

UNIVERSIDADE ABERTA

INSTITUTO SUPERIOR TÉCNICO



Improving Software Quality using Design Thinking with Scrum

Tatianna Arrais Rosal

Master in Information and Enterprise Systems

(master in association)

Dissertação orientada pelo:

Professor Doutor Miguel Mira da Silva

2019

UNIVERSIDADE ABERTA

INSTITUTO SUPERIOR TÉCNICO



Improving Software Quality using Design Thinking with Scrum

Tatianna Arrais Rosal

Master in Information and Enterprise Systems

(master in association)

Dissertação orientada pelo:

Professor Doutor Miguel Mira da Silva

2019

Resumo: O objetivo desta dissertação é associar o Design Thinking com o Scrum para a melhoria da qualidade do software. As metodologias associadas colaboram para aumentar a velocidade dos backlogs ao cliente-utilizador na fase inicial de um projeto de desenvolvimento de software. A metodologia de pesquisa é o estudo de caso por meio de entrevistas e inquéritos com profissionais de TI, e a apresentação prática de um único estudo de caso.

A recolha de dados quali-quantitativa por inquérito foi feita a profissionais no LinkedIn e a um grupo intitulado “Mulheres de Produto” do Slack.

O estudo apresenta como resultado variados métodos e ferramentas complementares ao Design Thinking e ao Scrum para prevenir problemas de software. Foi revelado que times multidisciplinares tendem a adaptar-se de forma ágil e criativa devido à combinação do Design Thinking com o Scrum sempre com a participação do utilizador.

De acordo com o público-alvo desta investigação, o valor percebido pelos utilizadores é satisfatório quando observadas as necessidades destes por meio da conjugação das entrevistas, da validação do MVP (Minimum Viable Product) e das sprints backlogs completadas.

As implicações diretas deste estudo são a comunicação eficiente e a capacidade analítica do time, a participação do utilizador, a entrega contínua, e o desenvolvimento de software com uma proposição de valor intrínseco. A limitação está no fato de o público-alvo ser de profissionais brasileiros, exclusivamente, e pelo fato de apresentarmos um único estudo de caso.

Palavras-chave: Desenvolvimento de software, Qualidade de software, Proposição de Valor, Scrum, Design Thinking.

Abstract. This dissertation aims to associate Design Thinking with Scrum for the improvement of software quality. The associated methodologies collaborate to increase the speed of the backlogs to the user client in the initial phase of a software development project. The research methodology is the case study through interviews and a survey with IT professionals, and the practical presentation of a single case study.

The quali-quantitative data collection was done by submitting a survey to professionals on LinkedIn and to a group titled "Mulheres de Produto" by Slack.

The study presents various methods and tools complementary to Design Thinking and Scrum preventing software problems. It has been revealed that multidisciplinary teams tend to adapt quickly and creatively because of the combination of Design Thinking and Scrum always with user participation.

According to the target audience of this research, the value perceived by users is satisfactory when their needs meet by combining interviews, validating the Minimum Viable Product (MVP) and completing sprints backlogs.

The direct implications of this study are efficient communication and analytical capacity of the team, user participation, continuous delivery, and software development with intrinsic value proposition. The limitation is the fact that the target audience is exclusively brazilian professionals, and that we present a single case study.

Keywords: Software development; Software quality; Value proposition; Scrum; Design Thinking.

Acknowledgements

To my husband for all the enthusiasm and understanding along the way.

To my thesis supervisor for his permanent incentive and availability.

To the interviewees from LinkedIn and Slack's group "Mulheres de Produto" for the solicitude.

To Mr. Alex Lima Campelo, CEO from JMJ Sistemas e Consultoria, for his partnership.

Table of Contents

1 Introduction	1
1.1 Research Question	3
2 Theoretical Background	4
2.1 Software quality.....	5
2.2 Scrum.....	7
2.3 Design Thinking	11
2.4 Scrum and Design Thinking	13
2.5 Discussion.....	19
3 Research Methodology	21
4 Design	25
4.1 Research Problem	26
4.2 Specific Objective.....	26
5 Prepare	28
5.1 Interviews.....	29
5.2 Survey.....	29
5.3 Single Case Study.....	30
6 Collect	31
6.1 Interviews.....	32
6.1.1 How to correct quality problems in software.....	33
6.1.2 The experience of using Design Thinking with Scrum.....	34
6.1.3 Problems of using Design Thinking with Scrum	35
6.1.4 Other tools, platforms, and practices.....	36
6.1.5 User participation	37
6.1.6 Scope changes	37
6.1.7 Sprints to deliver a release.....	38
6.1.8 Software quality improvement metric	39
6.2 Survey.....	41
6.2.1 Summary.....	52
6.3 Case Study	53
7 Analyze	60
7.1 Interviews.....	61
7.2 Survey.....	62

7.3 Case Study	65
7.4 Discussion on Related Work	66
8 Share	68
9 Conclusion.....	70
Appendix I Interview Guide	78
Appendix II Survey	80
Appendix III Case Study – Consent letter	85

List of Tables

Table 2.1 - DMAIC Model.....	6
Table 2.2 - Analytical Thinking vs. Design Thinking.....	12
Table 2.3 - The Right Conditions for Agile	14
Table 6.1 - Practitioners Profiles.....	32
Table 6.2 - DMAIC Model (time-consuming steps).....	44
Table 6.3 - Team stages and User's collaboration.....	46

List of Figures

Figure 2.1 - Scrum framework.....	10
Figure 2.2 - Problem and Solution spaces in Design Thinking.....	11
Figure 2.3 - Design Thinking and Scrum integration.....	16
Figure 2.4 - Team constellation in the Empagile.....	17
Figure 2.5 - Hybrid of Design Thinking and Scrum.....	18
Figure 3.1 - Case Study Process.....	23
Figure 6.1 - Job Titles/Skills.....	42
Figure 6.2 - More frequent software quality problems.....	42
Figure 6.3 - Scrum adopters.....	45
Figure 6.4 - Design Thinking adopters.....	47
Figure 6.5 - More used Design Thinking tools.....	48
Figure 6.6 - Design Insights.....	48
Figure 6.7 - GPS System.....	53
Figure 6.8 - Why and How questions.....	54
Figure 6.9 - Storypoints.....	54
Figure 6.10 - Burndown.....	55
Figure 6.11 - Control chart (tickets, bugs, team).....	56
Figure 6.12 - AS IS and TO BE.....	58
Figure 6.13 - Revenue Dashboard.....	58

1 Introduction

The perception that the quality of the software produced by information technology companies usually generates dissatisfaction to customers has become an attention point and an opportunity.

Software quality involves aspects of design and services that produce genuine delights for customers. It is in the phase of surveying users' needs that incomplete specifications are made, leaving doubts that persist during the development cycle.

Contrary to what should be, throughout the development process methodologies are not applied to advocate client collaboration, with rapid responses to changes, and innovation practices.

Based on the observed data and practices as a result from the Case Study methodology adopted on this thesis, we intended to gain a more in-depth insight into Design Thinking to prove that it is indeed a viable method to create solutions with quality and innovation in a company with an Agile mindset, precisely Scrum method.

The scope of a software development project includes not only the size of the project but the size and experience of the team and how they react on embracing change and often releasing.

The combination of Design Thinking and Scrum to be applied in a company is about team dynamics associated with creativity for high performance in agile environments, sharing a vision of how the software will be built.

The underlying principle here was that real working software is much more valuable to end-users than a stack of comprehensive documentation focused on delivering functionality.

Namely, the objective of our thesis was to prove that the technical solutions derive from an adequate initiate phase of Scrum Project when scope defines the quality attributes of users in the system regarding product vision, performance, usability, guarantee, security, availability, maintenance and technologies involved.

1.1 Research Question

The main research question of this thesis is, **do Scrum and Design Thinking improve software quality?**

As a cross-question, we also wanted to study how user's quality perception could be measured with the practices mentioned above associated with delivering a value proposition in an incremental, iterative way, to which the client can validate the solution.

In order to perceive more about the Design Thinking and Scrum methodologies addressed to software development and the impact on user's satisfaction, the most used methods and tools were investigated.

Therefore, our target audience were experts' practitioners who are software product management of both methodologies, Scrum and Design Thinking.

The scope of software quality metrics in this thesis will be restricted to the evaluation of methods and tools: "(...) evaluation's success depends on good experimental design, proper identification of the factors likely to affect the outcome, and appropriate measurement of factor attributes." (Fenton & Bieman, 2015).

The research will continue in order to understand the problem of the thesis and, consequently, the reasons for the customer's dissatisfaction when evaluating the current situation of software production.

This thesis is structured as follows. It is first showing the Theoretical Background, in Chapter 2. In Chapter 3 is presented the Research Methodology that is going to be addressed. In Chapters 4, 5, 6, 7, and 8 are presented the Case Study phases, respectively, Design, Prepare, Collect, Analyze, and Share, followed by the conclusion in Chapter 9. Some of the limitations of the study are also mentioned, as well as suggestions for future research.

The interview guide, survey, and a case study's consent letter are toward Appendixes.

2 Theoretical Background

In this Chapter, the main concepts related to the thesis were described. In sequence, in subsection 2.4, will be presented the work already done by some authors in this context.

2.1 Software quality

The ability to reduce the backlog is a measure of effectiveness, in the scope of companies whose end-activity is the provision of IT services.

Concerning operational efficiency, the prices charged for the services provided are high. The inputs used to produce software raise prices and limit the volume of products and services generated.

At the same time, the increase in the cost of customer service is not accompanied by an increase in satisfaction and perception of quality with this service.

In other words, in companies that demonstrate more concern about the contract, the service levels are inversely proportional to incidents reported by customers, a fact that causes frustration to the end.

The issue under analysis affects the essential characteristic for the service provided by a software supplier company, the economicity, in two aspects: in the productive process of these companies and in the prices paid by the clients that contract them to create solutions.

Regarding processes, there is not also an empathic lack of tools to measure critical activities for the development of systems, such as indicators related to the process of quality management and customer service. (Häger *et al.*, 2015)

By neglecting the iteration phase, where frequent testing with the client should be performed, the side effect is the lack of a systemic view. It corroborates to the inadequacy and the limited control of relevant information, such as the number of function points, programming language, demand situation, fault log, start and end of service; limitations that make it difficult to focus on solutions.

The delivery of software with quality that meets the needs of customers is a permanent challenge to companies and software development teams. A Fleming

(2016) contribution in describing the DMAIC model (Table 2.1), to define the quality level of software for process improvement was taken as a premise for the present thesis:

Table 2.1 - DMAIC model

DMAIC Model	Definition
DEFINE	the system, the voice of the customer and their requirements, and the project goals, specifically.
MEASURE	key aspects of the current process and collect relevant data.
ANALYZE	the data to investigate and verify cause-and-effect relationships. Determine what the relationships are, and attempt to ensure that all factors have been considered. Seek out root cause of the defect under investigation.
IMPROVE	or optimize the current process based upon data analysis using techniques such as design of experiments to create a new, future state process. Set up pilot runs to establish the process capability.
CONTROL	the future state process to ensure that any deviations from the target are corrected before they result in defects.

Source: adapted from Fleming, I., 2016.

Fleming (2016) explain that the goal of the process improvement project for software quality characterization is to identify procedures and standards that are subjected to verification by software quality control, using internal metrics.

According to Jones & Bonsignour (2011), among the attributes of quality can be found in these ten qualitative metrics:

1. Elegance or beauty *in the eye of the beholder*
2. Fitness of use *for various purposes*
3. Satisfaction of user requirements, *both explicit and implicit*
4. Freedom from defects, *perhaps to Six Sigma levels*
5. High efficiency of defect removal *activities*
6. High reliability *when operating*
7. Ease of learning *and ease of use*
8. Clarity of user guides *and HELP materials*

9. *Ease of access to customer support*

10. *Rapid repairs of reported defects*

Denning (2016) shows six distinct levels of software quality assessment that reflect different degrees of emphasis on user satisfaction:

- Level -1: No trust
- Level 0: Some trust, begrudging use, cynical satisfaction
- Level 1: Software fulfills all basic promises.
- Level 2: Software fits environment
- Level 3: Software produces no negative consequences.
- Level 4: Software delights

At last, the concept of the economic value of quality is a common sense through authors analysis when the pre-requisites are intrinsically human-centered needs, and how well it complies with or conforms to a given design.

2.2 Scrum

Comparing the traditional practice of waterfall to the agile methodology shows that in waterfall users feedback comes in a later stage of development when changes are more expensive to the customer (Dhir, Kumar & Singh, 2019). As an alternative, in Scrum agile method, the creative ideation must be considered by a multidisciplinary team already in the phase of software scope specifications.

