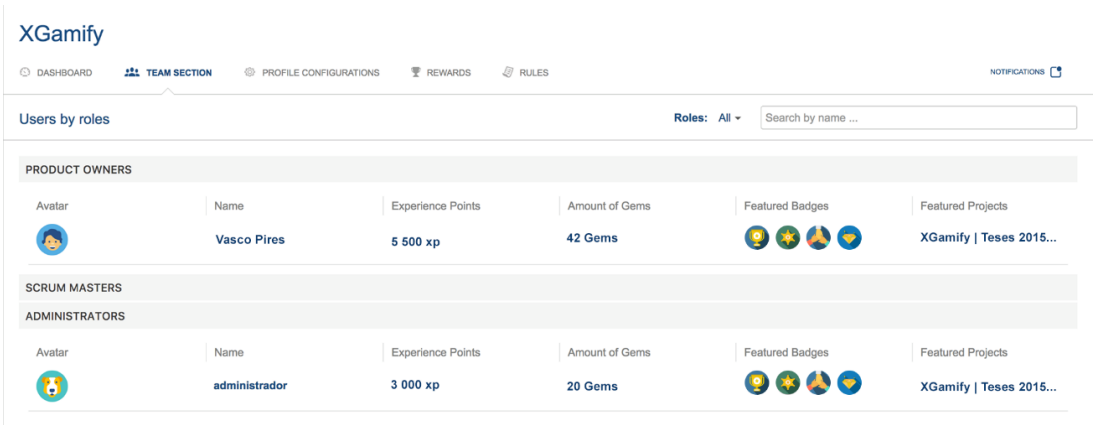


Figure 6.3: Team section, including a list of users, and a search box.

6.3.2 Team

Users are associated to teams that might change from project to project. For that reason, it is important for each user to know who is (s)he working with. Figure 6.3 represents the Team Section.

This section includes a collapsible list with all team members divided by role, including search box to simplify the process of searching for other members with specific parameters. Every team member is displayed with all available information from the user card (including featured projects), mentioned in the Project Dashboard subsection.

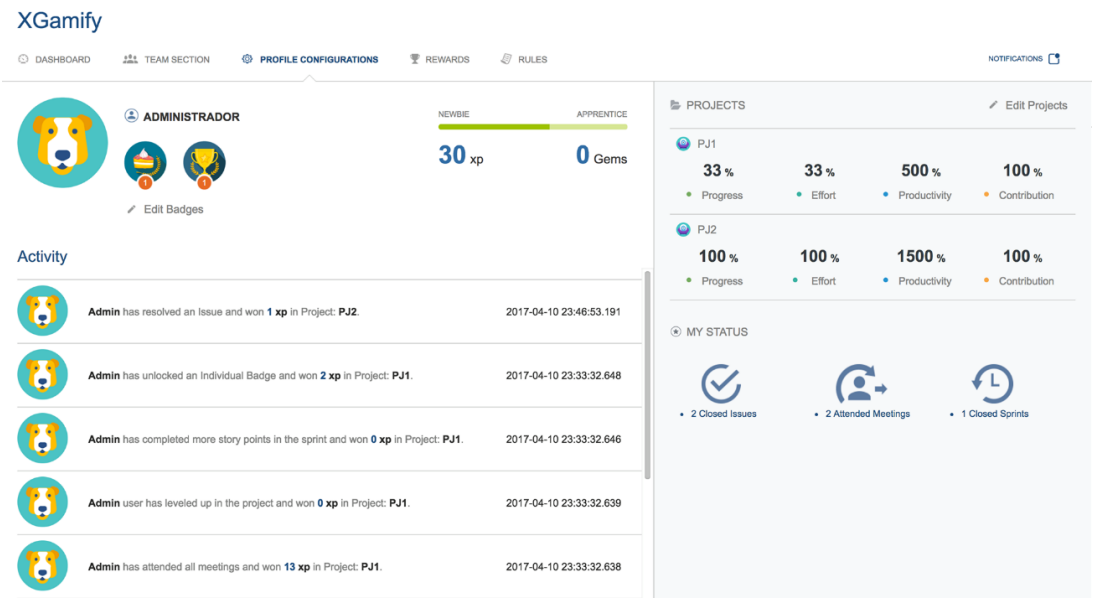


Figure 6.4: Profile Configurations section, including multiple projects, activity feed, and user metrics.

6.3.3 Profile Configurations

This section is similar to Project Dashboard subsection, with the difference of having information on multiple projects instead. The user card is the same, but the featured project can be edited. Also, the activity feed is related to all projects the user is working on, including events associated to all, sorted by most recent. This section is displayed at Figure 6.4.

Instead of having the closest badges, and the burndown chart, a section with featured projects, selected by the user, is displayed. It includes the same user's statistics as in the Project Dashboard (sprint progress, effort, productivity, and contribution). The intention is to provide a global view of the user's work.

Likewise, a section with three personal metrics is available with **resolved issues**, **attended meetings**, and **completed sprints**. This information provides the user with knowledge (s)he could not have had before since (s)he would need to access multiple reports, and Jira sections to retrieve that information. Again, our goal is to provide important information to the user without requiring multiple transitions to analyze it.

6.3.4 Rules

Rules section to contains all rules related to levels, XP, rewards, and special conditions. A collapsible list, divided by rule type, is available for the user to answer any question that emerges, as shown on Figure 6.5.

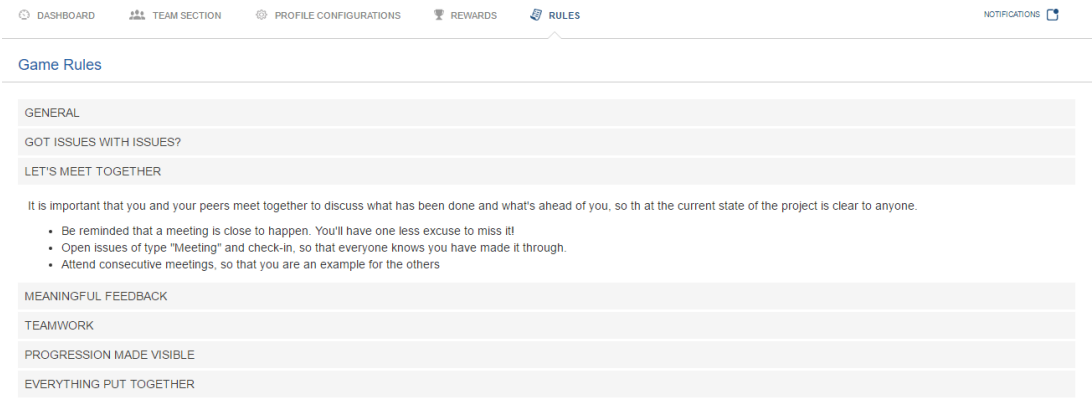


Figure 6.5: Collapsible list of rules, divided by rule type.

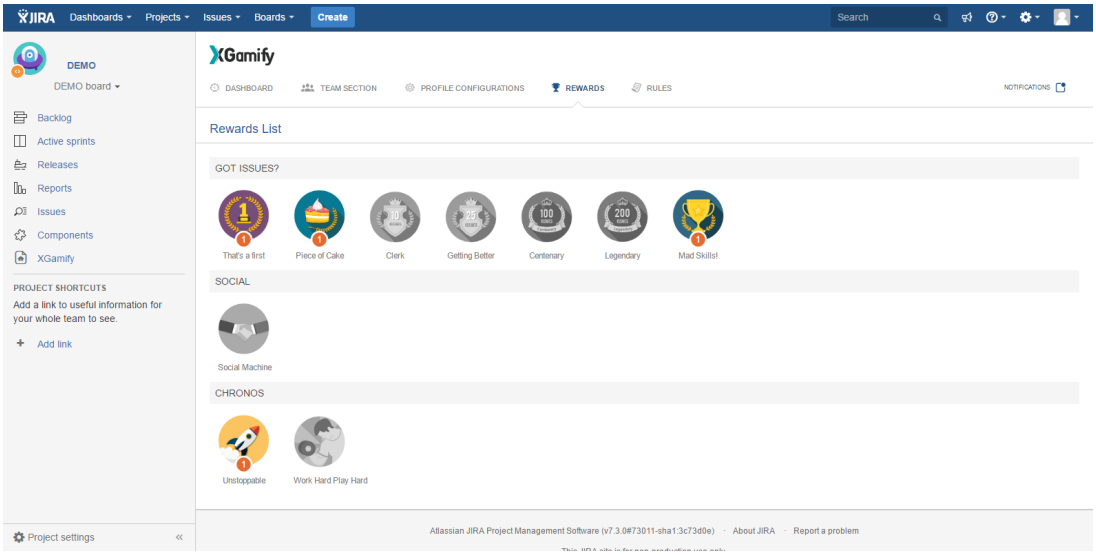


Figure 6.6: Badge section with the mouse over “Social Machine” badge.

6.3.5 Rewards

This is the section where all badges, and gems, achieved or not by the user, are displayed. All rewards that the user already have are normally displayed, with the right color saturation, and with a number below indicating in how many projects the user has achieved that reward so far. On the other side, the non-achieved badges are greyed but the user can consult the conditions (s)he must fulfill to achieve it. This app’s section is displayed at Figure 6.6, with multiple badges.

Description is available when the user hovers a single reward with the mouse.

6.3.6 App Configuration

Customization is a feature of major importance in gamification systems. The proposed app will provide multiple customization options divided by several sections:

1. **General:** Related to project selection for gamification, and estimation factors;
2. **Notifications:** Related to notifications, and tips management;
3. **Rules:** Related to rules management, including creation, editing, and removal of rules;
4. **Rewards:** Related to badges, and gems management. It is possible to create, edit, and remove rewards.

Figure 6.7 represents the global configurations in which general, estimation, tips, and rules' settings can be managed.

General settings include the possibility of disabling our app for specific projects or at a global level. It is important to provide the choice of participating since there might be users not willing to use it. In case a manager wants to reset all gamification-related data, a "reset gamification" feature is available for clearing all XP, badges, and levels for all users.

Estimation factors can also be edited. They are related to the existence of the four default ways of estimating an issue (see Section 6.2). For each of these estimations, a specific amount of points can be associated with a specific amount of the estimation (e.g. 1 story point could award 10XP). Those estimation values are converted to XP and given to the user when (s)he resolves an issue.