At the same time, they have the agility and the ability to adapt fast, not only dividing IT software development into no iterative alignment phases: requirement and specification, program design, coding, testing, and implementation; but considering users' feedback along the process.

Scrum asks for an all-embracing one-team approach in which all disciplines involved in the development process (architects, developers, tester, documentation experts, ...) pool their resources all the way through (Lindberg, Meinel & Wagner, 2011).

End-user features must be raised to help technical stakeholders (software development team) to produce software that brings value to the user. Different types

of stakeholders contribute in a complementary way during the elaboration of the artifact to tackle more complex problems, as defined by Pinheiro *et al.*, 2018.

Software teams that have lost the ability to communicate what they are building often seem to lack technical leadership, direction, and coherence about the core handover problems (Khan & Kajko-Mattsson, 2012). So, to ensure that everybody is contributing to the same end-goal is necessary to be able to communicate the vision of what are they building effectively.

The vision comes from an organized list that contains everything, from the user needs, the product should have, the Product Backlog.

The Product Backlog not only enables stakeholders to establish a vision but also to decide what desirements the team should address during the next sprint. By making desirements visible and explicit, the Product Backlog ensures a shared understanding. By clearly setting the priorities, it is much easier for the team responsible for “How” the software will be built to plan and monitor its work (Cardinal, 2014).

Elicitation of the needs and stakeholders involved in this process is part of the discussions about the tools to gather better scope management. Develop Epics is part of the initiate phase of Scrum processes related to the initiation of a project.

Cohn (2010) understands that a Scrum team needs to discuss the Product Backlog more frequently for leading to greater buy-in by all team members. Epics turn into User Stories and shift the focus from writing about features to discussing them.

User stories are one of the primary development artifacts for Scrum project teams.

Gannon (2013) explains Sprint Planning start with a set of user stories that had a similar theme and that when group together could result in a demo-able product.

As the Product Backlog, the Sprint Backlog is an artifact of Scrum. After the Sprint Planning is necessary to define what can be developed in the incremental deliveries as potentially releasable functionality (Sutherland & Schwaber, 2017).

During early sprints, Cohn (2010) concede that it is often difficult to find ways to demonstrate the value of this work to end users, but it's OK to sometimes struggle

in that regard, especially early on. Just because something is hard is no reason to abandon it. Instead, find ways to split those early infrastructural pieces into smaller pieces that can fit within a sprint.

Progressively, the team refines the requirements and user stories. After this final split, the team feels the stories are small enough to complete during a sprint and stop there, accomplished with satisfaction.

A coworking delegating tasks to individual members, determining delivery dates, attaching documents, viewing activity history, making comments, and sharing the task board is allowed. The target is to have some of the work made each sprint to result in features that users can see.

At the end of each sprint a Sprint Review Meeting is done. During this meeting, the Scrum Team shows what was achieved during the Sprint. Typically, this has the form of a demo of the new features.

The Product Owner and team member are responsible for understanding enough detail of stories in the backlog priorities to augment during the discussion for each sprint.

There are other aspects of the process such as a daily standup meeting where developers explain what they did, what they are going to do, and any problems.

Prioritization of tasks, assigning tasks, process improvements are some other processes to be conducted by the Scrum Master who is in charged with addressing any barriers to progress that come up.

During a Sprint, Scrum Master keeps the Sprint Backlog updating it to reflect what tasks are completed and how long the team believes it will take to complete those that are not yet ready.

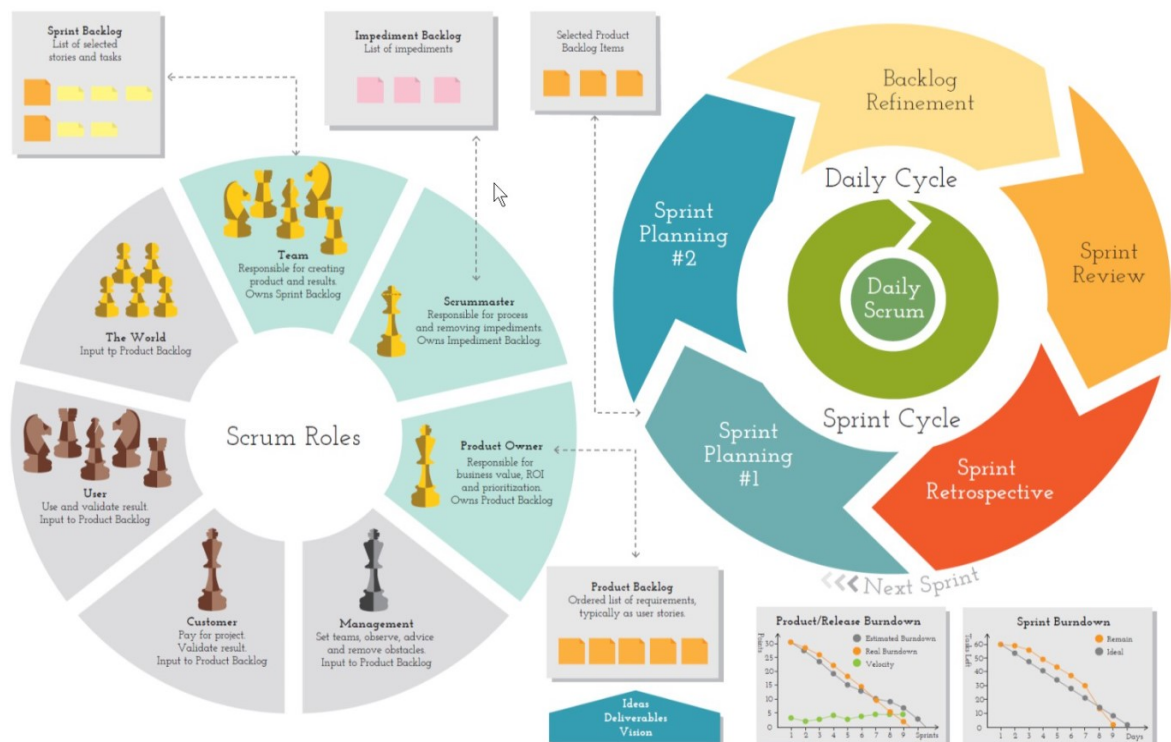
An estimate of work remaining to be done on Sprint is calculated daily and placed on a chart, resulting in a Sprint Burndown (Deemer *et al.*, 2012).

The Sprint Retrospective occurs at the end of a Sprint and serves to identify what worked well, what can be improved, and what actions will be taken to improve (Diebold *et al.*, 2015).

Scrum helps to improve the existing engineering practices (e.g., testing practices) in an organization, for it involves frequent management activities aiming at consistently identifying any deficiencies or impediments in the development process as well as the practices that are used (Abrahamsson et al., 2002: 28).

The roles and practices can be seen in Figure 2.1:

Figure 2.1 - Scrum framework



Source: www.scrummaster.dk/wp-content/uploads/2015/05/ScrumFlow.jpg

Moe, Dingsøyr & Dybå (2009) got to the conclusion that transitioning from individual work to self-managing teams requires a reorientation not only by developers but also by management. This transition takes time and resources but should not be neglected.

Agile methods also require a cultural adaptation, so is advisable for employees to be prepared for this - in addition to leadership, because hierarchy of command and decision making is something that only hinders the efficiency of agile teams.

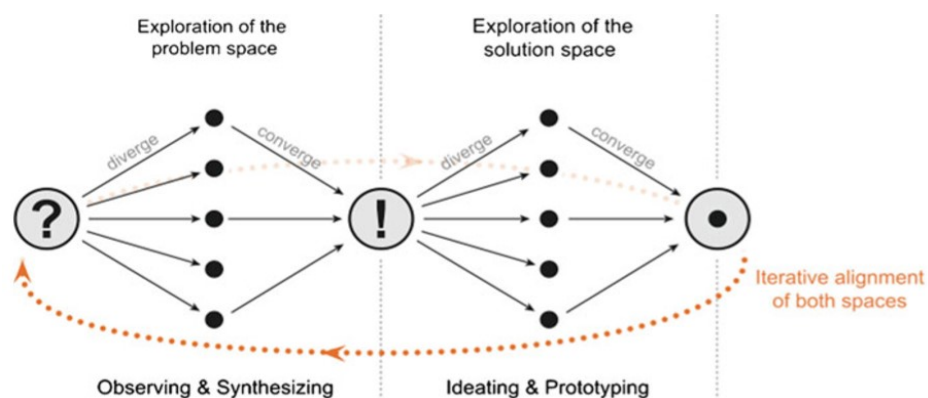
2.3 Design Thinking

Drawbacks can be avoided if software development has an approached research in human-computer-interaction based on innovation prospects (Lindberg, Meinel & Wagner, 2011).

The problem for the user relies on the inefficiency of the software and hence the rejection. Design Thinking brings a contribution as a problem-solving method, closer to everyday life, to create ideas with unique value and viable solutions to a specific group of users.

The perspective in a broad dimension of a problem is the opportunity to depict the users' needs in a creative diverging and converging practice of co-design (Figure 2.2).

Figure 2.2 - Problem and Solution spaces in Design Thinking



Source: Lindberg, Meinel & Wagner, 2011.

Creativity in Design Thinking is neither a representative (inductive) thinking nor a rationalized (deductive) thinking, is more the iterative practice of cognitive strategies with alignment between problem space and solution space. It maps how the design process passes from points where thinking and possibilities are as broad as possible to situations where they are deliberately narrowed down and focused on distinct objectives (Lindberg, Meinel & Wagner, 2011).

Razavian *et al.* (2016) recognize that creative design is about developing and refining the problem space and the solution space iteratively. Day-to-day, short

reflections are useful to adapt and adjust the strategies and assumptions of agile projects.

For Razavian *et al.* (2016) there are two minds, one that comprises the logical argumentation and the other concerned about the questioning and reflection of how we reason.

In Lindberg *et al.* (2012) the Analytical Thinking and Design Thinking suggest a paradigm for IT industry (Table 2.2):

Table 2.2 - Analytical Thinking vs. Design Thinking

	Analytical thinking	Design thinking
Problem perception	Ill-structured	Wicked
Relation problem/solution	Well-structured	Tamed
Key knowledge	Solution as a derivative consequence of a well-structured problem	Co-evolution of problem and solution
Key process	Expert knowledge	Stakeholder knowledge
Design paradigm	Defining and deriving	Iterations of observing and synthesizing, ideating and prototyping
	Rationalist problem solving	“Reflective conversation with the situation”

Source: Lindberg *et al.*, 2012.

Problem perception as assumptive Personas is a possible empathic practice to just an early iteration that will be verified and updated with scheduled interviews and surveys with customers. For Wallach & Scholz (2012) customers segment are defined by their behavior to build features that meet their needs.

After gain a fair amount of empathy from observing/interviewing users and defining Personas the next step is to transform insights into design questions. This step is the opportunity to generate as many questions as necessary to offer value to users based on the research insights. Some tools are useful for this attempt.

Sketch the objectives and features can be possible by using some of Design Thinking methods and tools cited from Chasanidou, Gasparini & Lee (2015), such as Personas, Stakeholders Map, Future Press Release, Blueprint, Journey Map, Pixar Storytelling, Double Diamond, How Might We, Crazy 8, Golden Path, 360 Lightning Talk.

After co-design sessions are time to prototype and test. Prototyping is about creating a Minimum Viable Product (MVP), or mockups, to capture the basics of experience and interactions. This is the stage to produce a tangible solution to be tested that would address a more loveable experience to the users and customers (Hokkanen, 2017).

This thesis intends to prove that Design Thinking and Scrum are complementary practices as customer-focused, analytics-driven to companies with a comprehensive value creation experience system.

2.4 Scrum and Design Thinking

In this chapter, we describe the work already done by some authors related to the main concepts of the thesis context.

To support process improvement some methods, such as Design Thinking, have the role in evaluating the organization's processes by taking the basis of some reference tool, which describes a set of principles and practices to be leading to better software.

Other methods complement the practice with measurements of quality to understand and evaluate the software produced to take actions that lead to the improvement of the process.

During the creation of the project vision, the composition of the team may need to be adjusted to match the work being done to impact into production that will lead to real change in the business. During the inception/warm-up of the project initiation, the requirements envisioning is composed of Epics to identify the scope of the system.

Try to cram the old processes into these new constraints, naturally, does not fit. They have to rethink the way they do software design. The expertise, talent, techniques, and tools were still very much necessary, but how they are executed, who are involved with them, and their timing, all require change.

When a system has a problem is necessary to correct the process that allowed to be inserted because, in this way, it would not be necessary to correct the same

problems in future work. Scrum can be placed as a practice of iteration to continuous improvement, as it will be possible to avoid fault occurrence.

At the same time, usability and user experience directly affect quality requirements and user acceptance as shown in Pinheiro *et al.* (2018) experimental study.

Rigby, Sutherland & Takeuchi (2018) synthesized Agile methods can be best applied in a complex context when the final solution might not be predictable.

Here is the turning point to a mindset change, once agile methods need training and potentially behavioral change, not just in the use of technology and approaches, but also concerning attitude toward contracting and involvement of clients and developers in project teams, as shown in Table 2.3:

Table 2.3 – The Right Conditions for Agile

CONDITIONS	FAVORABLE	UNFAVORABLE
Market Environment	Customer preferences and solution options change frequently.	Market conditions are stable and predictable.
Customer Involvement	Close collaboration and rapid feedback are feasible. Customers know better what they want as the process progresses.	Requirements are clear at the outset and will remain stable. Customers are unavailable for constant collaboration.
Innovation Type	Problems are complex, solutions are unknown, and the scope isn't clearly defined. Product specifications may change. Creative breakthroughs and time to market are important. Cross-functional collaboration is vital.	Similar work has been done before, and innovators believe the solutions are clear. Detailed specifications and work plans can be forecast with confidence and should be adhered to. Problems can be solved sequentially in functional silos.
Modularity of Work	Incremental developments have value, and customers can use them. Work can be broken into parts and conducted in rapid, iterative cycles. Late changes are manageable.	Customers cannot start testing parts of the product until everything is complete. Late changes are expensive or impossible.
Impact of Interim Mistakes	They provide valuable learning.	They may be catastrophic.

Source: Rigby, Sutherland & Takeuchi., 2018

From Table 2.3, is clear that dynamic environments are favorable to agile methodologies, involving company strategies and problem statement. This concept associated with Design Thinking formed the foundation of each operation mode of Häger *et al.* (2015) model to reach preliminary results in two university courses.

The operation modes are the Design Thinking Mode, the Initial Development Mode, and the Fully Integrated Mode. While the Design Thinking Mode emphasizes the Design Thinking phases and the Fully Integrated Mode focuses on software development, the Initial Development Mode aims at balancing the two kinds of activities.

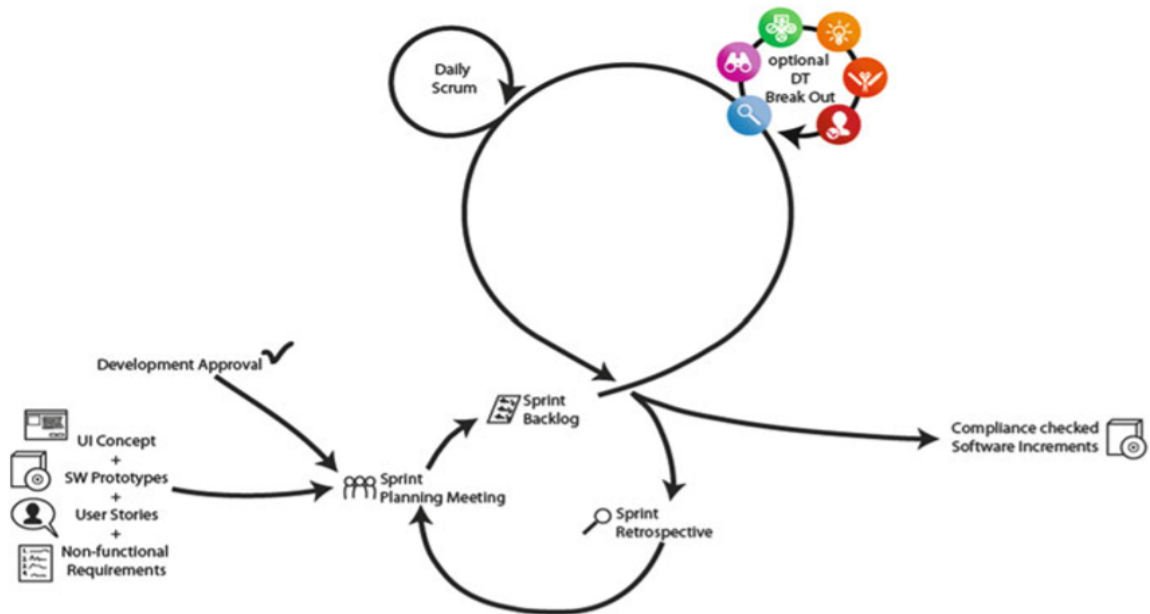
Design Thinking Mode used several techniques to understand the project environment and stakeholders: 360° research, Extreme users, Stakeholders maps, Persona, View Madlib, 2-by-2 Matrix, Brainstorming. Moreover, the sprints were accomplished by people with different areas of expertise.

Following Häger *et al.* (2015) model, for the Initial Development Mode, the teams should have a clear product vision, a set of high-level user stories, a functional prototype, and a non-functional requirement initial list.

Prototypes that provide a user interface should be tested with target users for maximum user satisfaction. Scrum teams were responsible for the planning and execution of the development sprints with a constant striving for user feedback (Häger *et al.*, 2015).

From Häger *et al.* (2015), Design Thinking and Scrum Integrated Mode denote the value of designing brought to the creative process and distributed to other roles on the team. As the teams mature, they realize that software design evolves from design facilitation as now a much bigger part of the process, at the same time as it is a broader understanding of all the other elements - performance, pricing strategies, customer management, and so forth - that encompassed user experience (Figure 2.3).

Figure 2.3 - Design Thinking and Scrum integration



Source: Häger *et al.*, 2015.

The experiment is comprised of two design challenges, each 1 hour long. In one challenge, the team decided how to use the hour themselves. In the other challenge, the team was required to use some time at the beginning of the hour to collect all tasks they want to do, assess them, and plan the course of the remaining time.

In the second version of the planned challenge, they asked the teams to take some time in the beginning to plan the hour. This questioning allowed the team to choose freely how much time to spend on planning and what techniques/tools to use.

According to the results from the survey, the participants' ratings for desirability and innovation potential of the solution between two challenges in three operation modes were higher in the second challenge. Also, the rating of stress in timelines were presented in a colored scale showing that teams found the first challenge to be less stressful.

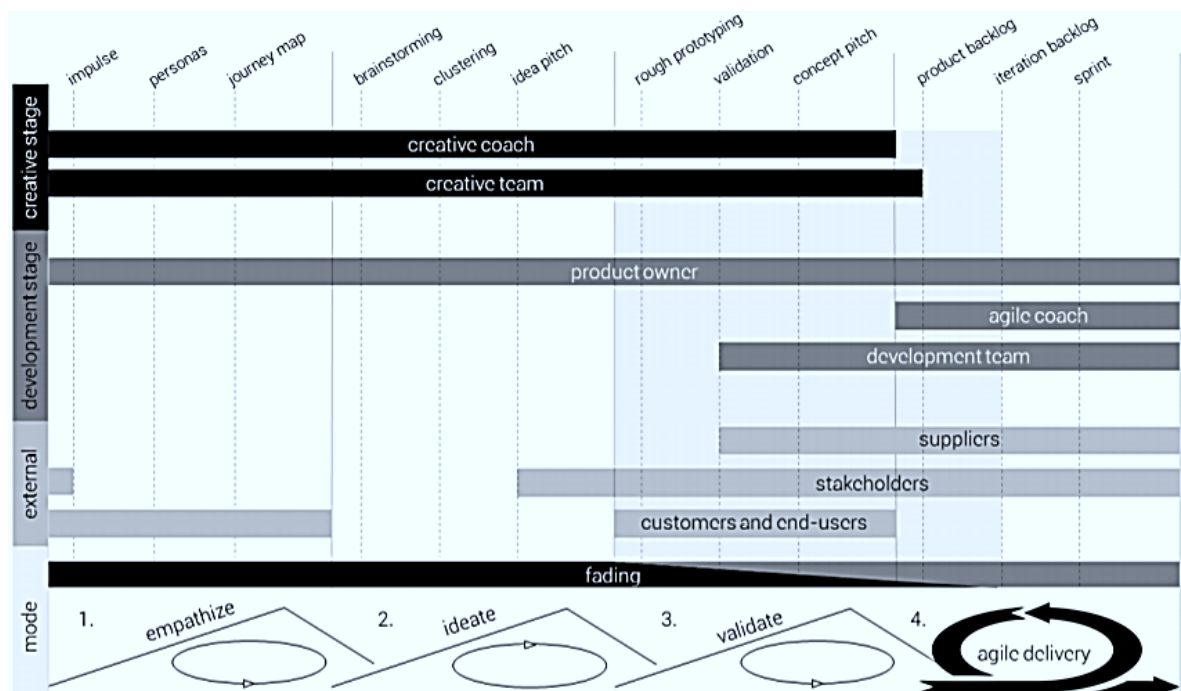
However, Häger *et al.* (2015) model is not clear of how to sum up the findings of the two university courses experiments could get to the conclusion of what was the quality perception value.

Grashiller, Luedeke & Vielhaber (2017) created a process for Agile product development, including a selection of empathic methods from Design Thinking. Consists of about 8 to 12 interdisciplinary creative members to ensure different roles and perspectives. In the Validate mode, there were at least four new concepts presented to the stakeholders.

Grashiller, Luedeke & Vielhaber (2017) signals that contemporary methodologies are often lacking in innovation focus and agility.

Three separate modes from creating and cluster insides, then by using ideation techniques to generate convergent ideas in the second stage. The third stage shows a divergent mode when prototypes are enhanced with customers and end-users, mutually agreed with stakeholders and the development team.

Figure 2.4 - Team constellation in the Empagile



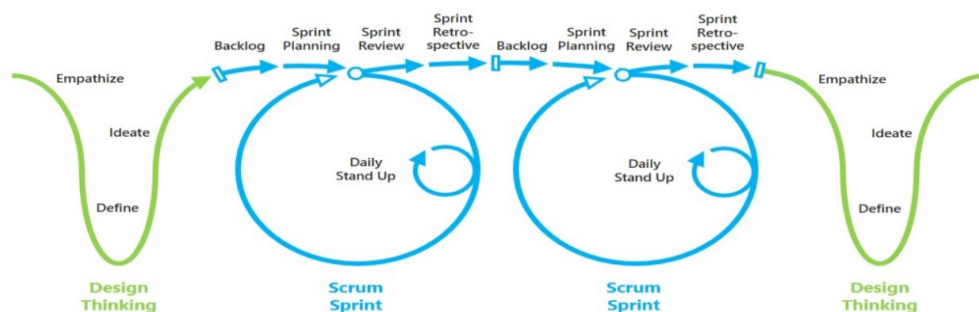
Source: Grashiller, Luedeke & Vielhaber, 2017.

However, as we can see in Figure 2.4, customers and end-users were part of the process only in Validate mode. The limitation is the fact that, as shown in the

practical example from the article, the customers and end-users' opinion were not taken into consideration in first and second stages. Their validation is extended to a third divergent mode for the iterative enhancement of the prototypes, but no metrics of quality or value perception were mentioned in *Empagile* model.

Yoshida (2018) shows another model that represents Design Thinking and Scrum as a convergent flow to this approach of what is a reasonable method to software development with intrinsic value to customers (Figure 2.5).

Figure 2.5 - Hybrid of Design Thinking and Scrum



Source: Yoshida, 2018.

The author infers that the result of this is a collaborative work and integrated approach to achieving a desirable, economically viable and technologically achievable result, left from somewhere of the client's mental model to apply techniques associated with empathy. Moreover, on this path, the error is part of the innovation process and is considered in Scrum Sprints as an essential step in software adaptation.

It follows prototyping, coming out of the abstract and going into something tangible. At the end of the acceptance tests, the author declares the result as a fantastic experience, with the materialization of the client's needs. However, no metrics support the hybrid model as a proof of improvement of software's quality.

2.5 Discussion

Häger, Grashiller and Yoshida models emphasize part of software quality problem, showing that it is necessary to investigate the value perception by the users.

Although in the discussion in Chapter 2 related to the association of Design Thinking with Scrum is not notorious if those models improve software quality with methods and tools available to measure it.

The value proposition for the relevant problem of software quality shows the utility of an artifact that combines Scrum approaches and Design Thinking to deliver value to the customer. It begins when the project vision statement serves as a basis for the development of Epics, in the initiate phase of Scrum.

Uncontrolled growth in a project's scope, at any point, after the project begins and a conflict between the process and the desired outputs could be avoided when the initiate scope phase is correctly defined in Scrum project.