Rules exist to reward users for specific behaviours that the admin wants them to achieve (e.g. create a rule to give 10 XP when a user attends three meetings in a row). Rules can be applied to different classes, including issues, meetings, and rewards. Although most of the attributes are common to all rules, each class can mean the existence of exclusive ones. Common attributes for all rules include:

- **Class:** The element to which the rule is applied. There are three classes:
  - Issue: Number of issues necessary to achieve a reward, and a specific amount of XP;
  - Meeting: Number of meetings (consecutive or not) necessary to achieve a reward, and a specific amount of XP;
  - Reward: Number of achieved rewards necessary to achieve a reward, and a specific amount of XP;
- **Context:** Part of the project in which the rule is applied;
- **Roles:** The roles of the project to which the rule is applied;

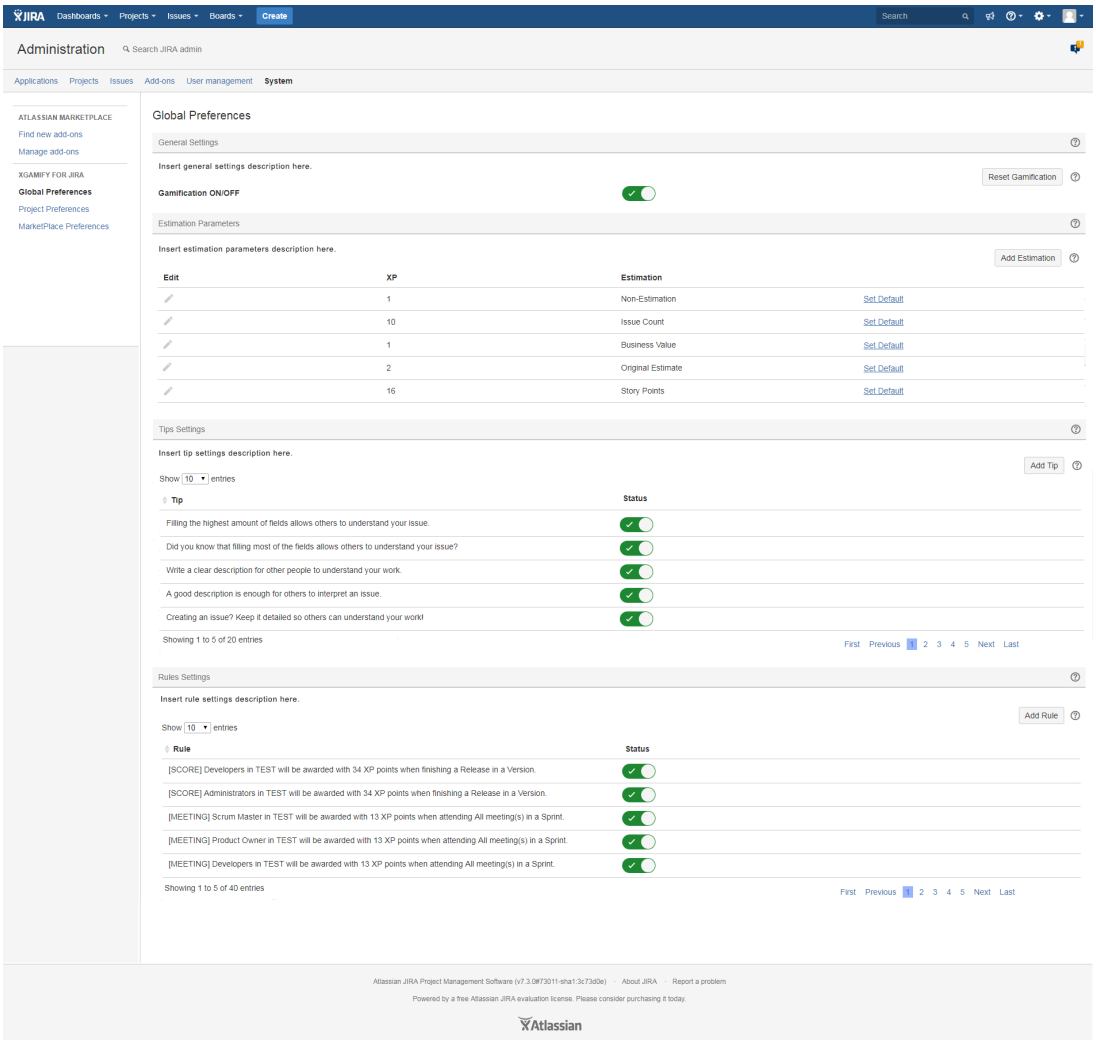


Figure 6.7: Global preferences section including settings for estimation, tips, and rules.

- **Projects:** The projects to which the rule is applied;
- **XP:** The amount of experience point a user gets for fulfilling the rule;
- **Main Type:** The main type of reward to which the rule is applied to;
- **Category:** The reward category to which the rule is applied;
- **Type:** The type of reward to which the rule is applied;
- **Notification:** The notification that pops up when the rule is fulfilled.

The remaining attributes depend on each class, being those divided in the following way:

- **Issue class**;
  - Amount of Rewards: The amount of rewards the user needs to win to fulfill the rule.
- **Meeting class**;
  - Action: The issue-related action to which the rule is applied to;
  - Amount of Issues: The amount of issues to which the user needs to perform the previous action in order to fulfill the rule.
- **Reward class**.
  - Amount of Meetings: The amount of meetings the user needs to attend to fulfill the rule;
  - In a Row: If checked, the rule is applied to consecutive meetings. Otherwise, the rule is applied the amount of meetings regardless of them being consecutive or not;
  - Multiplier: The value to which the number of XP is multiplied for every consecutive meeting the user attends to;

The creation of **rewards** is managed in a similar way, being new rewards described by the following attributes:

- **Type:** The type of reward (if it is a badge or gem);
- **Class:** The element to which the reward is applied;
- **Roles:** The roles of the project to which the reward is applied;
- **Projects:** The projects to which the reward is applied;
- **Condition:** The condition the user needs to fulfill in order to win the reward;

Table 6.2: Comparison between app features and the identified challenges.

Feature	Challenges
Dashboard	Poor communication and comprehension of the practitioner's own contribution to the project; Lack of Scrum knowledge (practitioners should be familiar with the burndown chart); Struggle to track the project's progress.
Team	Poor communication and cooperation.
Rewards	Resistance to adopt Scrum methodology; Motivation to use agile methods; Motivation to create user stories, sprint backlogs, prioritising tasks and estimating effort; Implementing personal rewarding.
App Configuration	Tailor gamification to specific contexts so it can be effective.
Notifications	Struggle to create and estimate user stories; Poor knowledge sharing since issues are not well detailed; Progress tracking; Motivation to use agile methods; Motivation to create user stories, sprint backlogs, prioritising tasks and estimating effort.

- **Extra XP points:** The amount of XP the user gets when winning the reward;
- **Repeatable:** If yes, the reward can be won more than once inside the chosen context (e.g. a repeatable reward in a sprint context can be awarded more than once inside one sprint);
- **Redeem Code:** When the code is given to a user, (s)he can use it to redeem that reward;
- **Description:** The reward description;
- **Notification:** The notification that pops up when the reward is won.

The difference between badges, and gems, regarding attributes is the fact that the former contains a name, and a category. Gems will be named after their value e.g., a gem reward worth 5 gems will be named "5 Gems". Each reward's image can be customised from a default set, and different background colours can be chosen. Default items can only be enabled or disabled, but items created by the administrator can also be edited or removed.

6.4 Discussion

Having our app features defined and drawn, there should be a connection between them and the challenges identified before. For that purpose, Table 6.2 presents those connections for an easy comprehension of the choices behind the defined features.

7

Demonstration

Contents

This chapter presents the planned activities to demonstrate the research proposal. Demonstration activities include results' analysis, and an interview with a Scrum Master.

7.1 Business Context

To demonstrate the app, we collaborated with a Portuguese IT company that uses Jira Software to support their development processes. The fact that Jira Software is already embedded in their projects, is very beneficial as their teams are familiar to it, and hence, understand what features are relevant for them. In addition, the intergration effort is expected to be low. Their experience could be also a valuable resource in what regards suggestions, and future modifications to enhance our app.

7.2 Activity Plan

This first iteration includes analysis, design, and development activities. Afterwards, demonstration activities take place to test and evaluate the app. The first demonstration activity is the extraction of historical data from the project where the app is supposed to be installed. Extracted data was based on the metrics defined in Section 7.3.2. Furthermore, the team assigned to the same project used the app to generate more data so we could compare against the previously collected. The objective is to compare "before", and "after" contexts to retrieve knowledge on how behaviours, and results have changed with the implementation of the app.

In addition, an interview with the Scrum Master was conducted to complement the results' analysis. The intention is to retrieve an intimate opinion from which we can understand what did work when using the app.

Configuration options presented in Section 6.3.6 are only available for demonstration, and evaluation, in a second iteration since its complexity and implementation effort are higher.

7.3 Historical Data Analysis

To evaluate our proposal, data from a previous project based on Scrum was used as a baseline measurement. The goal was to understand how the project was running and, after the app is installed, measure the variation between the former context and the current context.

Table 7.1: Metrics divided by sprint and assignee contexts

METRICS	
Sprint	**Assignee**
Total Issues (divided by type)	
Resolved/Not Resolved Issues	
Estimated/Not Estimated Issues	
Assigned/Unassigned Issues	Total Effort (in days)
Completed Sprints	Persistent Issues
Velocity	
Reopened Issues	

7.3.1 Project Considerations

The team working in ProjectX (anonymised name), a project managed with Scrum where a software product is being developed, was selected as the subject of this study. This project is organised in Jira Software by versions (i.e, releases) each containing multiple sprints with several issues. In ProjectX, estimation is performed using Original Time Estimate and, in contrast to what is advocated in Scrum, this team is not purely cross-functional as there are some practitioners fully committed with testing activities.

7.3.2 Data Source

ProjectX is a Scrum project containing more than 50 versions among which we picked the 9 most recent. These 9 releases include 27 sprints with a total of 306 issues that represent our sample. The choice behind this specific number of releases regards the fact that they represent the last two major releases. The oldest ones were discarded since they had multiple inconsistencies and represented a long-term performance that we did not considered realistic to tackle with the first demonstration phase.

As some metrics from table 5.1 could not be extracted, we grouped the available metrics in Table 7.1. These metrics fall into both sprint and assignee contexts, as presented in Section 5.2.2. As Jira Software does not support Scrum meetings, meeting-related metrics were not considered during this activity.

In what regards sprints, we intend to retrieve the total number of issues, and, from those, the number of the ones which are resolved so we can calculate the number of completed sprints, and the sprint velocity. In addition, we analyse assigned and unassigned issues, the estimated and not estimated issues, and the reopened issues. Assignment and estimation metrics support conclusions on sprint planning and work allocation. The number of reopened issues provides a perception of how many issues needed to be reworked throughout the project.

In the assignees' scope, the first three metrics are the same for sprint context, and two are specific to assignees: estimated effort, and persistent issues. The intent is not only to learn how the project

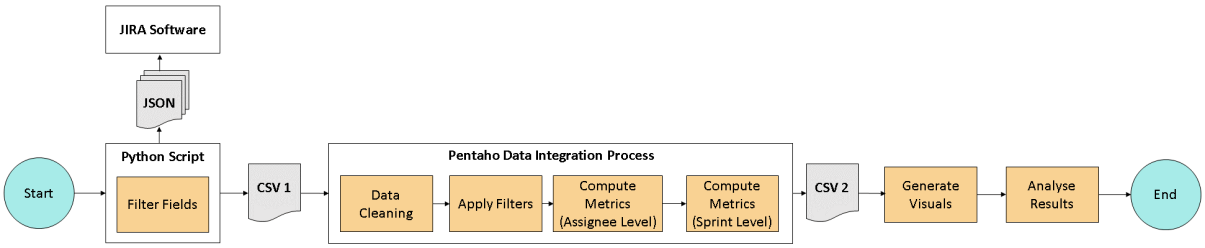


Figure 7.1: Process for extracting, processing and analysing research data.

flows from each sprint to another, but also comprehend the assignees' influence on them. For that reason, we seek to associate a profile and understand the influence of each assignee in the project. Added to these, the estimated effort is calculated, in days, by the sum of all Original Time Estimates for each assignee. The last metric provides a perception of how many issues, from each assignee, were transferred between sprints.

7.3.3 Extract and Transform Data

To extract data from a ProjectX's instance available in Jira Software, a Python script (see Appendix B) was written. It works by collecting all issues' information from their corresponding JSON file, and then filtering out the irrelevant fields. This process allows us to get each issue's lifecycle information and export it to a Comma-separated Values (CSV) file (CSV1).