At the end of the initiate phase, the Scrum core team reviews the user stories and determines the length of sprints. User stories are short requirements or requests written from the perspective of an end user (Diebold *et al.*, 2015).

The error in the first release generates incremental adjustments to the product in order to create value-added software. At the same time, Epics are refined elaborated, and then prioritized to create a Product Backlog for the project.

For the features and functions specified in the scope of the project to be comprehensive, client collaboration was required for the detailed product description to be as assertive as possible through validation testing.

We followed observing the continuous flow when assumptive Personas were updated to define User Personas by transforming insights into design questions. The purpose could be reached with a Divergent-Convergent Design. The result from prototyping software were releases to be validated by the users.

The quality characterization will depend on internal (reviews and inspections) and external (tests on production environment) metrics in an agile life cycle of development (Fleming, 2016)

Regarding the rigor of the research, in order to evaluate the usefulness, effectiveness, and quality of the software, quality control involves effective defect prevention, effective pretest defect removal such as inspections and static analysis, but not only (Jones & Bonsignour, 2011).

The first step sought to identify which approaches to Design Thinking could be used in software development projects, in the phase of surveying customer needs.

The second step was to confirm if the Design Thinking approach simultaneously with the agile methodology, specifically in scope initiate phase of Scrum, could be useful to evaluate the contribution to value deliveries with continuous improvement.

At the end of our proposition, the intention was to know if customer staff would justify if the software was suitable for the purpose intended or cannot be adaptable to user preferences and skills.

Among the experts that are the source of this research, part uses only Scrum, and part adopts the Scrum with some other tool to develop software for the customer. The proposal was to verify if the iteration of Scrum and Design Thinking from the beginning of the project planning was identified by the experts as possible to add value for customers.

3 Research Methodology

A Case Study Research Methodology is assumed as the guiding of this thesis to the exploration of a phenomenon and understand the problem of software quality.

As defined by Yin (2003), the case study is the method that aims to understand complex social phenomena, preserving the holistic and significant characteristics of real-life events.

Krefting (1991) informs that novice researcher should also plan for opportunities to have either a prolonged or intense exposure to the phenomenon under study within its context so that rapport with participants can be established and so that multiple perspectives can be collected and understood and to reduce the potential for social desirability responses in interviews.

Scientific research should seek to explain and predict what will happen in the world by seeking regularities and relations of cause and effect between the elements that constitute it based on positivist research.

According to the positivist paradigm (Lee, 1991), a hypothetical-deductive logic follows, that is, from prior knowledge, when gaps are identified; as unanswered questions. For these questions, hypotheses are generated, which are possible answers to the questions raised. These hypotheses are put to the test, trying to verify if they are false or true. To do this, one starts with the collection of data that will allow testing the hypotheses.

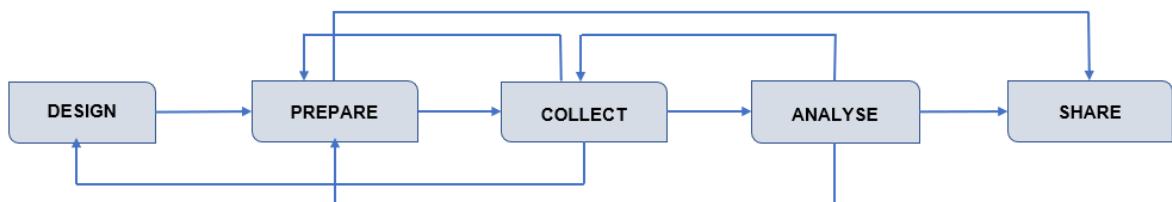
Based on Yin opinion's, the characteristics of the Case Study are:

- the phenomenon is observed in its natural environment;
- data are collected by various means;
- one or more entities (people, groups, organizations) are examined;
- the complexity of the case is studied intensely;
- no experimental controls are used;
- the researcher must specify the set of variables in advance;
- research involves the questions of how and why;

- does not consider prevalence or incidence;
- the study focuses on contemporary events;
- requires a problem that calls for the holistic understanding of an event or situation in question using the inductive logic, that is, the specific for the general.

Case studies are typically flexible design and process key parameters of the study may be changed while the study from Runeson & Höst (2008). The schema in Figure 3.1 summarizes authors major process steps in Case Study to the work that is going to be investigated in this thesis context.

Figure 3.1 - Case Study Process



Source: Adapted from Runeson & Höst, 2008.

In order to follow the research, we need to present a greater knowledge about the steps and planning in carrying out a case study, as follows:

- **Design:** A case study must begin with the theoretical framework related to the proposed objectives; be careful about generalizations and always seek scientific rigor in the treatment of the issue. The case study should not be considered exclusively qualitative. It can involve quantitative characteristics. All scientific research needs to define its object of study and, consequently, its methodological approach. The important thing is not to exclude the other methods, on the contrary, the combination of other techniques can benefit the research;
- **Prepare:** Definition of a research project and the presentation of its constituent components: study questions, propositions, units of analysis, the logic of the data, and criteria of interpretation and verification.

- **Collect:** Collection of evidence from the case study focused on six sources of possible evidence (documentation, records in archives, interviews, direct observations, participant observations and physical artifacts) and three principles of data collection (use of multiple sources of evidence, of data from the case study and the maintenance and linking of evidence).
- **Analyze:** Represents the selection, analysis, and interpretation of data. Yin states that it is convenient to consider all the evidence and be analytical, aiming at a functional analysis of the data collected. The selection of data should consider the research objectives and their limitations.
- **Share:** One difference between the case study and the other types of research is that the final case study report is a significant communication and propagation resource between experts and other stakeholders.

The thesis is structured in accordance with Design, Prepare, Collect, Analyze and Share phases of Case Study Research Methodology.

4 Design

Design chapter is the definition of the research project and the presentation of its constituent components adhering to the research problem, how delivering value as early as possible helps the customer understand what will bring more value in the near future.

4.1 Research Problem

A low productive efficiency has the effect of the difficulty of the IT companies in giving vent to the requests of the clients, being one of the factors that contribute to the high stock of pending demands of conclusion (backlog).

The end user's dissatisfaction stems from the existence of deficiencies in the provision of the system development service. Lindberg, Meinel & Wagner (2011) explain the rejection mean the client looks forward to a contribution as a problem-solving method, closer to the everyday life, to create ideas with value and solutions viable to a particular group of users.

This situation affects the quality of the software and is due to inadequate scoping in the initiate phase of the project when they establish the allocation of people, tools and services for the specified project according with user needs.

For incremental progress, the short iterations of Agile methodology, in a Scrum process with a feedback loop, the sprint, the evidence is gathered quickly to determine whether something is working or not. They constantly inspect and adapt the software to the stakeholders' specifications through short feedback loops (Cardinal, 2014).

In this sense, sprints and releases should converge to a better understanding of the stakeholders' perceptions by applying Scrum adapted with some other tools and practices.

4.2 Specific Objective

The specific objective of this thesis is to contribute to increasing the quality of on-demand software by proposing and evaluating an extension to Scrum based on

Design Thinking in order to improve the scope definition, by mapping Design Thinking in the initiate phase of a Scrum project.

This study aims to construct a concept to improve the software process as one of the top priorities for IT organizations who sell software, due to the market requirement for higher quality products, which are delivered more quickly and with less cost of development.

In order to design a software is necessary to understand the requirements, constraints and principles at a high-level, but mainly, detailing the scope in a basic level of understanding to help in reducing the number of options that are open, mainly to find the drivers to include the end-user needs and not fall in the scope creep.

Capture the sense of what is quality from users is the primary goal to achieve, in a sense to establish the difference between perceptions and expectations, from the beginning to the end of the software life cycle.

The business value perspective involves two objectives: more frequent major releases and more features in releases.

Using metrics to show the value of the features created for the users, with methods and tools for this purpose, is the ultimate goal of this research.

5 Prepare

This Chapter covers Prepare step of Case Study Research Methodology. It is an empirical study that seeks to determine or test a theory. It has significant sources of information employed to understand and learn more about Design Thinking and Scrum techniques applied to software development.

It is an investigation about a specific aspect, software quality, trying to find the characteristics and what is essential in it, gathering qualitative (semi-structured interviews), quali-quantitative (survey) methods, and a case study.

Interviews, questionnaires, artifacts, and documents review from a single organization were employed to collect and generate data with triangulation of methods and data collection (Harrison *et al.*, 2017).

5.1 Interviews

The interviews for this thesis follow a qualitative method and contains open-ended questions and prompts that, *prima facie*, appear relevant to the research topic, although the interview is conducted with flexibility in the ordering of questions (Madill, 2011).

The interview script (Appendix I) was organized with basic (primary) questions, in order to allow them to be complemented by other questions inherent to the momentary circumstances of the interview. As a semi-structured interview, the intention is to make information emerge more freely, unrelated to alternatives that can be suggested by the script used, allowing the respondents to be more spontaneous.

The practitioners were IT professionals with a more critical view of the business, which let them make good decisions quickly and efficiently, blending skills and responsibilities.

5.2 Survey

The survey (Appendix II) was based on the theme and objectives, to narrow the search and keep the focus on getting the information we need. The questions limit

the target audience since they are the ones who will dictate the language and terms used in the research, in addition to the subjects addressed in the questions.

After all, we need to ensure that the participants know how to respond to the questions, created on the hypotheses, for demonstrating exactly what we will have to ask to confirm or refute the predictions made (Lietz, 2010 & Stopher, 2012).

The inquiry was based on the interviews whose target audience were Brazilian professionals from the IT industry.

This type of research involves the quantitative and qualitative methods of research in order to have a broader vision and understanding of the subject studied. A qualitative-quantitative approach allows the researcher to achieve a cross-over of data, having more confidence in their data.

5.3 Single Case Study

A single case study about the Brazilian company JMJ Systems and Consulting is part of our research (Appendix III).

Baxter & Jack (2008) explain that a holistic case study is used to reflect the convergence of the results from our research. Also, as complementary reasoning, Benbasat, Goldstein & Meead (1987) investigation show how a practical and contemporary case can drift from a “(...) situation previously inaccessible to scientific investigation (...)” as theory testing.

6 Collect

This Chapter presents the results obtained through the data collected from the case study, interviews, and survey used in the context of the applied research methodology and corresponds to the Prepare, Collect and Analyze steps of the Case Study Research Methodology.

6.1 Interviews

This method is useful when the phenomenon to be studied is broad and sophisticated and cannot be studied out of the context where it occurs naturally. Through them, the interviewee will express their opinion on a specific subject, using their interpretations.

During March of 2019, we did 25 interviews from LinkedIn searched profiles with our master's thesis key words to know how the software quality is perceived by the experts involved in the development of software, and how they make choices to give valuable and viable solutions to the users.

Multidisciplinary profiles composed the twenty-five Brazilian professionals interviewed, and the answers were based on the experience and knowledge accumulated in different projects and to specific final users (Table 6.1).

Table 6.1 – Practitioners functions

Age	Profession	Years of experience
43	UX Designer	17
41	CEO	22
38	Software Engineer	15
36	Developer(Back end)	14
36	IT Project Manager	6
35	Product Owner	12
34	Scrum Master	10
34	Head of Innovation	8
33	Product Designer	7
32	UX/UI Designer	11
32	Marketing Manager	9
32	Product Owner	7
31	UX Designer	6
31	Scrum Master	6
31	Developer	6
30	System Analyst	10
28	Developer (Front end)	8
28	Developer	9
28	Tester	6
28	Agile Coach	6
27	Head of Agile Delivery	4
27	Develop Manager	4
27	Product Owner	5
25	Product Manager	4
25	Tester	4

Source: Elaborated by the author

The following questions were composed taking into consideration our thesis premises: software quality, Design Thinking and Scrum methods, and other tools, platforms and practices associated. User participation, scope changes, sprints, and metrics questions were complementary concepts to our study.

6.1.1 How to correct quality problems in software

The question was about the problem-solving process and the kinds of issues presented to have a firm grasp of this process along with any specific technical skills they may need.

Testimonies:

“Among the problems, they are: not to comply with the solicitation of the client, mistakes/defects in the execution of the software, lack of resilience, the difficulty of maintenance. Typically, techniques of demand refinement, tests, and revision of the code, for example, are adopted.”

“Consistency, feedbacks, problems of corrupted flows.”

“Problems usually arise from trying it moves fast - it will be an example, the scope gets cut (and maybe never picked up), the code is hacky if the foundation is not flexible, or things just get pushed out the door without necessary testing.

Solving these issues plows tricky, it is probably better it just avoids them in the first place. However, if have it correct them, it usually only eats it down the team, potentially rolling the feature back and putting in the necessary amount of effort. That is why it is always high; it tests features in the client’s environment before shipping directly it the general public.”

“Lack of vision of what the user needs, problems of usability, bugs. Solution: Carrying out meetings with the client.”

“However, generally, they had checked with more attention to the rules of business and requisites, if they are being carried out when it was established in the documentation of the project.”

“Poor development quality, unstructured or poorly structured architecture, obsolete services, APIs, and database. Bad projects structured, without processes, with poor management. Poorly developed or poorly specified products. All of the above factors entail a massive number of bugs and rework in the project. To correct, you must identify/understand the point that is causing the problems and correct and improve the process, both in the business and in the development teams. So, measure over the next sprints if the changes are getting the effect of expected improvement.

The inclusion of automation test and incorporation of regressive tests throughout each release also dramatically minimizes the number of problems, consequently increasing the quality of the software.”

“Problems:

- 1 - Prioritization of demands according to the needs of the client.
- 2 - Focus on the easiest to implement and not on the most important for the client.
- 3 - Planning does not go into the detail of demands and let pass problems that impact the estimated.

Solutions:

- 1 - Need to understand who the customer is and validate scope.
- 2 – Change the culture. So they could know that how to develop is essential, but it should match a better product for the customer.
- 3 - Improve techniques used to plan, go into details, and breaking into smaller pieces to understand the demands.”

6.1.2 The experience of using Design Thinking with Scrum

We wanted to know from experts about the experience of combining the concepts of Design Thinking and Scrum to develop adequate software to the client needs.

Testimonies:

“We use Scrum in software development and the Design Thinking techniques for the process of product discovery. We try to use the double track. A track is focused

on product discovery, with interviews, prototyping, tests. It is always the front of the track of development as it is what produces inputs in order to create the histories of the backlog which will be planned for the Sprints of Dev Team.”

“Generally, when it is a question of a new product, or new features, or of usability, we use approaches of cocreations to guarantee alignment of expectations - design thinking is one of them. To execution, the agile models such as the Scrum and the Kanban help mainly on the processes of path correction, since we recurrently revise the adherence of the expectations using the proposed ceremonies (mainly retrospective and sprint reviews).”

“For me as a Product Owner is excellent because we can identify from the beginning (conception) the customer real problems/pains and propose and develop products much more directed and assertive with the customers’ wishes.”

6.1.3 Problems of using Design Thinking with Scrum

We asked experts about the problems observed of using Design Thinking concerned to the initiate phase of Scrum when Epics are part of the scope to define the quality attributes of use.

“The Scrum is a methodology, and the design thinking is a mindset, then this is possible, but to have sprints with closed time-boxes maybe it can work when having a clear goal for each sprint. However, the tasks themselves maybe still not much tangible, depending on the user's inquiry.”

“Inferences are the most prevalent mistakes. To work with “proxy users” instead of effectively wrapping the one who in fact will use the software is a risk for the construction of the work plan. Another common mistake is to enter in design sessions thinking with taken decisions and try to influence through the products the adhesion to what only someone judge to be the best solution.”

“Scrum, Design Thinking and other agility tools are excellent tools for generating quality products/software, but in practice, if the company and the client do not have the right cultural fit, a good process structuring that can use the tools and the Scrum framework properly. Deadlines defined by the team and not from top-down, obsolete

legacy, etc., it does not do any good using those tools, and the quality will consequently be terrible.”

6.1.4 Other tools, platforms, and practices

We proposed to the specialists to mention other tools, platforms, and practices that can be used as a complementary technique to sustain software development.

Testimonies:

“Lean Inception”

“Design Research and Design Sprint.”

“Trello, to offer a quicker communication; Jira, like it being the software to register the information and progress through the Gantt; sprints of 15 days.”

“As tools, we use the Meistertask for tasks management.”

“We try to use the ceremonies, mainly daily, retrospectives to identify the improvement points.”

“OKR tracking allied with JIRA as a ticket tracking and setting up sprints.”

“Value stream mapping for operations areas seems to be a proper methodology of agile execution, as well as constant search for operational efficiency. DevOps also prints extreme operational efficiency in case of software development, increasing the delivery capacity for iteration of the teams very much.”

“We use the technique of the exploratory tests with the whole team to every 15 days to accompany the quality of how we are doing code iterations.”

“Customer Journey Map”

“We use the Bitrix for all management, but they do not have everything Scrum needs, so excel is used as a back-up tool, as well as a post-it whiteboard for activity control.”

“At times we use the pair programming, most used in XP. Tools to control sprint are VSTS and JIRA.”

6.1.5 User participation

We asked if users participate in the whole process or only in part of it.

Testimonies:

“With the tests of usability, validations (or invalidation) of hypotheses.”

“For products already in support, we used to access the users through inquiries, and we always read the reviews. For products in conceptualization, the first phase consists of investigating and understanding our target audience, through surveys, interviews, and tests.”

“The evolution of the features is always validated and accompanied by customers.”

“Always. The client participates with the teams as the Product Owner or owner of the product, and his primary interventions happen in the projection of the work, in the revision/reprioritizations of backlog, and in the reviews of the Sprints, to confirm the acceptance of the work.”

“Yes! However, the problems I face are unstructured processes and poor management of both parties, which makes delivery very difficult. In more structured companies with more defined processes, the error mitigation to increase quality is much more detailed and assertive. The search for quality is constant in every process, from design to delivery.”

“We are currently working more with the support team representing the customer. There are conflicts of interest that may not have existed if the experience were directly with the client.”

6.1.6 Scope changes

We made a question about the adaptative practices used by the specialists to keep the project going smoothly, just in case of scope changes during the Scrum project.

Testimonies:

“The needs related to the changes are revisited, reevaluating the adequacy to the new direction. If they are timely adjustments the person responsible for the product

realigns the items, otherwise it is necessary to join the whole team to devise/align again.”

“The changes are welcome in Scrum. A change within the Sprint that is running can lead to some ways or cancel Sprint if that change leads to the Sprint goal, not making more sense; or prioritized to the next Sprint. In my experience as PO, I never had a change that obligated us to cancel the sprint.”

“We have always been able to lead to prioritization. Primarily, analyzing the Points of Change, adaptation of the sprints with compression, or resource leveling.”

“If the scope expands, we have weekly team meetings to discuss progress and, at this point, we would discuss what would need to be cut in order to account for the unexpected extra work (or if the release date needs to shift).”

”Design Thinking to understand what the client wants or War Room when there is some very drastic change.”

“We use the backlog refinement, always in conjunction with the Product Owner. Team and PO meet to discuss reprioritization/substitutions, either due to the closeness of the deadline for delivery or due to budget constraints. The PO adjusts the backlog, and the developers begin to look for the items for prioritization.”

“I do not use a specific practice, but I try to follow a rule, if the change happens in a product or feature that is already developing in the sprint, the change goes into park lot and will be prioritized in the backlog. If it is not in development, it will enter the backlog and will be analyzed and prioritized.”

“Remove stories that end up losing priority when compared to the new scope, postpone sprint and overtime.”

6.1.7 Sprints to deliver a release

We asked how many sprints are necessary to handover a release, on average, to estimate a continuous delivery of releases.

Testimonies:

“It depends. I worked already with releases of 6 Sprints, with releases of 10 Sprints. It is what it does hurt for the product, for the business.”

“Two sprints.”

“Depends very much of the size and the necessity of the client.”

“2 per month.”

“Completely depends on the feature. Sometimes we run 1-2-week sprints and scope the work that we know we can complete in that timeframe, other times we make the best guess with estimates but ultimately we release the feature once it is ready.”

“On average, we hand over minimum viable products in even 8 Sprints. Features can be placed eventually in production daily, so many people for iteration.”

“Two sprints, and each sprint lasts two weeks.”

6.1.8 Software quality improvement metric

The question was about what metrics were used to estimate the software quality improvement by the users.

Testimonies:

“First, the quality is evaluated through the quality plan carried out by the project office. After that, the scope is evaluated together with the customer through deliveries.”

“Several bugs, changes, storypoints handed out, claims of the client in the introduction phase.”

“The main ones are NPS (Net Promoter Score), feedback, and the number of histories of user accepted by iteration.”

“Quantity of bugs after each release.”

“It depends on the hypotheses you defined establish to reach a certain objective for a particular user.”

“The best metric in this scenario is software working and producing the expected result.”

“NPS, App Evaluation, call opening amount (customer service), and interview to collect feedbacks. With those parameters is already possible to generate good KPIs of quality metrics.”

6.2 Survey

As was said previously, the problem research is the negative perception about the software quality by customers and the combination of practices - methods and tools - of development to deal with it. Thus, we elaborated a survey (Appendix II) to ground the mentioned investigation.

We collected data through interviews, and we did a survey based on that to deepen our knowledge about the production of software. The answers, the stakeholders involved, the phases of the process, and the tools, as well as opportunities for improvement and innovation, were complementary findings to provide specific knowledge to the research.

The mentioned findings were held by the literature where is presented how solving the right problem with user stories (Cardinal, 2014), the scrum team and events (Sutherland & Schwaber, 2017), and how iterations and increments help deliver working product (PMI and Agile Alliance, 2017).

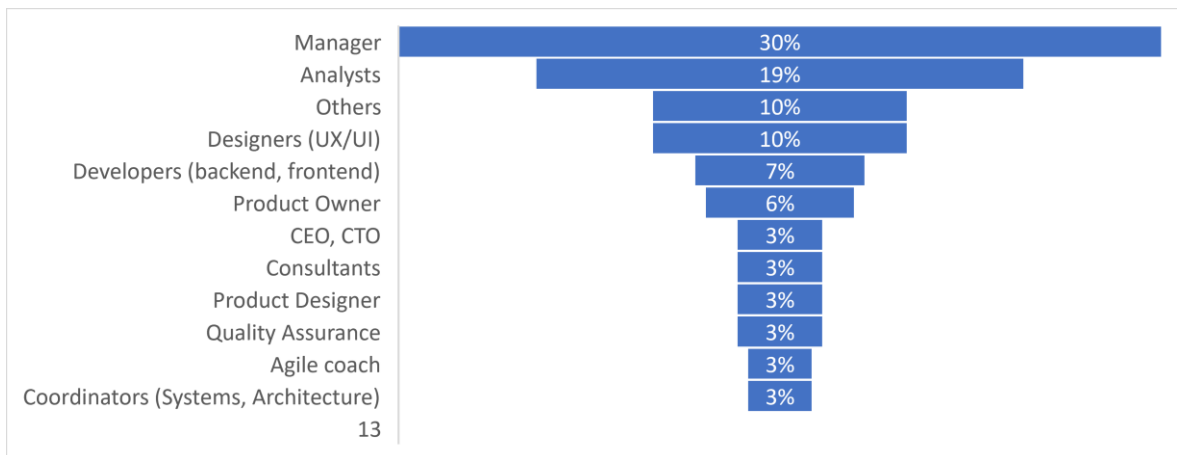
From the middle to the end of April 2019, an online Google form collected data for our investigation. The focus group of 120 Brazilian practitioners was composed of IT professionals from LinkedIn social networking and from a Slack's collaboration hub named "Mulheres de Produto."

In LinkedIn the 40 profiles selected were the result from the search of words "scrum" and "design thinking" on the search box. After choice the profile the next steps were to write a message to that person, introducing the thesis objectives and inviting to answer the anonymous survey.

The same survey was presented to "Mulheres de Produto" Slack community using *#general* direct message to 2.761 female members.

First, we asked about the experts' occupation, and the 120 answers reflected a broad distribution of skills, as we can see in Figure 6.1:

Figure 6.1 - Job Titles/Skills



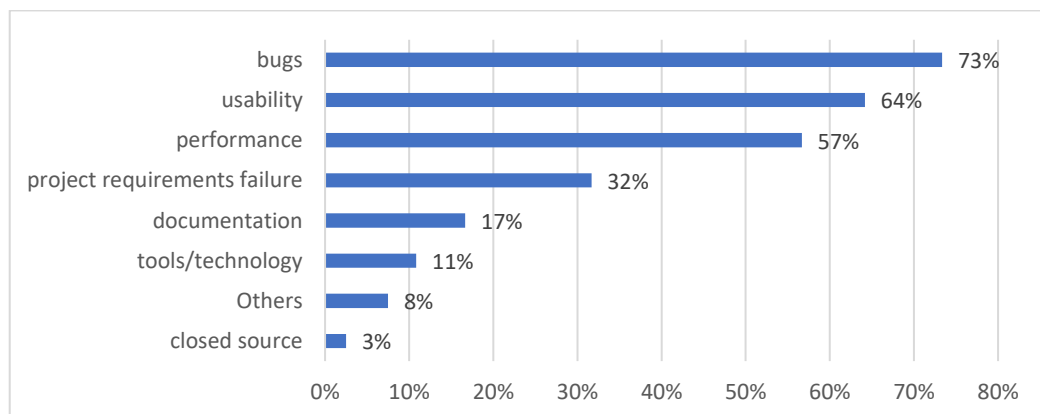
Source: Elaborated by the author

Analysts who were from Marketing, Business, Product, Project, Quality, Test, Data, and Architecture, correspond to 19% from 120 answers.