Since Jira Software data is complex, and the fact that most target metrics are not reflected in this raw data, Pentaho Data Integration[1] was used to extract, clean, and process the collected data stored in CSV1. Two Pentaho processes were built by composing several steps (i.e., operations) to obtain a structured table with the required information.

Both processes starts by importing CSV1 and cleaning existent data (define the right data type for each field; resolve NULL values; and group issues by sprint). The next step is to **Filter Rows** and remove the unnecessary data, which is followed by **Select Fields** steps. The latter is used to choose the values to compute each metric, being all metrics later used by **Group By** and **Row Denormalize** steps to compute them in the sprint or assignee level. All metrics are finally merged with **Merge Join** steps to generate a table with all metrics grouped by sprint, or assignee, which is then exported as a CSV file (CSV2). Based on the treated data available in CSV2, graphs were generated to visually depict information.

The full process for extracting, processing, and analysing data is summarised in Figure 7.1. Regarding Pentaho processes, both can be consulted in Appendix C.

[1] Pentaho Data Integration - http://community.Pentaho.com/projects/data-integration/ (accessed 12/08/2017).

7.3.4 Retrieved Information

The extracted data related to both sprint, and assignee context, is structured in tables to provide an easier analysis. These tables are available at Appendix D. Moreover, there are limitations to be accounted for:

- Fourteen issues were dragged along different sprints even though they resolved several times without being reopened. Since this is a special case and we did not find further explanation to this, we removed them from this analysis;
- Sprint velocity was calculated with each issue's estimation, but this is not defined for all of them. Even though the calculated velocity does not represent the most accurate effort for a specific sprint, we considered it meaningful enough to be included in our analysis.

292 issues among 27 Sprints were analyzed, including 29 stories (9,9%), 74 tasks (25.3%), 152 bugs (52.1%), 34 improvements (11.6%), and 3 new features (1%). From the sample, 82 issues were persistent (28%), and 151 estimated (51.7%). All issues were resolved and assigned to some practitioner. Considering the sprint context, three sprints were complete (11.8%), and 24 sprints contain reopened issues (88.9%).

Regarding the assignee context, ”sdrg” is the practitioner with more issues, 110 (37.7%), while other practitioners are assigned to less than 100 issues each. Again, ”sdrg” is the assignee with more allocated effort (200 days), and with more persistent issues. The only assignee with all issues estimated is ”mfda” (100%), which is also the second assignee with less effort allocated (10 days). In addition, ”mjmb”, ”pmmr” and ”sdrg” have more than half of their issues as bugs (75%, 67.9% and 50.9%, respectively).

7.4 Interview with Scrum Master

After analyzing historical data from ProjectX, we conducted a semi-structured interview with the ProjectX's Scrum Master. Although there was a script to follow, the interviewee was allowed to explore other paths, and improvise throughout the interview. The interview was conducted through Skype with two researchers that had different responsibilities. During thirty minutes, one researcher conducted the conversation and the other took notes on the computer using the script as template for data collection. In the end, the notes were revised by both researchers and a final report was arranged.

The interview script is available at Appendix E.

8

Evaluation

Contents

This section comprises the analysis of results extracted from the demonstration phase, depicted with graphs for a better, and more visual comprehension. In addition to results' analysis, we conducted an interview with the Scrum Master in charge of the ProjectX. Conclusions on both results, and interview, is comprised in the last section of this chapter.

8.1 Analysis

All tables and graphs presented in this section were generated in a Jupyter Notebook[1] using Matplotlib[2], and Seaborn[3] packages. On average, each sprint contains 16.5 issues, being those specifically 7.3 bugs, 3.3 tasks, 2.4 stories, 1.9 improvements, and 0.1 new features. Each sprint has an average of 63% issues resolved, 59% estimated, and 23.5% reopened. Mean velocity is 32 days. Table 8.1 contains the chosen summary statistics for each sprint. Considering that all issues are assigned, the ratio of assigned issues have no statistical meaning when compared to the other variables.

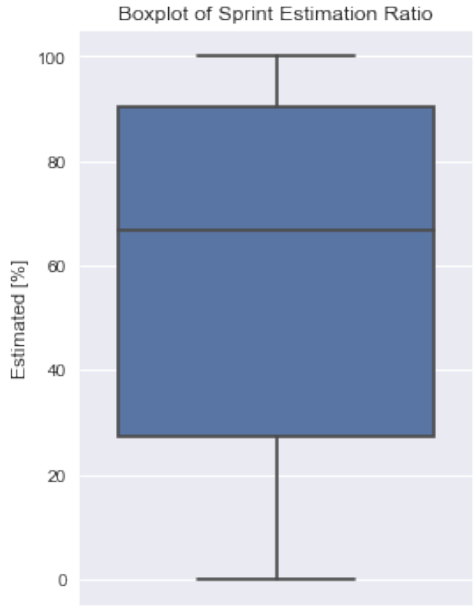
Table 8.1: Summary statistics for sprints, including issues of type story (S), task (T), bug (B), improvement (I), and new feature (NF).

	Issues						Estimated Issues		Resolved Issues		Reopened Issues		Velocity
	#	S	T	B	I	NF	#	%	#	%	#	%	(Days)
mean	16.5	2.4	3.3	7.3	1.9	0.1	10.4	59	11.3	63	3.7	23.5	32
std	10.9	3.3	2.8	4.4	1.9	0.4	9.1	33.6	9.1	29.1	3.8	15	38.9
min	2	0	0	0	0	0	0	0	0	0	0	0	0
25%	8.5	0	1	4	0.5	0	4.5	27.2	4.5	48.2	1	13.2	9
50%	13	1	3	7	1	0	8	66.7	9	66.7	3	23.1	17
75%	25	3	4.5	10	2.5	0	12	90.5	16.5	86.2	4.5	33.3	27.4
max	46	12	12	17	7	1	33	100	32	100	15	50	138.5

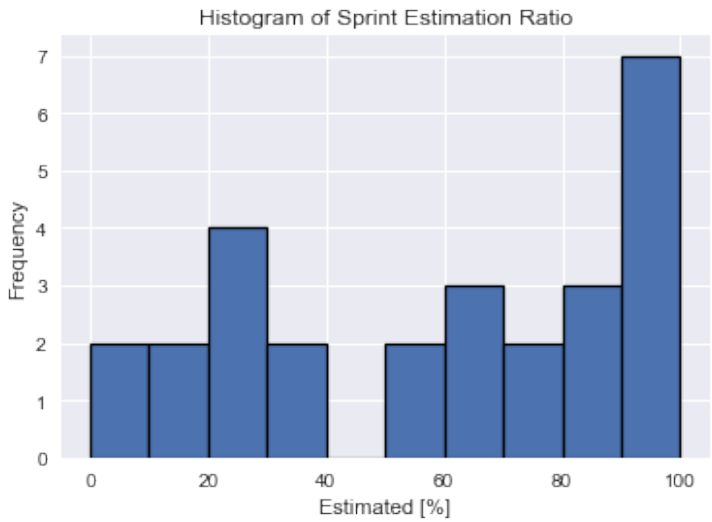
Regarding issue estimation, a boxplot, and a histogram were drawn to take some conclusions. We learned that half of the sprints have at most 66.7% of their issues estimated, as presented in Figure 8.1(a). Furthermore, in Figure 8.1(b) histogram we understand that 10 in 27 sprints (37%) do not have more than half of their issues estimated. Considering that different issue types represent distinct problems, we decided to further investigate if there was a correlation between the types and the existence of estimation. As only a residual number of issues were of type "New Feature", these were not considered for analysis.

Figure 8.2 is a heatmap representing the Pearson correlation coefficients for each case. The Pearson correlation coefficient is a value between -1 and 1, which represents a positive, negative, or null (i.e., without correlation) correlation with different strengths: very weak (0 - 0.19); weak (0.2 - 0.39), moderate

[1] Jupyter Notebook - http://jupyter.org/ (Accessed in 22/08/2017)
[2] Matplotlib - https://matplotlib.org/ (Accessed in 13/10/2017)
[3] Seaborn - https://seaborn.pydata.org/ (Accessed in 13/10/2017)



(a)



(b)

Figure 8.1: (a) Boxplot of estimation ratio for all sprints. (b) Histogram representing the sprint frequency for each bin with 10% interval.

(0.4 - 0.59); strong (0.6 - 0.79); or very strong (0.8 - 1).

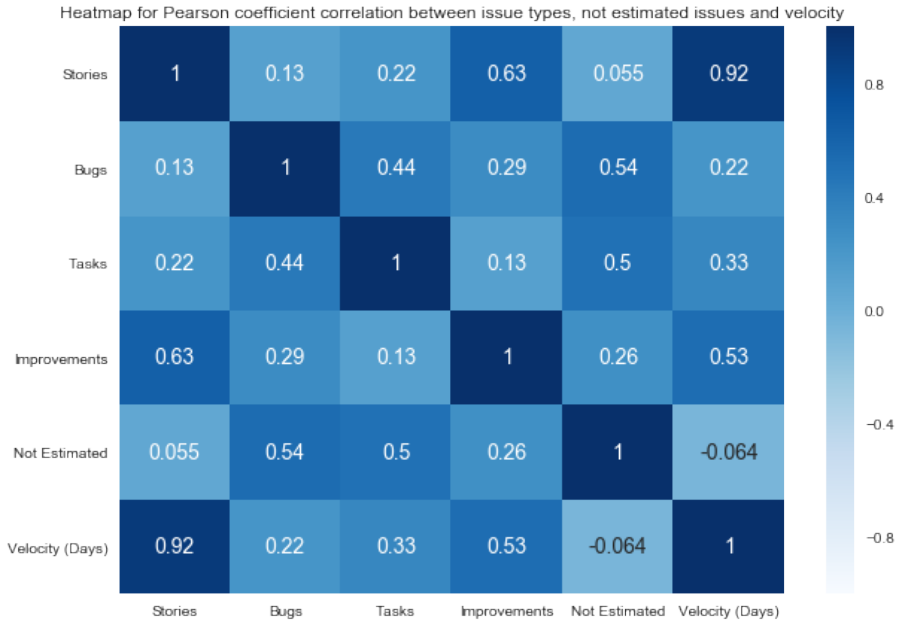


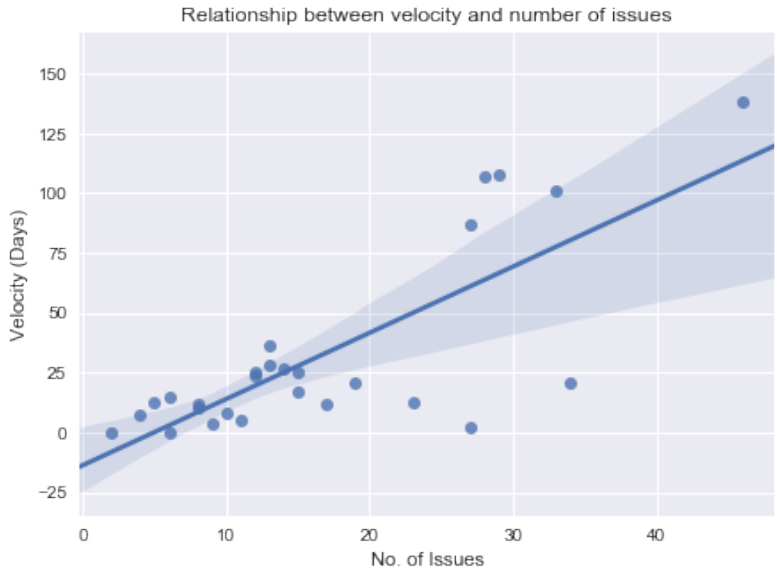
Figure 8.2: Heatmap representing the Pearson coefficient correlation between issue types with not estimated issues and velocity.

Bugs and tasks have a moderate positive correlation with not estimated issues (0.54 and 0.5, respectively) thus a high number of those might justify a low ratio of estimated issues. However, stories have a very strong positive correlation with velocity (0.92) which could mean that these are usually the ones that are mostly estimated and hence increase the overall velocity. This statement is confirmed by the very strong positive relationship between stories and estimated issues given by the correlation value of 0.85.

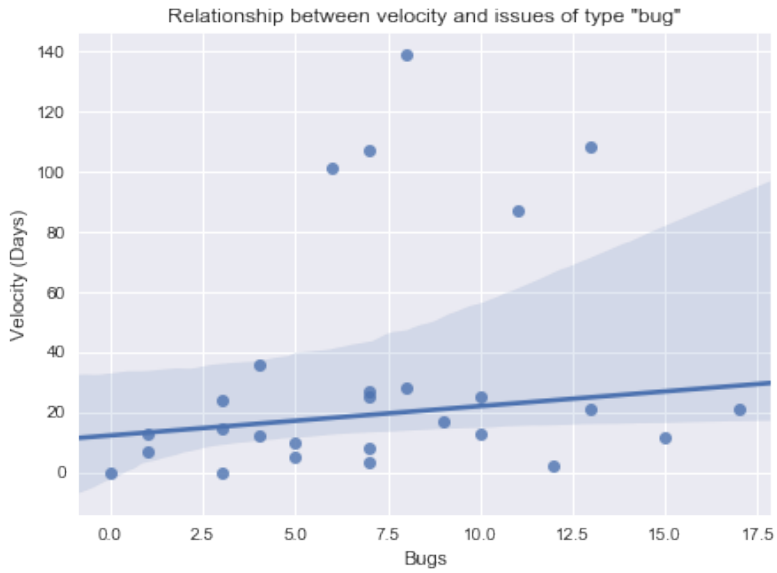
Since issues of type Bug are more frequent, and usually not estimated, we decided to study their relationship with velocity. Scatterplots of Figures 8.3(a) and 8.3(b) were used to show the relation of velocity with total number of issues, and velocity with number of bugs, respectively. It is possible to see that velocity increases with both the number of issues, and number of bugs, but the growth is slower for the latter. Also, we calculated the R^2 (or r-squared, a value between 0 and 1 that measures how close the data is fitted to the regression line) associated to both regressions so we could comprehend which metric has stronger influence on the variations.

The results were approximately 0.6 and 0.25, respectively, showing that 60% of the velocity variation is explained by the total number of issues while a lower 25% is explained by the number of bugs. Still, the fact that issues of type Bug are not always estimated can justify a slower velocity growth.

Following these results, we analyzed if some issue type was more likely to be reopened. From the Pearson correlation coefficients shown in Figure 8.4 we can see that there is a weak significant corre-



(a)



(b)

Figure 8.3: (a) Scatterplot comparing velocity with the number of issues. (b) Scatterplot comparing velocity with issue type ”bug”.

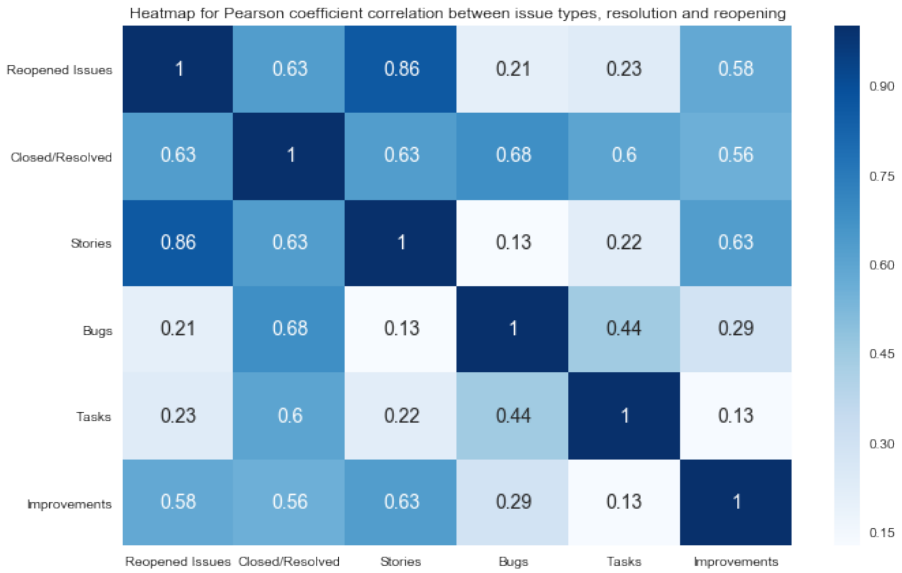


Figure 8.4: Heatmap representing the Pearson coefficient correlation values between issue types, reopened and resolved issues.

lation between bugs and reopened issues, yet, stories tend to be reopened. Regarding the number of resolved issues, all types show a strong positive correlation with this variable. To take clear conclusions, we performed the same calculation between the issue types and the ratios of reopened, and resolved issues.

Results shown that bugs are usually resolved and do not tend to be reopened, even though they are not always estimated. This is, again, shown by the moderate positive relationship between the latter and the ratio of resolved issues. On the opposite, stories seem to be the most likely to be reopened.

The challenge faced by Scrum teams, identified in Section 2, of having poor specification of requirements and little documentation, was further studied as we analyzed if the number of issues with a description could be related with the number of reopened issues. As the metric is represented by 1 if the issue has been reopened, and 0 otherwise, a special case of the Pearson correlation, the Point Biserial Correlation, also ranging from -1 to 1, had to be used instead. The result was 0.017, leading us to conclude that the existence of an issue description has, apparently, no impact on issue reopening.

This analysis have been focusing on the sprint context. However, the assignee context should be also considered to improve our conclusions. Each assignee has, on average, 43.6 issues assigned, from which 29.9 are bugs, 15.4 are tasks, 10.1 are stories, 8 are improvements, and 0.6 are new features. Also, each one has an average of 40.3% of their issues estimated, is allocated with 62.4 days of effort, and have 20.4 persistent issues. This statistics can be found in Table 8.2.

There is a strong positive correlation (0.73) between the effort allocated for an assignee, and the number of his/her not estimated issues. Additionally, four out of the seven assignees have at least 50%

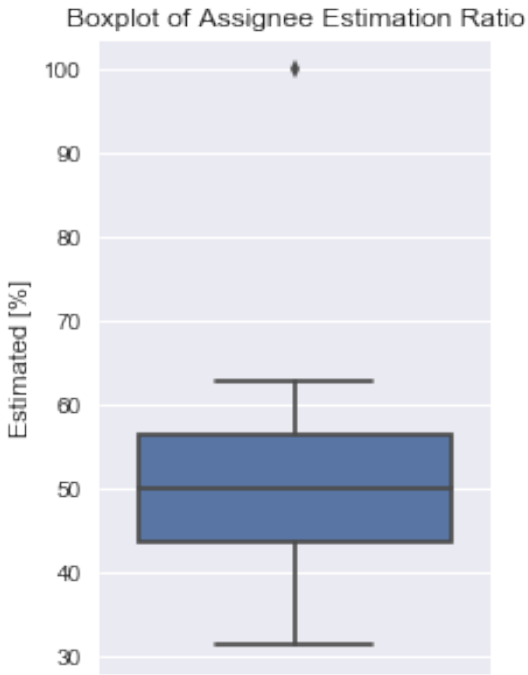


Figure 8.5: Boxplot of estimation ratio for all assignees.

of their issues estimated (see Figure 8.5 and Table C.2) representing 131 issues (44.9%), and 239,2 days of estimated effort (64%) of the full sample.

Since the allocated effort has, apparently, a negative influence on issue estimation, we also studied its influence on persistent issues. The Pearson correlation coefficient between the first, and the number of persistent issues shows a very strong correlation of 0.89. We argue that an incorrect amount of allocated tasks could lead practitioners to prioritise their work to the extent that the will leave some tasks undone.

Table 8.2: Summary statistics for assignees, including issues of type story (S), task (T), bug (B), improvement (I), and new feature (NF).

	Issues						Estimated Issues		Persistent Issues		Total Effort (Days)
	Total	S	T	B	I	NF	#	%	#	%	
mean	41.7	4.1	10.6	21.7	4.9	0.4	21.6	54.4	20.7	51.6	53.4
std	42.8	6.3	11.2	24.3	5.2	0.5	25	22.5	22.1	31.6	68.6
min	5	0	1	0	0	0	3	31.2	1	18.8	6.2
25%	8	1	1	3	1.5	0	5	43.4	3.5	28.8	12.6
50%	16	2	5	12	3	0	5	50	11	49.1	23
75%	72.5	3.5	19	39	6.5	1	32	56.4	36	63	59.8
max	110	18	28	56	15	1	69	100	54	110	200

8.1.1 Highlights

This analysis had into consideration multiple variables from 27 sprints. Having all of them assigned to some practitioner narrowed some of our conclusions as we cannot measure differences with constant values. Still, we took the number of issues, their types, estimation, velocity, and the number of reopened and persistent issues, to analyse them through two perspectives.

Estimating issues is a very important task as it influences the others. Without knowing the complexity of the tasks allocated to a user, it is harder to plan and to fulfill deadlines. Also, it would contribute for a better sprint planning and a better distribution of work. In this project, bugs and tasks show a strong influence on estimation but we would need more data to infer the reasons behind this. The fact that bugs are unexpected and not planned might be the reason for them to kept without estimation.

Results have shown that the ProjectX's team struggled to estimate tasks, and plan sprints, being both problems also identified in the literature. Results from the survey also show that practitioners are not motivated to perform these activities. A possible cause of delayed work identified by us consisted on the lack of efficient descriptions for new issues i.e., people working on those issues would not understand clearly what to do. However, we did not verify any clear impact of descriptions on delayed work.

Regarding velocity and effort metrics, the fact that they are calculated with issues' estimation means that both metrics will be affected by the rate of estimated issues. Contrary to bugs and tasks, stories were often estimated. We hypothesised that testers reported bugs, and developers estimated their effort even though they might not clearly understand the complexity associated to them.

Most sprints presented reopened issues. Stories and tasks were often reopened while bugs were always resolved in the first time. We have also seen that these practitioners are more likely to work on the same issue through multiple sprints when they have more effort allocated to them. This could mean that an overload of work might force them to postpone other tasks, and consequently, delay important work.

Finally, we found that three assignees (pmmr, rmbr, and sdrg) have significantly more work than the others. We hypothesise that these assignees might play a different role in the project even though we have no data to claim specific roles.

8.2 Interview with Scrum Master

To complement the previous analysis, and clarify some of the retrieved results, we conducted an interview with the Scrum Master involved in the demonstration phase of this research. We prepared a semi-structured interview, so the interviewee would feel free to provide suggestions, and go deeper on some facts that he considered relevant.

The interviewee currently holds a Msc degree in Informatics Engineering, but no Scrum certifications.

He has been working in the company for six years, from which four have been dedicated to the project in which our app was installed. Regarding Jira, he was not given any course so most of the learning was self-paced. Currently, our interviewee used Jira on a daily-basis, and spends around 30% of his day using it for management (plan releases, handle existent issues, etc.), and service desk tasks.