Managers and other job titles as UX & Design Thinking, Product, Process, Software Quality, Service Design, Innovation, Digital & Innovation, Design Research, Design Sprint & Design Thinking Facilitator, Head of Agile Delivery, and Scrum Master are represented in the rest of the sample showing the variety of experts from different areas of knowledge.

The next question was about software quality problems in terms of frequency (Figure 6.2), and 120 answered the question.

Figure 6.2 - More frequent software quality problems



Source: Elaborated by the author

Avoid problems is a permanent challenge to test the expert's ability to achieve business impacts in a structured way. Bugs, usability, and performance were mentioned as the most common problems in software development.

In order to understand the differences in performance and non-performance bugs is necessary to define some aspects, such as impact on the stakeholders, the context of the bug, the bug fix, and bug fix validation (Zaman, Adams & Hassan, 2012).

In so far as it applies to usability problems Tarkkanen, Harkke, & Reijonen (2015) concluded that the early stages of the development bring out more utility problem, very context-dependent and relying heavily on the procedure of testing with real users in actual usage context.

Question 4 was about what steps to solve the problem inspired in the DMAIC model, taken as a premise for the present thesis to help reduce inefficiencies while achieving predictable solutions. The result from 120 answers was the emphasis on the "Analyze" phase as the most intuitively observed by the practitioners to solve software quality problem in continuous improvement.

From Karout & Awashti (2017), the results from the Analyze phase can be specified by several techniques to specify the root cause of a large number of bugs found in production. Each organization defines what the more appropriated tools to this end.

Once the problem is well defined and measured, the Analyze step gives a complete understanding of what is causing the problem.

That is represented in Table 6.2 and implies that the root causes (define) and data (measure) work to prove or disprove their hypotheses. After that, Analyze is the most time-consuming step to test the best ways to combat these problems and create opportunities for improvement.

Table 6.2 - DMAIC Model (time-consuming steps)

DMAIC Model	Definition	%
DEFINE	the system, the voice of the customer and their requirements, and the project goals, specifically.	64,2
MEASURE	key aspects of the current process and collect relevant data.	65
ANALYZE	the data to investigate and verify cause-and-effect relationships. Determine what the relationships are, and attempt to ensure that all factors have been considered. Seek out root cause of the defect under investigation.	80,8
IMPROVE	or optimize the current process based upon data analysis using techniques such as design of experiments to create a new, future state process. Set up pilot runs to establish the process capability.	68,3
CONTROL	the future state process to ensure that any deviations from the target are corrected before they result in defects.	66,7

Source: Adapted from Fleming, I., 2016

The reasoning follows to define the team's profile and the knowledge necessary to give a solution for those problems that reflects the customer needs.

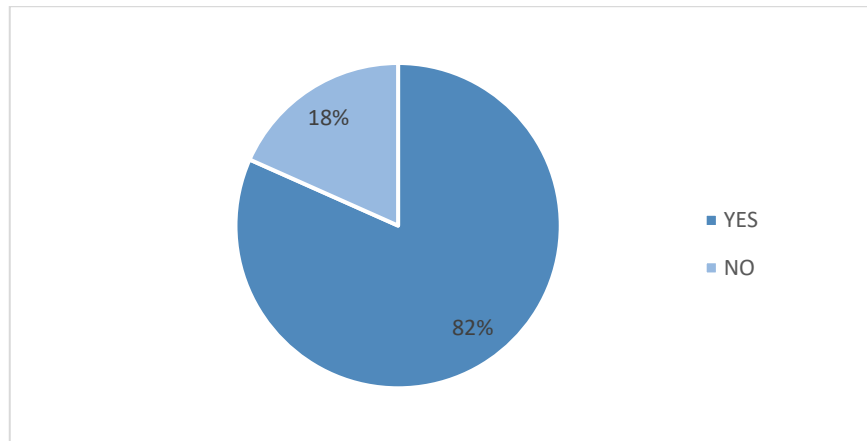
As related by the experts interviewed previously, especially in the universe of startups, a variation of a team is the "squad" which are teams co-located with a high degree of autonomy to make decisions in an organizational model that separates employees into small multidisciplinary groups with specific objectives.

From this point, on *Question 3*, was possible to identify the team or squad structure as a dynamic composition of 27% of developers, 17% of designers, 14% of product owners, 13% of quality assurance analysts, 10% of other analyst (business, requirements, product), 7% of scrum masters and 5% of product managers, based on 120 answers.

Jones & Thomas (2019) study finds that successful collaboration between designers and developers can be facilitated by focusing on the following factors: 1) Close proximity, 2) Early and frequent communication, 3) Shared ideation and problem solving, 4) Crossover of knowledge and skills, 5) Co-creation and prototyping and 6) Making joint decisions.

From *Question 11*, until further notice, the composition of the team is self-organized and multidisciplinary, and about 82% of all the experts confirmed to adopt Scrum methodology (Figure 6.3).

Figure 6.3 - Scrum adopters



Source: Elaborated by the author

In both cases, team or squad, everyone must understand the Backlog as a set of tasks to be prioritized and developed to generate the Minimum Viable Product (MVP) for the user to validate, as we concluded by the interviews.

Professionals who fill these roles work together daily to ensure proper flow of information and quick resolution of change.

The Product Owner role in an agile team is responsible for taking the most important decisions of the project according to the needs of the client. Product Owner provides business knowledge in the form of requirements for the team, as well as their order of application.

As a practitioner said, Epics are prioritized and refined by the product owner according to their vision of the product and understanding that it will bring more value (and faster) to the customer to enter the next sprint(s).

The Scrum Master aligns the team, defining well the role of each one, and ensures the agile culture as well as the unlocking of impediments.

In the sequence, a Sprint Backlog represents a contract established by the team with the items that will be delivered. Sprints can be synchronized and sequenced according to the team's convenience.

Then, in *Question 13*, 78 experts from 95 answers were affirmative when we asked if the team responds quickly to changes in scope during the Scrum project. A complementary *Question 14*, answered by 86 experts, was about the average required to deliver a release and the result is: from 2 to 4 sprints.

Regarding the sprints, in *Question 16*, we asked the experts to agree or not with the affirmative from the Theoretical Background chapter: "The target is to have some of the work made each sprint to result in features that users can see". We had 74 affirmative answers from a total of 94 Scrum adopters.

At this point, *Question 18* was answered by 120 experts concerned to what stage of the process and how the users work together with the team to improve software quality (Table 6.3), showing the tendency for the partnership to be permanent.

Table 6.3 – Team's stages and User's collaboration

Stages	%
All the time	35%
Tests	17%
Discovery and tests	16%
Planning, Review, Retrospectives	10%
None	9%
Discovery (Interviews and User Personas)	8%
At the end	4%
By the ceremonies	2%

Source: Elaborated by the author

A description to reflect the 35% of co-participation is when the client collaborates from the specification of requirements to customer validation. During this process, brief research will be conducted to complete and create personas, talking to some users and surveying the experience in past projects, creating MVP and putting the client to use it.

Tests for validation had a similar percentage of Discovery and Tests stage, 17% and 16%, respectively.

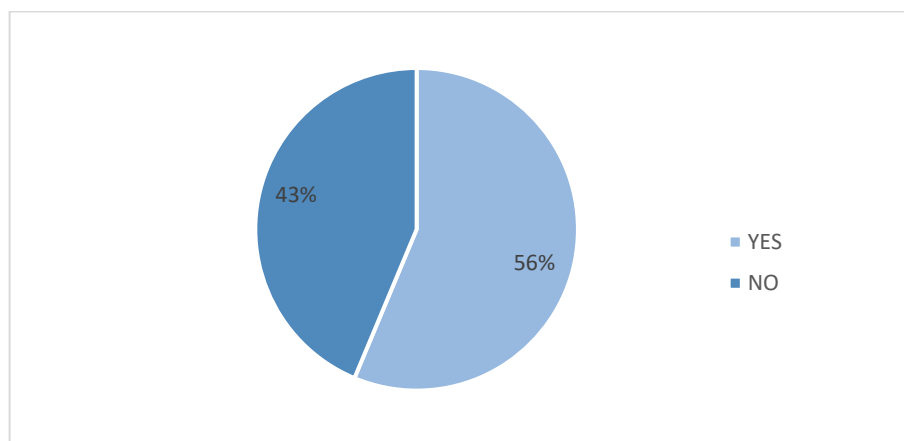
Test stage have metrics involving product acceptance survey and usability testing such as feedbacks, research, and NPS to confirm customer loyalty, resulting in a continuous improvement process.

The Scrum ceremonies Planning, Review, and Retrospective reflect 10% of the engagement steps when customer take part in the overall meeting to talk over the process and the solutions.

Question 15 was about Planning, Daily, and Retrospective Meetings to leverage how far those ceremonies help stakeholders to collaborate to the same objective; and 95 experts were convinced about its relevance.

Following our investigation, from *Question 5*, Design Thinking tools are usual in software development for 56% of the 120 practitioners (Figure 6.4). From all the affirmative responses, in *Question 6*, 63% said that they use Design Thinking in the initiate phase of software development, and the others stated that they use alongside development process.

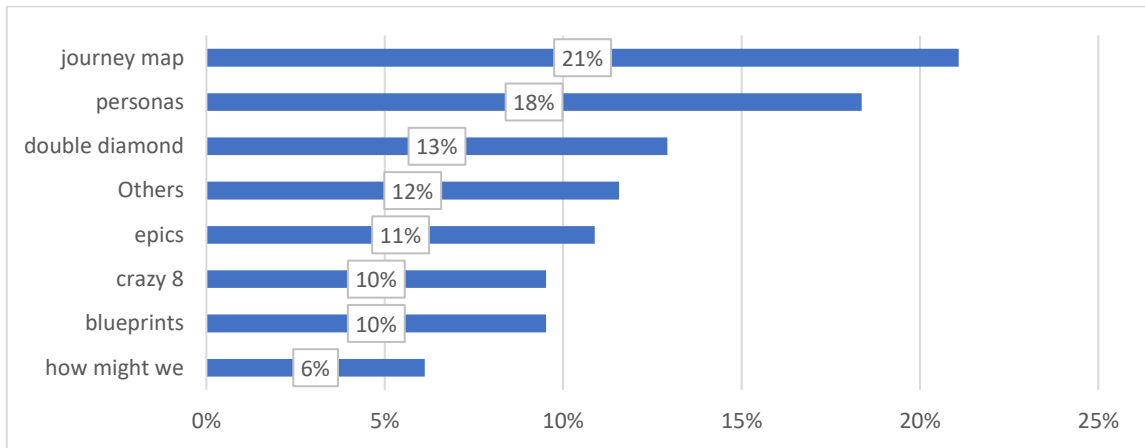
Figure 6.4 - Design Thinking adopters



Source: Elaborated by the author

The *Question 7* was about the relevance of Design Thinking tools: Journey Map and Personas were mentioned more frequently; this is, in 66 replies (Figure 6.5).