He has been using an adapted Scrum methodology since he started in his current company. Regarding his team, he evaluated Scrum compliance to be somewhere between 80 to 85%, and he considers them motivated to follow the required practices. Daily meetings, as advocated by Scrum, are always the first activity in the morning so any constraints can be rapidly solved before remaining work is handled.

Our interviewee has already used other issue trackers but considers Jira as one of the best as it is good-looking, very customizable, and allows an efficient management of both planning, and development phases. However, he pointed one downside: the inability to integrate Jira with the companies resource management so he could see their current distribution, and understand how to arrange them to plan further sprints. At the moment, he uses a spreadsheet to conduct this management.

The interviewee was familiar with the gamification concept as the company has used BadgR[4], a Bitbucket app to gamify repositories. He mentioned that the badges feature in our app was similar in BadgR, which he enjoyed using.

An inconsistency was found regarding the company's workflow. This workflow states that developers send their issues for testing after they are worked on, and then testers resolve them in the system, after testing. Because of this, only testers could gain XP.

He considered important to have badges broke down by availability, and closeness to be achieved. One downside was the number of given notifications since he would not read all of them. A suggestion was to comprise more information (like a report) in a single pop-up so users would not need to shift attention between different elements on the screen.

The interviewee agreed that Scrum roles, and meetings, were relevant but they should be customisable. He suggested the existence of a mechanism to assess a user's skills, and to calculate his/her issues' estimation, so they could better plan their work. Also, having each users checking-in to meetings, and someone creating them, was considered extra work to be done. A suggestion for this was having the Scrum Master (or the person in charge of the meeting) assigning the attendants to a list.

In general, positive feedback included visuals, user experience, and notifications (even though they should show up in a lower amount). However, the fact that XP were only given to the ones literally resolving the issues, was unfair to developers, as explained before. An additional suggestion was to include a leaderboard as they like the intra-team competition, and it keeps them motivated to do more rather than afraid to fail (as shown in chapter 4). Overall, using our app was considered an experience positive, and our interviewee claimed he would use it everyday as long as challenges keep on coming.

[4]BadgR - https://marketplace.atlassian.com/plugins/nl.stefankohler.stash.badgr/server/overview (Accessed in 06/10/2017

9

Conclusion

Contents

Keeping practitioners motivated to comply with their work, and to adapt the necessary Scrum practices have been revealed as difficult challenges. Solutions have been proposed with several of them using gamification, and representing promising approaches to tackle this problem, however, none of them were tested with Scrum teams in real projects.

Many instances of software development processes using Scrum keep failing for different reasons, and Jira Software does not promote the Scrum best practices important to improve the process efficiency.

This research proposes a gamification solution strongly based on the reviewed literature concerning practitioners' motivation, software metrics, and gamification. This solution was implemented as a Jira App with the goal of increasing practitioners' motivation to adopt Scrum practices. It intends to use game elements in a strategically way, including immediate feedback to engage users in specific, and important moments of their work.

Data from a Scrum project was extracted, and analysed, to achieve a baseline. This research's proposal was then implemented in the same project and it is currently being used by a Scrum team.

9.1 Lessons Learned

The single adoption of agile practices is not enough to mitigate all problems related to software development processes. Although agile methodologies are more engaging, there is still resistance in what regards adoption of agile practices.

We have found that the main problems practitioners struggle with, are maintaining proper documentation, planning sprints, and estimating tasks' effort. The fact that this work can be perceived as excessively time consuming, may lead practitioners to feel less motivated to comply with such practices.

The lack of good documentation may cause confusion among practitioners as they may not understand the project's requirements, and what they must accomplish. In addition, failure to understand each tasks' complexity could result in delayed work and incomplete tasks that will later be reopened for corrections.

Regarding delayed work, we found this is not only result of bad documentation but also of unbalanced work assignment. The fact that some assignees had a higher amount of associated effort (the sum of issues' estimation) had impact on their ability to resolve their issues before the end of the sprint as Scrum advocates.

Gamification is an efficient strategy to mitigate these problems, but a wrong use of it can harvest undesired results. The implementation context of a gamification system must be clear, and the users' needs must be taken into account to better adapt the game elements. Additionally, competition between practitioners must be carefully addressed as it may become too extreme and affect teams' performance.

9.2 Communication

As stated in DSRM, our research must be exposed to the scientific community as results, and lessons learned might have relevance to others. Throughout this document we have shown the problem, a strategy to tackle it, and a solution to solve it. This DSRM step exists to communicate the conducted research, and expose its utility, and effectiveness. In addition to this, we must clarify the problem's importance to the target audience, and attempt to evaluate the proposed solution.

Under the scope of this research, we produced four papers with the goal of exposing results to the scientific community. Two of them were accepted, and the others have been submitted, and are still pending acceptance. The list of papers is available in Table 9.1.

Table 9.1: List of produced papers with corresponding conferences.

COMMUNICATION		
Conference	**Article**	**Status**
ECIS 2017 (CORE Rank A)	A Survey of Failures in the Software Development Process [33]	Accepted
VS-Games 2017	Gamifying Software Development Scrum Projects [52]	Accepted
ICSE 2018 (CORE Rank A*)	Challenges for Implementing Scrum	Submitted
	A Literature Review of Motivators for Software Engineering	Submitted

9.3 Research Limitations

JIRA Software presents restrictions in what can be changed/added regarding new elements, and mechanisms. The platform does not provide detailed documentation on development tasks and that raised the difficulty level of developing the solution. Some metrics, such as the amount of completed sprints, were also not possible to be extracted considering that Jira Software does not record the planned date, and the real date of completion.

Since the app was developed with a company, communication problems were present during the process. There problems caused some delays and the project was prolonged.

9.4 Future Work

A Scrum team is currently testing the proposal hence, results analysis, after a period of no less than a month, will be required to evaluate the impact of gamification in the project. This analysis can be useful to understand which are the game elements to keep or not, and how are these motivating the team. However, this evaluation has not been performed before because of Jira implementation problems, and issues related to the demonstration company.

Based on results and practitioners feedback, the app should be improved and new iterations should be run with the same team. In addition, testing the app with other Scrum teams (inside or outside the company we worked with) would be very beneficial considering that different teams, and projects, may have different necessities.

Bibliography

[1] K. Peffers, T. Tuunanen, M. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *J. Manage. Inf. Syst.*, vol. 24, no. January, pp. 45–77, 2008.

[2] M. Bhardwaj and A. Rana, "Key Software Metrics and its Impact on each other for Software Development Projects," vol. 6, no. 1, pp. 242–248, 2016.

[3] K. Werbach and D. Hunter, *For the Win: How Game Thinking Can Revolutionize Your Business*. Wharton Digital Press, 2012.

[4] B. Fogg, "A behavior model for persuasive design," *Proceedings of the 4th International Conference on Persuasive Technology - Persuasive '09*, pp. 40:1–40:7, 2009. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1541948.1541999

[5] S. Overhage, S. Schlauderer, D. Birkmeier, and J. Miller, "What Makes IT personnel adopt scrum? A framework of drivers and inhibitors to developer acceptance," *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2011.

[6] A. Marchenko and P. Abrahamsson, "Scrum in a multiproject environment: An ethnographically-inspired case study on the adoption challenges," in *Agile 2008 Conference*. IEEE, 2008, pp. 15–26.

[7] "11th annual State of Agile Report," VersionOne, Tech. Rep., 2017.

[8] K. Schwaber and M. Beedle, *Agile software development with Scrum*. Prentice Hall Upper Saddle River, 2002.

[9] K. Conboy, S. Coyle, X. Wang, and M. Pikkarainen, "People over process: Key challenges in agile development," *IEEE Software*, vol. 28, no. 4, pp. 48–57, 2010.

[10] a. Jain and S. Angadi, "Gamifying Software Development Process," *Gamification: Rediscover the Power of Engagement*, vol. 11, no. 3, pp. 21–29, 2013. [Online]. Available: http://www.infosys.com/infosys-labs/publications/Documents/gamification/gamifying-software-development-process.pdf

[11] A. H. Mohammad, T. Alwada'n, and J. M. Ababneh, "Agile Software Methodologies : Strength and Weakness," *International Journal of Engineering Science and Technology (IJEST)*, vol. 5, no. 03, pp. 455–459, 2013.

[12] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *MIS Quaterly*, vol. 28, no. 1, pp. 75–105, 2004.

[13] T. DeMarco and T. Lister, *Peopleware: productive projects and teams*. Addison-Wesley, 2013.

[14] C. K. Riemenschneider, B. C. Hardgrave, and F. D. Davis, "Explaining software developer acceptance of methodologies: A comparison of five theoretical models," *IEEE Transactions on Software Engineering*, vol. 28, no. 12, pp. 1135–1145, 2002.

[15] M. L. Sánchez-Gordón and R. V. O'Connor, "Understanding the gap between software process practices and actual practice in very small companies," *Software Quality Journal*, vol. 24, no. 3, pp. 549–570, 2016.

[16] K. Kautz, B. Hansen, and D. Jacobsen, "The Utilization of Information Systems Development Methodologies in Practice," *Journal of Information Technology Case and Application Research*, vol. 6, no. 4, pp. 1–20, 2004.

[17] R. Kaur and J. Sengupta, "Software Process Models and Analysis on Failure of Software Development Projects," *International Journal of Scientific & Engineering Research*, vol. 2, no. 2, pp. 1–4, 2011.

[18] L. McLeod and S. G. MacDonell, "Factors that affect software systems development project outcomes," *ACM Computing Surveys*, vol. 43, no. 4, pp. 1–56, 2011.

[19] A. Mandal and S. C. Pal, "Identifying the reasons for software project failure and some of their proposed remedial through BRIDGE process models," *International Journal of Computer Sciences and Engineering*, vol. 3, no. 1, pp. 118–126, 2015.

[20] H. Hajjdiab, A. S. Taleb, and J. Ali, "An industrial case study for Scrum adoption," *Journal of Software*, vol. 7, no. 1, pp. 237–242, 2012.

[21] C. J. Stettina and W. Heijstek, "Necessary and Neglected? An Empirical Study of Internal Documentation in Agile Software Development Teams," in *Proceedings of the 29th ACM International Conference on Design of Communication (SIGDOC 2011)*. ACM, 2011, pp. 159–166.

[22] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in Human Behavior*, vol. 51, pp. 915–929, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.chb.2014.10.046

[23] J. Cho, “Issues and Challenges of Agile Software Development with Scrum,” *Issues in Information Systems*, vol. 9, no. 2, pp. 188–195, 2008.

[24] C. R. Prause, J. Nonnen, and M. Vinkovits, “A Field Experiment on Gamification of Code Quality in Agile Development,” *Psychology of Programming Interest Group (PPIG) Annual Conference*, vol. 64, pp. 175 – 186, 2012.

[25] D. Turk, R. France, and B. Rumpe, “Limitations of agile software processes,” *Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP 2002)*, p. 4, 2002.

[26] K. Dikert, M. Paasivaara, and C. Lassenius, “Challenges and success factors for large-scale agile transformations: A systematic literature review,” *Journal of Systems and Software*, vol. 119, pp. 87–108, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2016.06.013

[27] R. Hoda and L. K. Murugesan, “Multi-level agile project management challenges: A self-organizing team perspective,” *Journal of Systems and Software*, vol. 117, pp. 245–257, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2016.02.049

[28] N. Sekitoleko, F. Evbota, E. Knauss, A. Sandberg, M. Chaudron, and H. H. Olsson, “Technical Dependency in Large-Scale Agile Software Development,” in *International Conference on Agile Software Development*. Springer, 2014, pp. 46–61.

[29] T. Dybå and T. Dingsøyr, “Empirical studies of agile software development: A systematic review,” *Information and Software Technology*, vol. 50, no. 9-10, pp. 833–859, 2008.

[30] E. Herranz, R. Colomo-Palacios, A. Seco, and M. Yilmaz, “Gamification as a Disruptive Factor in Software Process Improvement Initiatives,” *Journal of Universal Computer Science*, vol. 20, no. JUNE 20141. Herranz, E., Colomo-Palacios, R., Seco, A. & Yilmaz, M. Gamification as a Disruptive Factor in Software Process Improvement Initiatives. 20, 885–906 (2014)., pp. 885–906, 2014.

[31] Y. I. Alzoubi and A. Q. Gill, “Agile Global Software Development Communication Challenges: A Systematic Review,” in *Proceedings of Pacific Asia Conference on Information Systems*. AIS, 2014, pp. 1–13.

[32] E. Hossain, M. A. M. Babar, and H.-y. Paik, “Using Scrum in Global Software Development: A Systematic Literature Review,” *Fourth IEEE International Conference on Global Software Engineering, 2009. ICGSE 2009.*, pp. 175–184, 2009. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5196931

[33] R. Marques, G. Costa, M. Silva, and P. Gonçalves, “A Survey on Failures in the Software Development Process,” in *European Conference on Information Systems*, Guimarães, Portugal, 2017.

[34] J. A. Highsmith, “What Is Agile Software Development?” *The Journal of Defense Software Engineering*, vol. 15, no. 10, pp. 4–9, 2002.

[35] K. Schwaber and J. Sutherland, “The Scrum Guide,” Scrum.Org and ScrumInc, Tech. Rep., 2016.

[36] R. Leblanc, A. Dingle, J. D. Hagar, and J. Knight, *Software Metrics Fundamentals of Dependable Computing for Software Engineers*. CRC Press, 2015.

[37] E. Kupiainen, M. V. Mäntylä, and J. Itkonen, “Using metrics in Agile and Lean software development - A systematic literature review of industrial studies,” *Information and Software Technology*, vol. 62, no. 1, pp. 143–163, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2015.02.005

[38] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, “From game design elements to gamefulness,” *Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments - MindTrek ’11*, pp. 9–11, 2011.

[39] S. Deterding, D. Dixon, R. Khaled, L. Nacke, M. Sicart, and K. O’Hara, “Gamification: Using Game Design Elements in Non-Game Contexts,” *Proceedings of the 2011 Annual Conference Extended Abstracts on Human Factors in Computing Systems (CHI 2011)*, pp. 2425–2428, 2011. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1979742.1979575

[40] S. . P. Singh, “Gamification: A Strategic Tool for Organizational Effectiveness,” *International Journal of Management*, vol. 53, no. 9, p. 108, 2013.

[41] S. Mcleod and L. Albert, “Behaviorist Approach The History of Behaviorism,” no. 1897, pp. 1–4, 2007.

[42] S. A. McLeod, “Skinner - Operant conditioning,” *Simply Psychology*, vol. 1, no. 1, p. 2, 2007. [Online]. Available: http://www.simplypsychology.org/operant-conditioning.html

[43] F. Groh, “Gamification: State of the Art Definition and Utilization,” *Research Trends in Media Informatics*, pp. 39–46, 2012.

[44] J. Hamari, J. Koivisto, and H. Sarsa, “Does gamification work? - A literature review of empirical studies on gamification,” *Proceedings of the Annual Hawaii International Conference on System Sciences*, pp. 3025–3034, 2014.

[45] A. Domínguez, J. Saenz-De-Navarrete, L. De-Marcos, L. Fernández-Sanz, C. Pagés, and J. J. Martínez-Herráiz, “Gamifying learning experiences: Practical implications and outcomes,” *Computers and Education*, vol. 63, pp. 380–392, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.compedu.2012.12.020

[46] J. McGonigal, “Reality is Broken: Why Games Make Us Better and How They Can Change the World,” *New York*, p. 402, 2011.

[47] D. J. Dubois and G. Tamburrelli, “Understanding gamification mechanisms for software development,” *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, no. August 2013, p. 659, 2013.

[48] W. Snipes, A. Nair, and E. Murphy-Hill, “Experiences gamifying developer adoption of practices and tools,” *36th International Conference on Software Engineering, ICSE Companion 2014 - Proceedings*, pp. 105–114, 2014. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84903639340&partnerID=40&md5=90e13c478393b48523660bd564de7f2f

[49] E. Herranz, R. Colomo-Palacios, and A. d. A. Seco, “Gamiware: A gamification platform for software process improvement,” *Communications in Computer and Information Science*, vol. 543, pp. 127–139, 2015.

[50] F. Q. B. Da Silva and A. C. C. França, “Towards understanding the underlying structure of motivational factors for software engineers to guide the definition of motivational programs,” *Journal of Systems and Software*, vol. 85, no. 2, pp. 216–226, 2012.

[51] A. C. C. França, F. Q. B. da Silva, A. D. L. C. Felix, and D. E. S. Carneiro, “Motivation in software engineering industrial practice: A cross-case analysis of two software organisations,” *Information and Software Technology*, vol. 56, no. 1, pp. 79–101, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2013.06.006

[52] R. Marques, G. Costa, M. Silva, and P. Gonçalves, “Gamifying Software Development Scrum Projects,” in 9th *International Conference on Virtual Worlds and Games for Serious Applications*. Athens, Greece: IEEE, 2017.

A

Gamification Components

For completing actions, users can win XP, gems, and badges. Points are defined using the sequence of Fibonacci in which every number after the first two is the sum of the two preceding ones. Both badges, and points, along with the required actions are listed in Table A.1, and Table A.2[1].

Badges are represented by Figures A.1, A.2, A.3, A.4, A.5, A.6, A.7, A.8, A.9, and A.10.

Levels are implemented to give the user a sense of progressing. Two types of levels, global and local, were defined. The former has a global scope i.e., a user can gain XP in multiple projects and all of them will be considered for his/her global level. The latter is specific for each project, making the user feel that (s)he can progress in each of them. Table A.3, and A.4 provide two lists of both types of levels.

[1] Calculations include the estimation factor (EF) which is the conversion of the used estimation type to XP as shown; round(arg) which rounds the argument inside the function; avg(arg[]) which retrieves the mean value of all values in the argument array; and combo[2x] which accumulates the value of points awarded in the previous meeting.

Table A.1: List of Badges including the milestones to achieve them.

REWARDS			
Name	**Category**	**Type**	**Metric**
That's a first!	**Got Issues?**	Badge/Static/Individual	First time all issues are resolved by the end of a sprint.
Piece of Cake!		Badge/Static/Cooperative	
Team Work		Gems/Static/Cooperative	Team resolved all issues in a sprint.
Clerk		Badge/Static/Individual	Resolve 10 issues.
Getting Better		Badge/Static/Individual	Resolve 25 issues.
Centenary		Badge/Static/Individual	Resolve 100 issues.
Legendary		Badge/Static/Individual	Resolve 200 issues.
Mad Skills!		Badge/Dynamic/Individual	User that completed more estimation units in the sprint.
Social Machine	**Social**	Badge/Dynamic/Individual	User with the most attended meetings in a sprint.
Unstoppable	**Chronos**	Badge/Dynamic/Individual	User with lowest velocity negative deviation.
Full Throttle		Gems/Static/Cooperative	Team keeps sprint velocity equal or above the planned velocity.

Table A.2: List of points, and required actions to gain them.

SCORE	
Event	**XP**
Unlock a reward - Cooperative	**2 (per user)**
Unlock a reward - Individual	**2**
Resolve an issue	**EF**
(Badge "Clerk") Close 10 issues	**3**
(Badge "Getting Better") Close 25 issues	**5**
(Badge "Centenary") Close 100 issues	**8**
(Badge "Legendary") Close 200 issues	**13**
Resolve all issues before the end of the sprint	**Round(3* avg(EF))**
(Gems) Close a full sprint (The whole team resolves all issues)	**Round(5 * avg(EF))**
Attend a meeting	**2**
Attend next consecutive meeting	**2 + Combo[2x]**
Attend all meetings in a sprint	**13**

Figure A.1: ”That’s a First” badge.

Figure A.2: ”Piece of Cake” badge.

Figure A.3: ”Mad Skills” badge.

Figure A.4: ”Social Machine” badge.

Figure A.5: ”Unstoppable” badge.

Figure A.6: ”Work Hard Play Hard” badge.

Figure A.7: ”Full Sprint” badge.

Figure A.8: ”Team Work” badge.

Figure A.9: ”Centenary” badge.

Figure A.10: ”Legendary” badge.

Table A.3: List of local levels with the required XP to achieve them.

LOCAL LEVELS	
Level Name	**XP Interval**
Newbie	0 - 25
Apprendice	26 - 50
Resolver	51 - 100
Warrior	101 - 200
Hero	201 - 400
Pro	401 - 800
Boss	801 - 1600
Master	1601 - 3200
Veteran	3201 - 6400

Table A.4: List of global levels with the required XP to achieve them.

GLOBAL LEVELS	
Level Name	**XP Interval**
Newborn	0 - 100
Beginner	101 - 200
Climber	201 - 400
Fighter	401 - 800
1st Officer	801 - 1600
Captain	1601 - 3200
Big Chief	3201 - 6400
Godfather	6401 - 12800
God	12801 - 25600