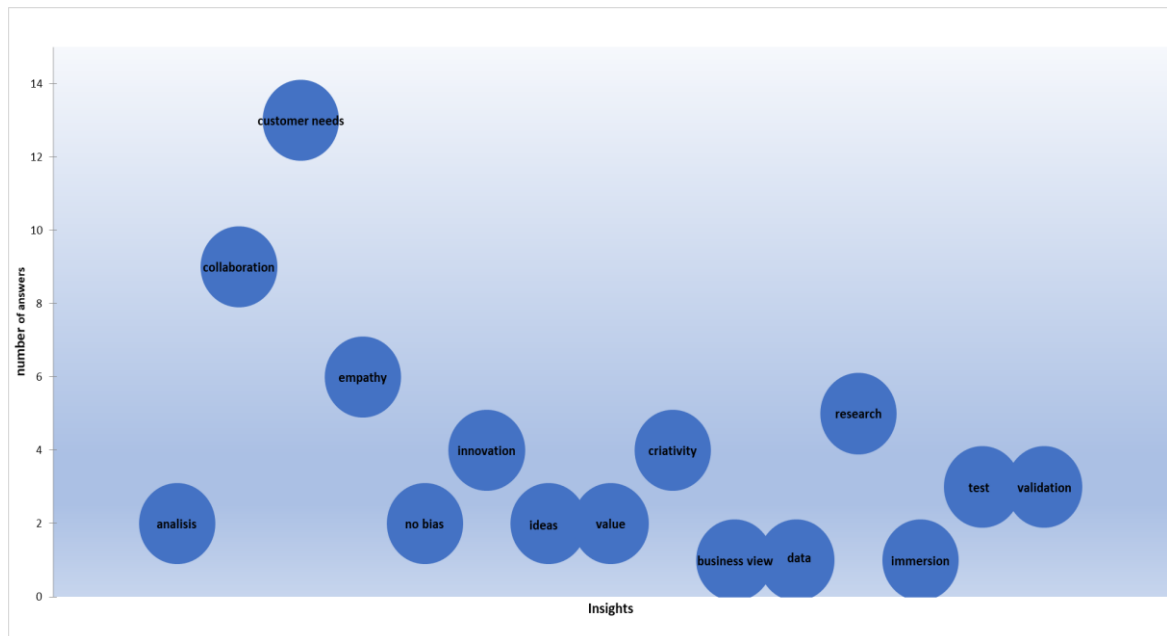
Figure 6.5 - More used Design Thinking tools



Source: Elaborated by the author

In *Question 10*, we asked experts to complete the sentence: “Turning insights into design issues involves...”, and the result, from 58 answers, convinced us that the participation of all the team and client or stakeholders (through research, usability testing, feedbacks, NPS) generates fantastic and innovative products. Therefore, the words more frequently cited are represented in Figure 6.6.

Figure 6.6 - Design Insights (number of answers)



Source: Elaborated by the author

Question 9 was based in an affirmative from this thesis, in our Proposal Description chapter, and evokes experts to criticize it: “The error in the first release generates incremental adjustments to the product in order to create value-added software. Epics are refined elaborated and then prioritized to create a Product Backlog for the project.”

The comments reflect the professional’s experience to enrich the investigation in terms of concept’s convergence, such as value proposition, epics, MVP, product backlog. Also, intending to be transparent and favor the discussion, we transcribe some testimonies:

“Value-added delivery cannot depend on incremental adjustments to provide that the delivery of the value is paramount since the first version. The most important is to define the purpose of the Product Backlog, there is no need to refine all the epics to prioritize; it is necessary to define the MVPs and releases.”

“Not only the error generates adjustments, but subsequent testing and monitoring of metrics will generate inputs for product improvement.”

“The continuous feedback from the user enables the product to be continuously improved and adjusted according to their real needs.”

“Make a quick mistake and re-test the hypothesis to clarify the way forward.”

“A backlog (even if minimal and prioritized) must be done even before the first version (MVP). It is correct to state that incremental adjustments must be made (in the next versions) to add more value, but based on research, feedback, and metrics. Moreover, the epics are prioritized, those ones should be refined because they are the stories associated with the epics for creating the Backlog and give a more precise definition of the roadmap.”

In terms of disadvantages of using Design Thinking, the 51 replies to *Question 8* can be resumed in the *time consumed* to role all the phases. The delay in creating certain functionalities, the client's resistance to carrying out and "stop" many people for a long time, customer response time, team engagement, not manage well the tools are some of the essential reasons why experts hesitate to put in practice Design Thinking methods.

Moreover, *Question 12* was a core question answered by 69 experts: ***How the conjunction of Design Thinking and Scrum can improve software quality?***

The answers resumed the context as the gain of the speed of deliverables, with quality, continually adapting them. They believe in the identification of problems and innovative solutions created.

Participation of all the team understanding the problem and designing solution, not only in the execution phase was a highlighted advantage for the experts.

In a holistic view, detailing the steps to follow until this point of the survey, Product Owner work on the prioritization of the deliverables. It is based on the challenges listed by the Design Thinking prioritized releases from Epics that go into development in one or more sprints, working on to organize Backlog to deliver immediate value.

Both team and customer can work together from the discovery till the construction of the prototype. After the prioritization in the backlog, the demands gain more agility and alignment. Each sprint should be the result of learning the previous one, adjusting what was not good, and questioning what is not well resolved.

The *Question 17* related to tools, platforms, methods, and practices was answered by 120 IT practitioners and reflects the most common frameworks and methods know by the teams to conduct software projects, besides Design Thinking and Scrum. The results were as follows:

- Most comments were about collaborative tools to share documents and files: *Confluence, Slack, Basecamp, Miro, Asana, Glip, Jell, Prodpad, Meistertask, Trello, TFS, and Cucumber;*
- *Jira* was the most common tool to track issue and bugs related to software;
- *Kanban* and Lean methods were more quoted, with the variations of LeanThinking, such as *Lean Startup, Lean Product, Lean Inception, A3 Hoshin Kanri, Gemba walk, SAFe;*
- *Design Sprint, Design Research, Service Design* to conceive an idea into something tangible and testable;

- *Management 3.0* (delegation Board, moving motivators, situation wall, agile wheel) as a redefinition of complexity management and focused on people, in an Agile methodology;
- Measure an application's source can identify significant trends with the use of code analysis and quality metrics (*TDD/ATDD/BDD/pair programming/XP/CMM*) in the opinion of some professionals. *Devops, ITIL, SWEbok* were complementary practices of software engineering;
- *OKR, Lead Time, KPI, Burndown*, and the digital metrics - *a/b test, conversation funnel* and *leads* – were combined to face the essential objectives and the results of the firm and to calculate the engagement of the potential customer;
- *Google Analytics, Hotjar*, and *Usersnap* were examples of site analysis tools that provides navigation data and user behavior through heat maps;
- Tools to provides a visually structured approach for scrum teams to manage product backlog were identified as the main translator of Epics tier: *User Storing Map, Storyboarding, Storypoints, Journey Map, Product Board, Jobs to be done, Stream Value Mapping, Sketches*;
- Communities of developers to host, discover, share, and build better software were essential to the practitioners, and platforms such as *Gitlab, Github, Azure, Bitbucket*, and *Unity*, are some of them.

Following the script, we asked the professionals, on *Question 19*, to criticize an affirmative sentence of our thesis “Capture the sense of what is quality from users is the primary goal to achieve, in a sense to establish the difference between perceptions and expectations, from the beginning to the end of the software life cycle.” All the 120 IT practitioners answered, and we present testimonies to elucidate the statement above:

“I understand that listening to the customer and not inventing a solution that he does not need is the key to everything. Listening requirements such as performance, usability, safety, quality, etc. are also critical, and usually, the people responsible for products only describe functional requirements for teams.”

“Complementary to this view on software quality: when a system solves a problem without creating new ones.”

So finally, by *Question 20* we wanted to know how the team, at the end of the process, can affirm the software is suitable or not to the client’s need, the 120 IT practitioners indicated:

- interview with the clients and MVP validated, by 57%,
- when the software is released to the production environment by 30%, and
- 13% by the sprint backlogs completed.

6.2.1 Summary

The interviews were a primary data collection to guide the survey’s questions. These questions intended to cover the multiplicity of dimensions presented in *Improving Software Quality using Design Thinking with Scrum* thematic, taking into consideration that reality is always complex.

The result was a wide range of tools, practices and complementary methods to facilitate and put in practice our purpose, which is to scale the client’s needs and value perception of quality.

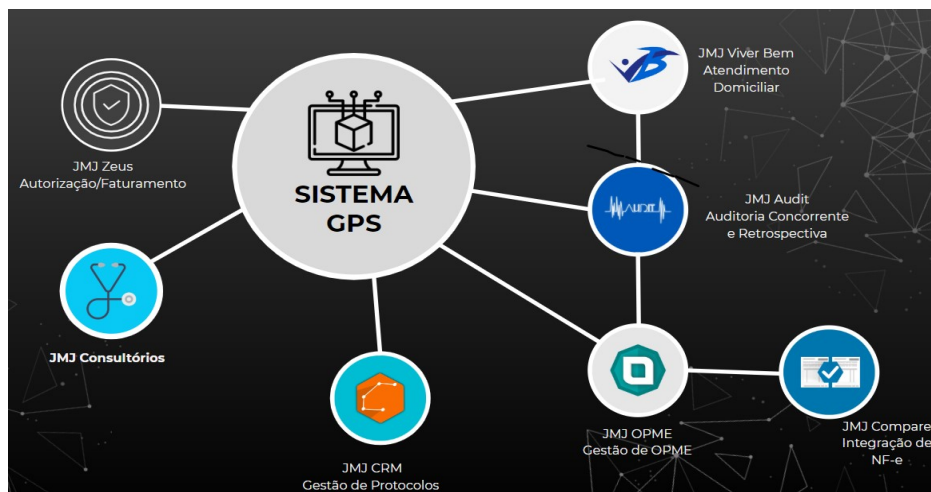
6.3 Case Study

JMJ Systems and Consulting¹ is a Brazilian company, founded in 2014, to develop solutions made and adapted according to the needs of the health insurance companies, aiming at ease and efficiency in the operational process, assisting the management process.

The objective is to align the improvement in health assistance, decrease in hospital stay and cost reduction.

The core system is GPS – Health Insurance Management, an Enterprise Resource Planning software - ERP integrated with other functional modules such as financial, audit, and CRM (Figure 6.7).

Figure 6.7 - GPS System



Source: JMJ Sistemas e Consultoria.

There are 15 professionals grouped by business, development, and technical support teams. JMJ intends to facilitate the required creativity enabling cross-functional teams with professional background to design and develop solutions.

JMJ uses Design Thinking for idealization, mapping the user profile, and prototyping to user validation. The user is part of the whole process.

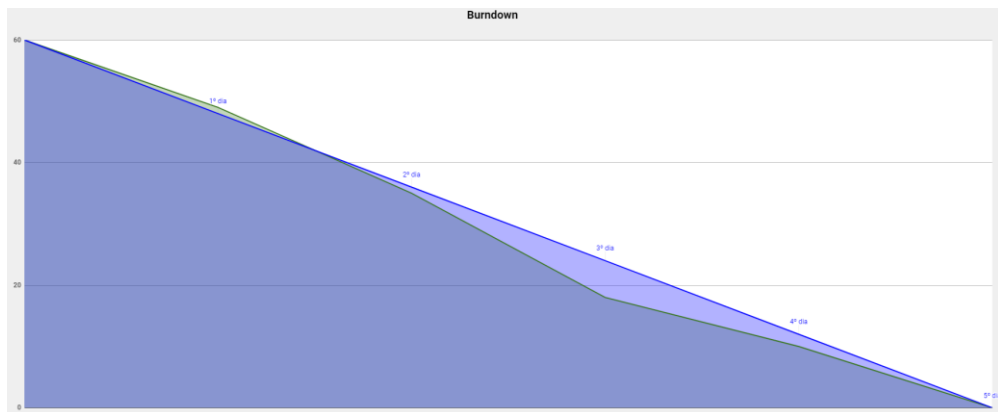
¹ <https://jmjsistemas.com.br/>

The average of productivity is measured according to the time spent for it, by a target parameter of:

- 30 hours for a new product development;
- 30 hours for maintenance;
- 30 hours for customization.

During the project is calculated the work already done versus the time by the Burndown graph (Figure 6.10). Often used in agile software development methodologies such as Scrum, it is useful to predict when all the work will be completed.

Figure 6.10 - Burndown



Source: JMJ Sistemas e Consultoria.

The graphic represents the amount of work remaining to be done on the vertical axis (y) versus the time on the horizontal axis (x). The Y-axis time unit can be in days, hours, week or sprints in the case of a burndown release, and the unit of X-axis quantity can be in working hours or periods.

The business team brings a pre-ready project after including the functional and non-functional requirements join through user story. Then, the development team works with a database analyst to explain how to develop the screen, functionality triggers, and so on.

The business team is multidisciplinary to avoid rework, adapted to specialized knowledge in health business rules and data structure.

The process starts with a one-week Design Thinking sprint, which effectively and efficiently leads the team through the problem and solution phase, starting with problem validation and finishing with a first tested prototype. Design Thinking sprints are usually repeated 2-3 times until the team have a final prototype which has been positively validated by the users.

The architectural specialists ensure that processes and technology solutions meet the demands of the organization, generating compliance and alignment with the company's strategic objectives and ensuring that IT team is doing the right actions to deliver benefits quickly (Figure 6.11) to be tested by the users.