B

Data Extraction Script

To extract data from Jira Software, a python script was written to access JSON files, and retrieve the required data. The script is shown below.

```
#!/usr/bin/env
import requests
import pandas as pd
import numpy as np
from requests.auth import HTTPBasicAuth
import json
import getpass
from datetime import datetime

#general vars
sprints = [88, 45, 39, 34, 91, 77, 41, 81, 37, 48, 31, 35, 82, 83, 46, 43, 53,
    50, 42, 64, 70, 60, 79, 57, 52, 84]
```

```
12
13 #Issue keys from CSV
14 #keys = pd.read_csv('keys.txt')
15 #keys_array = np.asarray(keys)
16 line = ''
17
18 #Login info
19 uname = input('Username: ')
20 pwd = getpass.getpass()
21
22 #Array of lines and sprints
23 lines = []
24 sprintCollection = []
25
26 #Iterate each sprint call each json file and collect general information
27 for sprint in sprints:
28     #url with json
29     urlIssues = 'http://atlassiandemo.xpand-it.com/jira/rest/agile/1.0/sprint/' +
           str(sprint) + '/issue?expand=changelog'
30     urlSprints = 'http://atlassiandemo.xpand-it.com/jira/rest/agile/1.0/sprint/'
           + str(sprint)
31
32     #request login and store page in r
33     ri = requests.get(urlIssues, auth=HTTPBasicAuth(uname,pwd))
34     rs = requests.get(urlSprints, auth=HTTPBasicAuth(uname,pwd))
35
36     #read json and get issue variables
37     issueInfo = json.loads(ri.content)
38     issueTotal = issueInfo['total']
39
40     #read json and get sprint variables
41     sprintInfo = json.loads(rs.content)
42     sprintName = sprintInfo['name']
43     sprintStart = sprintInfo['startDate'].split('.')[0].replace('T',' ')
44     sprintEnd = sprintInfo['endDate'].split('.')[0].replace('T',' ')
45
46     sprintCollection.append([issueInfo,issueTotal,sprintInfo,sprintName,
           sprintStart,sprintEnd])
```

```
47
48 #Iterate each sprint from collection to get the specific information
49 for sprint in sprintCollection:
50     #read json and get issue variables
51     issueInfo = sprint[0]
52     issueTotal = sprint[1]
53
54     #read json and get sprint variables
55     sprintInfo = sprint[2]
56     sprintName = sprint[3]
57
58     for issue in range(0, issueTotal):
59         issueKey = issueInfo['issues'][issue]['key']
60         historyItems = issueInfo['issues'][issue]['changelog']['total']
61         closedSprints = issueInfo['issues'][issue]['fields']['closedSprints']
62         fixVersions = issueInfo['issues'][issue]['fields']['fixVersions']
63
64         #Issue Current Fields
65         #',' are replaced for (comma) to avoid confusion when reading the csv
66         try:
67             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Issue Type
                   ,-,-,' + str(issueInfo['issues'][issue]['fields']['issuetype']['
                   name']) + ',-')
68         except:
69             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Issue Type
                   ,-,-,None,-')
70
71         try:
72             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Version,-,-,
                   ' + str(pd.DataFrame(issueInfo['issues'][issue]['fields']['
                   versions']).index.size) + ',-')
73         except:
74             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Version,-,-,
                   None,-')
75
76         try:
77             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,assignee
                   ,-,-,' + str(issueInfo['issues'][issue]['fields']['assignee']['
```

```
            name']).replace(',','[comma]') + ',-')
        except:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,assignee
                ,-,-,None,-')

        try:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Component
                ,-,-,' + str(pd.DataFrame(issueInfo['issues'][issue]['fields']['
                components']).index.size) + ',-')
        except:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Component
                ,-,-,None,-')

        try:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Epic Link
                ,-,-,' + str(issueInfo['issues'][issue]['fields']['epic']['name'
                ]).replace(',','[comma]') + ',-')
        except:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Epic Link
                ,-,-,None,-')

        try:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,description
                ,-,-,' + str(issueInfo['issues'][issue]['fields']['description'])
                .replace(',','[comma]') + ',-')
        except:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,description
                ,-,-,None,-')

        try:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Environment
                ,-,-,' + str(issueInfo['issues'][issue]['fields']['environment'])
                .replace(',','[comma]') + ',-')
        except:
            lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Environment
                ,-,-,None,-')

        try:
```

```
102             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,labels,-,-,'
                     + str(pd.DataFrame(issueInfo['issues'][issue]['fields']['labels'
                    ]).index.size) + ',-')
103         except:
104             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,labels,-,-,
                    None,-')
105
106         try:
107             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,priority
                    ,-,-,' + str(issueInfo['issues'][issue]['fields']['priority']['
                    name']).replace(',','[comma]') + ',-')
108         except:
109             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,priority
                    ,-,-,None,-')
110
111         try:
112             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,security
                    ,-,-,' + str(issueInfo['issues'][issue]['fields']['security']['
                    name']).replace(',','[comma]') + ',-')
113         except:
114             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,security
                    ,-,-,None,-')
115
116         try:
117             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,summary,-,-,
                    ' + str(issueInfo['issues'][issue]['fields']['summary']).replace(
                    ',','[comma]') + ',-')
118         except:
119             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,summary,-,-,
                    None,-')
120
121         try:
122             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Original
                    Estimate,-,-,' + str(issueInfo['issues'][issue]['fields']['
                    timeoriginalestimate']).replace(',','[comma]') + ',-')
123         except:
124             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Original
                    Estimate,-,-,None,-')
```

```
125
126         try:
127             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Attachment
                ,-,-,' + str(pd.DataFrame(issueInfo['issues'][issue]['fields']['
                attachment']).index.size) + ',-')
128         except:
129             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Attachment
                ,-,-,None,-')
130
131         for fv in fixVersions:
132             try:
133                 lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Fix
                    Version,-,-,' + str(fv['name']).replace(',','[comma]') + ',-'
                    )
134             except:
135                 lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Fix
                    Version,-,-,None,-')
136
137         for s in closedSprints:
138             try:
139                 lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Sprint
                    ,-,-,' + str(s['name']).replace(',','[comma]') + ',-')
140             except:
141                 lines.append(str(sprintName) + ',' + str(issueKey) + ',-,Sprint
                    ,-,-,None,-')
142
143         try:
144             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,status,-,-,'
                 + str(issueInfo['issues'][issue]['fields']['status']['name']).
                replace(',','[comma]') + ',-')
145         except:
146             lines.append(str(sprintName) + ',' + str(issueKey) + ',-,status,-,-,
                None,-')
147
148         for hst in range(0, historyItems):
149             itemTotal = pd.DataFrame(issueInfo['issues'][issue]['changelog']['
                histories'][hst]['items']).index.size
150             if hst == historyItems-1:
```

```
151                 itemid = 'Last History'
152             else:
153                 itemid = issueInfo['issues'][issue]['changelog']['histories'][hst
                        ]['id']
154             for item in range(0, itemTotal):
155                 resolSprint = False
156                 histType = issueInfo['issues'][issue]['changelog']['histories'][
                        hst]['items'][item]['field']
157                 fromState = issueInfo['issues'][issue]['changelog']['histories'][
                        hst]['items'][item]['fromString']
158                 toState = issueInfo['issues'][issue]['changelog']['histories'][
                        hst]['items'][item]['toString']
159                 timeStamp = datetime.strptime(issueInfo['issues'][issue]['
                        changelog']['histories'][hst]['created'].split('.')[0].
                        replace('T',' '),'%Y-%m-%d %H:%M:%S')
160
161                 if histType == 'Sprint':
162                     for spr in toState.split(','):
163                         resolSprint = False
164                         sprt = spr.strip()
165                         for s in closedSprints:
166                             if set(s['name'].strip().split(' ')) == set(sprt.
                                    split(' ')):
167                                 resolSprint = True
168                         if resolSprint == True:
169                             lines.append(str(sprintName) + ',' + str(issueKey) +
                                    ',' + str(itemid) + ',' + str(histType) + ',' +
                                    str(fromState).replace(',','[comma]') + ',' + str
                                    (toState).replace(',','[comma]') + ',-,-')
170                         else:
171                             lines.append(str(sprintName) + ',' + str(issueKey) +
                                    ',' + str(itemid) + ',' + str(histType) + ',' +
                                    str(fromState).replace(',','[comma]') + ',' + str
                                    (toState).replace(',','[comma]') + ',-,' + str(
                                    sprt))
172                 else:
173                     lines.append(str(sprintName) + ',' + str(issueKey) + ',' +
                            str(itemid) + ',' + str(histType) + ',' + str(fromState).
```

```
                        replace(',','[comma]') + ',' + str(toState).replace(',','[comma]') + ',-,-')

                #check which sprint the issue was reopened at
                if histType == 'status' and toState == 'Reopened':
                    for spr in sprintCollection:
                        if timeStamp >= datetime.strptime(spr[4],'%Y-%m-%d %H:%M:%S') and timeStamp <= datetime.strptime(spr[5],'%Y-%m-%d %H:%M:%S'):
                            lines[len(lines) - 1] = str(lines[len(lines) - 1][:-1]) + str(spr[3])
                            break

#Write file with all info
histories = open('histories.csv',mode='w+')
histories.writelines('Sprint,Issue Key,Hist ID,Field Type,From,To,Last Value,Pe Sprint\n')
for item in lines:
    editedItem = item.replace('"','*')
    histories.writelines(editedItem.replace('\n',' | ') + '\n')
histories.close()
```

C

Analysis Processes

To analyze the data extracted from Jira, it needed to be cleaned and transformed. For that purpose, two Pentaho processes (one for each context) were used. Figure C.1 regards the sprint context and Figure C.2 regards the assignee context.

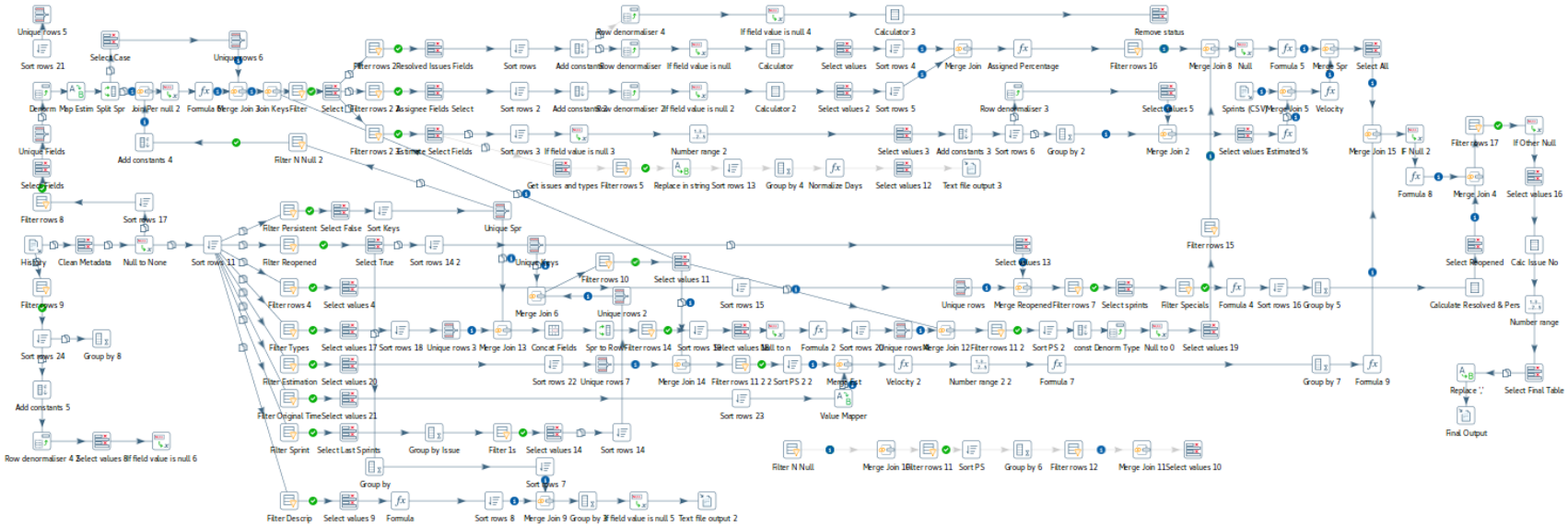


Figure C.1: Pentaho process for the sprint context.

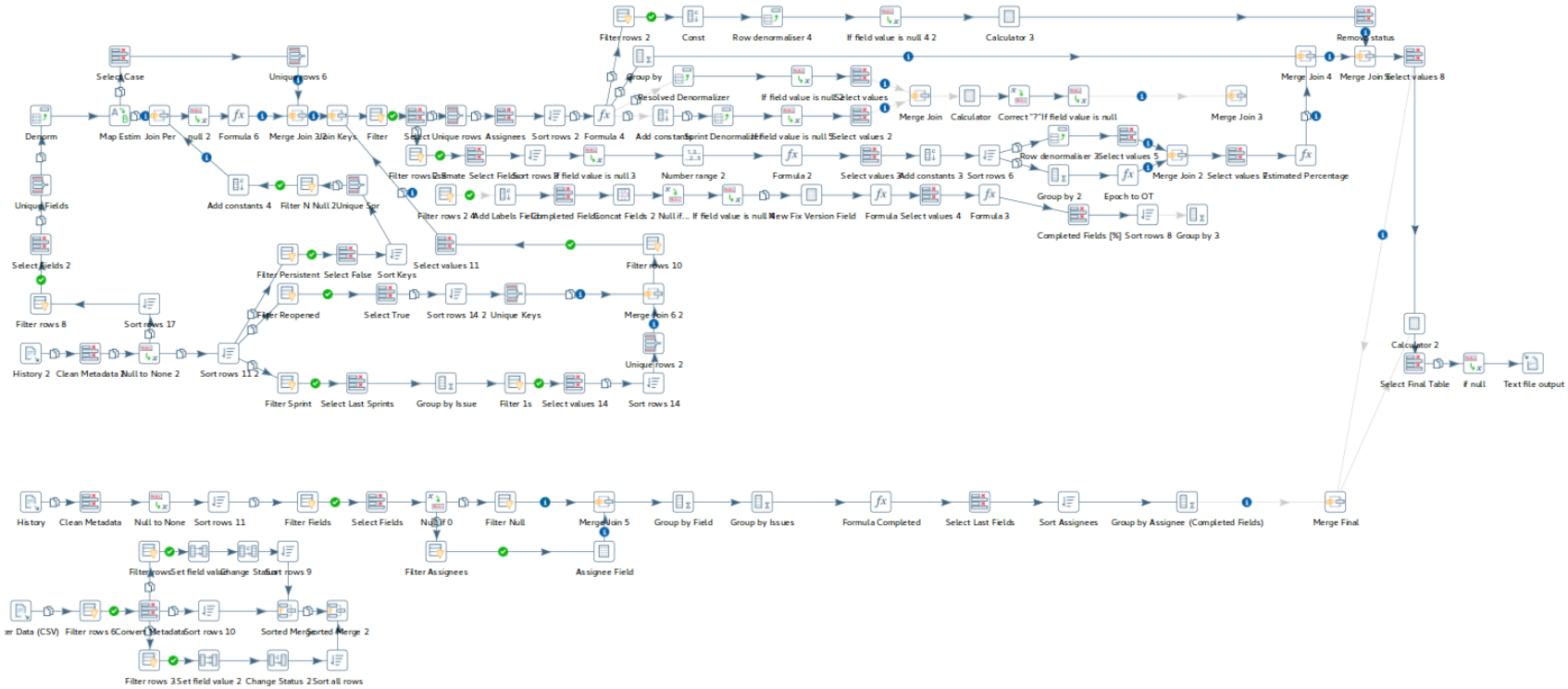


Figure C.2: Pentaho process for the assignee context.

D

ProjectX Historical Results

The extracted data is structured in a table that portrays each single Sprint in what regards their respective issues. Table D.1 contain the retrieved information for each sprint, structured in a way that allows further analysis to take significant conclusions.

The mutual metrics for both sprint and assignee contexts only change their scope, being them calculated, and grouped by each of them. However, to enhance the analysis on this context, the effort estimation metric (i.e., total effort measured in days) for each assignee was calculated. Table D.2 represents the extracted data on the assignee context.

This context provides knowledge on each team member, and allows us to take conclusions on how their amount of work impacts their results. Considering that each assignee presents different results for effort estimation, some with steeper values than others, we can hypothesize that some assignees have different roles than others. However, the existence of issues that are not estimated is a limitation of this measure because we cannot retrieve the effort associated to those.

Table D.1: Results from a real-world project regarding sprint context, including issues of type story (S), task (T), bug (B), improvement (I), and new feature (NF).

Sprint	Issues						Resolved Issues		Full Sprint?	Assigned Issues		Estimated Issues		Velocity (Days)	Reopened Issues
	#	S	T	B	I	NF	#	%		#	%	#	%		
R4.0.0 S1	12	0.0	3.0	3.0	5.0	1.0	1	8.33	N	12	100.0	12.0	100.0	23.75	3
R4.0.0 S2	14	2.0	1.0	7.0	3.0	1.0	7	50.0	N	14	100.0	14.0	100.0	26.75	2
R4.0.0 S3	6	2.0	1.0	3.0	0.0	0.0	0	0.0	N	6	100.0	6.0	100.0	14.75	3
R4.0.0 S4	5	2.0	1.0	1.0	1.0	0.0	2	40.0	N	5	100.0	4.0	80.0	12.5	1
R4.0.0 S5	23	2.0	9.0	10.0	2.0	0.0	20	86.96	N	23	100.0	7.0	30.43	12.75	3
R4.0.0 S6	27	0.0	12.0	12.0	2.0	1.0	24	88.89	N	27	100.0	5.0	18.52	2.125	0
R4.0.1 S1	9	0.0	1.0	7.0	1.0	0.0	9	100.0	Y	9	100.0	5.0	55.56	3.375	1
R4.0.2 S1	8	0.0	4.0	4.0	0.0	0.0	4	50.0	N	8	100.0	7.0	87.5	12.0	0
R4.0.2 S2	17	0.0	2.0	15.0	0.0	0.0	15	88.24	N	17	100.0	6.0	35.29	11.5	0
R4.0.3 S1	10	0.0	3.0	7.0	0.0	0.0	9	90.0	N	10	100.0	2.0	20.0	8.0	1
R4.1.0 S1	12	1.0	3.0	7.0	1.0	0.0	5	41.67	N	12	100.0	11.0	91.67	25.00	3
R4.1.0 S2	13	0.0	3.0	8.0	2.0	0.0	10	76.92	N	13	100.0	9.0	69.23	28.0	3
R4.1.1 S1	19	0.0	3.0	13.0	1.0	0.0	12	63.16	N	19	100.0	12.0	63.16	21.0	5
R4.1.1 S2	15	0.0	4.0	10.0	1.0	0.0	4	26.67	N	15	100.0	11.0	73.33	25.0	2
R4.1.1 S3	15	0.0	2.0	9.0	1.0	0.0	10	66.67	N	15	100.0	8.0	53.33	17.0	5
R4.1.2 S1	11	0.0	5.0	5.0	0.0	0.0	11	100.0	Y	11	100.0	2.0	18.18	5.0	2
R4.2.0 S1	6	1.0	0.0	3.0	2.0	0.0	6	100.0	Y	6	100.0	0.0	0.0	0.0	2
R4.2.0 S2	2	1.0	0.0	0.0	1.0	0.0	0	0.0	N	2	100.0	0.0	0.0	0.0	1
R4.3.0 S1	34	4.0	2.0	17.0	7.0	1.0	29	85.30	N	34	100.0	10.0	29.41	21.0	6
R4.3.0 S2	8	1.0	2.0	5.0	0.0	0.0	5	62.5	N	8	100.0	2.0	25.0	10.0	3
R4.3.0 S3	4	1.0	1.0	1.0	0.0	0.0	3	75.0	N	4	100.0	1.0	25.0	7.0	2
R5.0.0 S1	13	5.0	0.0	4.0	4.0	0.0	8	61.54	N	13	100.0	12.0	92.31	36.0	1
R5.0.0 S2	27	8.0	4.0	11.0	4.0	0.0	22	81.48	N	27	100.0	27.0	100.0	87.0	10
R5.0.0 S3	29	6.0	7.0	13.0	2.0	0.0	18	62.07	N	29	100.0	28.0	96.55	108.0	4
R5.0.0 S4	28	8.0	5.0	7.0	4.0	0.0	13	46.43	N	28	100.0	25.0	89.29	107.25	11
R5.0.0 S5	46	12.0	6.0	8.0	6.0	0.0	32	69.57	N	46	100.0	33.0	71.74	138.5	15
R5.0.0 S6	33	8.0	5.0	6.0	2.0	0.0	26	78.79	N	33	100.0	22.0	66.67	101.0	11

Table D.2: Results from a real-world project regarding assignee context.

Assignee	Issues						Estimated Issues		Persistent Issues		Total Effort (Days)
	Total	S	T	B	I	NF	#	%	#	%	
afro	10	3	5	0	2	0	5	50	4	40	23
fcma	6	1	1	3	1	0	3	50	2	33.3	6.2
mfda	5	1	1	3	0	0	5	100	1	20	10
mjbm	16	0	1	12	3	0	5	11	3	18.8	15.2
pmmr	81	2	18	55	5	1	40	49.4	28	34.6	68.4
rmbr	64	4	28	23	8	1	24	37.5	10	15.6	51.1
sdrg	110	18	20	56	15	1	69	62.7	34	30.9	200

E

Scrum Master Interview Script

A semi-structured interview was conducted with the ProjectX's Scrum Master. The interview script is presented below.

E.1 Interview Objectives

- Understand the main problems that lead to a low adoption of processes in the organisation;
- Comprehend the opinion of the interviewee regarding initiatives that use game elements to improve employees motivation;
- Identify the ways game elements can be helpful regarding the adoption of processes in the organisation.

E.2 Questions

E.2.1 Warm-up

- Describe briefly your main responsibilities/tasks in ProjectX;
- For how many years have you been in the company? And in ProjectX?
- Describe briefly your professional path (including the number of years as a practitioner;
- Describe briefly your academic background;
- Do you have any Scrum course/certification?

E.2.2 Jira

- Did you have any Jira course?
- How frequently do you use Jira (daily, weekly, etc.)?
- Which kind of activities do you perform with Jira?
- What are the main advantages and disadvantages of using Jira?
- Do you like working with Jira?

E.2.3 Scrum

- For how many years have you been using Scrum?
- How do you evaluate the team's importance and motivation to follow Scrum? And to comply with Scrum meetings?
- Tell us what are the main reasons that you think that explain the lack of motivation of practitioners to follow the Scrum methodology.

E.2.4 Gamification

- Have you heard of gamification before?
  - If yes, which apps did you know? Did you use any?
  - If no, what did you think gamification was?

E.2.5 The App

- On a daily basis, how long did you spend using the app?
- Regarding the ”Project Dashboard” section:
  - Do you think the presented information is relevant?
  - Do you think there is relevant information missing?
  - Do you consider that the project metrics are a good feedback?
  - What do you like most about this section? And least?
- Regarding the ”Profile Configurations” section:
  - Do you think the presented information is relevant?
  - Do you think there is relevant information missing?
  - Do you consider that the individual metrics are a good feedback?
  - What do you like most about this section? And least?
- Regarding the ”Team” section:
  - Do you think the presented information is relevant?
    - If yes, is there anything you find relevant to add?
    - If no, what could be different?
  - What do you like most about this section? And least?
- Regarding notifications:
  - Do you consider useful pop-up’s information?
  - Do you consider useful the tips used in issue creation forms?
  - What do you like most about this feature? And least?
- Regarding ”Rewards” section:
  - Do you consider that the default rewards are significant for the team?
  - What do you like most about this feature? And least?
  - Regarding the points system:
    - Do you think the score system is easy to understand?
    - Do you think the score system is fair?

    * What do you like most about this feature? And least?

The app implements two elements of Scrum that Jira does not support: roles and meetings.

- Do you think it is important for Jira to support both elements?
- In which way the definition of Scrum roles in Jira could be benefitial for projects?
- Do you consider the proceeding for meeting registration simple and understandable?
    * If yes, do you consider that your team would check-in with no problems?
    * If no, do you have any suggestions for improvement?

E.2.6 Final

- In general, what you think it is the best and worst things about the app?
- Do you think the app would motivate the teams to follow the Scrum methodology?
- Of all features, which do you consider as having higher positive impact on the team's motivation to follow Scrum? And with negative impact?
- Do you consider that the app's usage experience was positive?
- Would you use this app on a daily basis?
    - If yes, for how long?
    - If no, what would you change to improve its usage?

Thank you for your time and information shared with us. We hope this study's results can help improving the app and, consequently, your job.