Figure 6.11 - Control chart (sprint, tickets, bugs, team)

Backlog #1 - JMJ Alpeuz

Arquivo Editar Ver Inserir Formatar Dados Ferramentas Complementos Ajuda A última edição foi feita em 15 de abril por Tainan Seneda

Coluna	A	B	C	D	E	F	G	H	I	J	K	L	M	N
SITUAÇÃO	Sucesso													
OBJETIVO SPRINT	Módulo de arquitetura, configuração e cadastro do JMJ Alpeuz.													
Categoria	#	Ticket	Obs	Bugs	Situação	Executor	Tester	Início	1º dia	2º dia	3º dia	4º dia	5º dia	
Arquitetura	1	Arquitetura do sistema		0	Pronto	Tainan	Luziane	6						
Perfil	2	Pesquisa de Perfil		0	Pronto	Tainan	Luziane	1	1					
	3	Cadastro/Edição de Perfil		0	Pronto	Tainan	Luziane	2	2					
Usuário	4	Pesquisa de Usuários		0	Pronto	Tainan	Luziane	1	1					
	5	Cadastro/Edição de usuário		0	Pronto	Tainan	Luziane	2	2					
	44	Pesquisa de Usuários / Histórico de Acesso		0	Pronto	Tainan	Luziane	1	1					
Configuração	6	Configuração de E-mail		0	Pronto	Alan	Luziane	1	1					
	53	Configuração da Empresa		1	Pronto	Tainan	Luziane	4	4	4				
Embarcação	16	Cadastro/Edição de Embarcação		0	Pronto	Tainan	Luziane	6	6	6				
	7	Componente Autocomplete com Pesquisa de Cliente		0	Pronto	Alan	Luziane	2	2	2	2			
Cliente	9	Pesquisa de Clientes		0	Pronto	Alan	Luziane	3	3	3				
	10	Cadastro/Edição de Cliente		2	Pronto	Tainan	Luziane	8	8	8	8			
	11	Pesquisa de Formas de Pagamento		0	Pronto	Alan	Luziane	2	2	2				
Formas de Pagamento	12	Cadastro/Edição de Formas de Pagamento		0	Pronto	Tainan	Luziane	2	2	2				
	18	Cadastro/Edição de Vagas		0	Pronto	Alan	Luziane	5	5	5	5			
Vagas	19	Pesquisa de Vagas		0	Pronto	Alan	Luziane	3	3	3	3			
	57	Template de e-mail de novo usuário / esqueci minha senha.		0	Pronto	Alan	Luziane	1						
Layout E-mail	58	Fundo da tela de login.		0	Pronto	Alan	Luziane	6	6					
	8	Criação de Logo / Logo Pequeno / Favicon do Sistema		0	Pronto	Alan	Luziane	4						
Total				3				60	49	35	18	10	0	
Meta				7,5				60	48	36	24	12	0	
								Início	1º dia	2º dia	3º dia	4º dia	5º dia	
								19.mar	19.mar	20.mar	21.mar	22.mar	25.mar	

Source: JMJ Sistemas e Consultoria.

Once that is done, the development sprints start – they usually take 2-4 weeks and deliver a working product increment by the development team (front-end + back-end). After that, the MVP is validated as a ‘final product release’, making sure the customer value is always maximized meaning that the most value-adding features are the ones to be implemented first.

The company considers the best way to make lists is using applications to organize the tasks, and they are currently developing an integrated app to save everything about the projects in the cloud.

Following the internal process, the first sprint is broken in two to measure the burndown:

- First Validation (business team): prototyping, prototype revision, revision of user story, creation of requirement document, modeling (duration: 3 to 5 days).

After that, the business sprint is validated by the development team: with increasingly accurate validation, all speaking the same language, and a lean presentation to avoid rework allows a continuous flow of improvement. When the development team completes the coding phase, it will be the business team that will validate with user participation.

- Second Validation (technical support team): After validation and test the product vision document is revised after the business team approval of the version (duration: 3 to 4 days)

At the end of the homologation phase is presented the revision document to the client. After updated the version document with client opinion, also the script of the database is completely updated, generating the database script of that revision. So, for every new customer, the database is always up to date to meet customer needs and avoid rework.

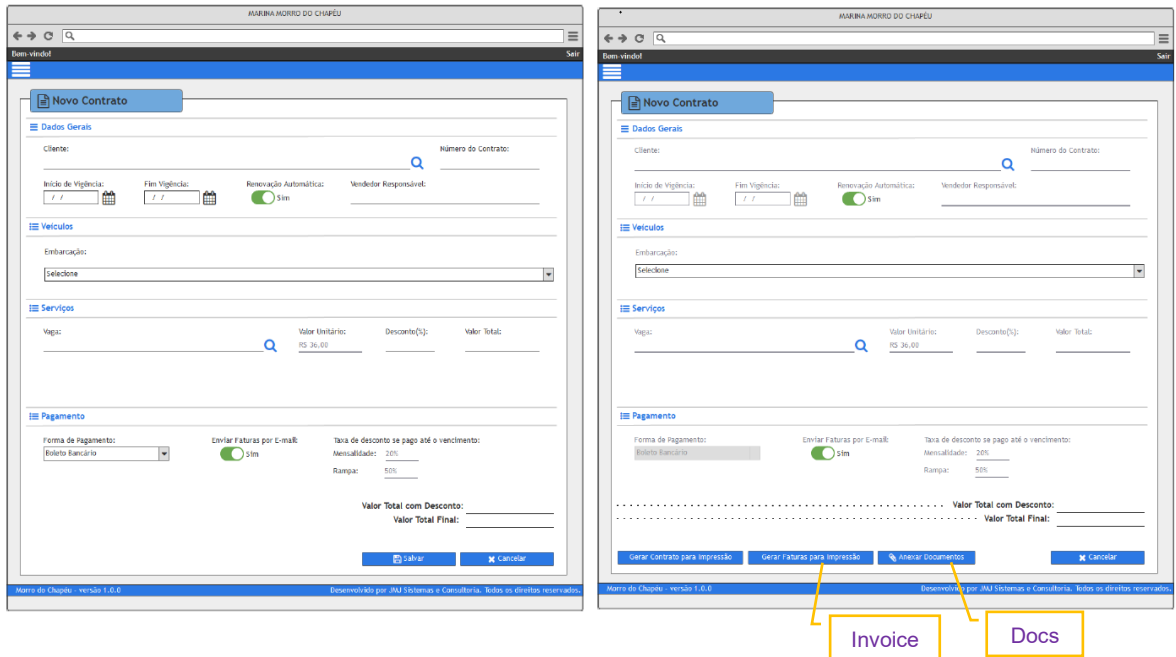
Other advantages are decreasing in bugs, cost reduction, and improvement of quality to differentiate themselves from competitors.

The Retrospective meeting points out what has gone wrong but must be managed if the user or team denote the same fault to several sprints. In this case, an action plan must be started.

The improvement of quality is perceived by the customer when it reduces his work.

An example of the operational impact can be seen in Figure 6.12.

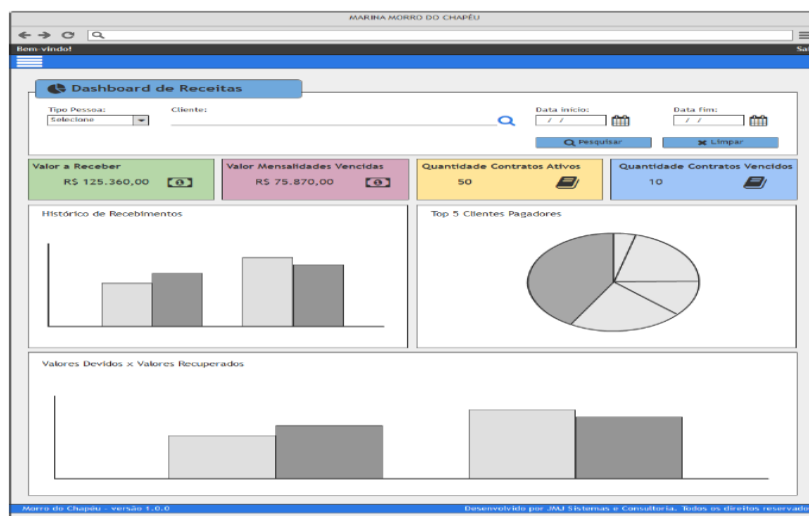
Figure 6.12 - AS IS and TO BE



Source: JMJ Sistemas e Consultoria.

Before the features, many employees were necessary to organize and gather all the documents of the contract, besides issuing the invoices for the payment of health insurance. Now is the system platform that makes it, including issuing management reports (Figure 6.13), bringing transparency.

Figure 6.13 - Revenue dashboard



Source: JMJ Sistemas e Consultoria.

This report was given to us by the CEO of JMJ Systems to explain their method to improve software quality along with Design Thinking and Scrum, and the client's contribution.

The positive impact of JMJ ERP Systems and Consulting for the clients is to raise the flow and connect the processes, since the provider, authorization team, purchasing team, audit, billing area, till the administrative area of health insurance. Moreover, this serves as a metric because the client can see the "as-is" and "to-be" by areas that have been automated.

7 Analyze

7.1 Interviews

The qualitative analysis of the interviewees' experience shows ways to work the diversity of roles and expertise to reach customers' needs in terms of software quality, with Design Thinking and Scrum combination.

Engaging teams and adjusting the mindset of companies to the desires of this new consumer, these methodologies bring to light the importance of collaboration at work for innovation: the risks are smaller, and there are significant benefits.

Design Thinking is focused on the client's mind making possible the gap of errors becomes much smaller. When creating and developing new products or services, the strategy assumes a greater tangibility of ideas and concepts, which can be interesting both for the corporation and for the end user.

User-centered evaluation is conducted by various IT professionals and has the potential to impact software development practice as Lárusdóttir, Cajander & Gulliksen (2013) lists: error rates, surveys, interviews, observing users, feedback, peer review and so on.

The interviews were conducted by questions faced to the domain of knowledge intending to explain how to correct quality problems in software. Much of this learning came through correcting defects caused by their mistaken understanding of the business domain.

How Design Thinking could be joined together to Scrum and improve software quality were showed off when we asked about the variety of outcomes. Disclosure of incorrect assumptions, just like the successful experience, were part of the experience.

The realization was that we should investigate more about user participation, scope changes, sprints to deliver a release, software quality improvement metric, other

tools, platforms, and practices to clarify the concept of how to provide a professional environment for software development.

7.2 Survey

The scope in a Scrum software project, as an initial assumption, is validated as early as possible with customers. Assembled based on several cycles, with fragmented closures and deliveries, produced by the team involved, serve as a basis for the process to be completed much more clearly.

Therefore, it is always important to know well the different kinds of tools, platforms, and practices with their concepts, functionalities and the role of the team involved in the project, ensuring more transparency in deliveries, and reducing risks.

A great variety of tools, platforms, and complementary practices were mentioned by the interviewees to conduct the software development and how they can be applied together, much more assertively to prevent from bugs, usability problems, performance, and some other considered as the most common software problems.

DMAIC model was taken as a premise for this thesis to specify the root cause of defects. Specially Analyze phase was recognized by the practitioners as an intuitive process to identify variation sources, the data to investigate and verify cause-and-effect relationships from software problems. The majority indicated it is also necessary to work with continuous deliveries, avoiding that problems should be found only at the end of the process.

Problems adjustment is viable if the tests are made in the client environment, with Scrum ceremonies and documentation. Dobrigkeit, Wilson & Nicolai (2018) summarizes that during the sprint, daily Scrum meetings ensure everybody knows what is going on. At the end of each sprint a review meeting is held to inspect and review the developed software. Furthermore, a retrospective meeting is held to reflect on the process and teamwork and discuss required changes for the next sprint.

The challenge is to solve problems in the initiate phase of Scrum software project when the user stories are merged as Epics to give rise to a higher development load or even represent the idea of the project as a whole. Mindset, inferences, taken decisions, sprints with no tangible tasks, user's inquiry not well conducted were identified as issues to be addressed by the Design Thinking.

Avoid those problems were indicated as possible with the user participation since the product discovery to create the Epics of the backlog with interviews till prototyping and tests validation. This cycle is represented by the sprints to deliver a release in two to four weeks, on average.

A bunch of tiny changes can make a real impact on overall project success. For dealing with scope changing, Scrum methodology can manage it correctly if the team clearly understand the change. Also, fast delivery with quality and continuous adaptation is the result of the association of Design Thinking and Scrum.

Journey Map and Personas are the most common tools for Design Thinking and seen as a time-consuming process. The team who conduct the practices must be proficient in the tools to facilitate the understanding of the methodologies and avoid losing perspective.

Transforming insights into design question to the interviewees, among others, means to raise user needs, with empathy, collaboration, and co-creation; working together with the user during all the process.

In Scrum, the activities to be developed, their priorities and deadlines, are recorded (product backlog) and related from sprints - which represent the time frame in which a task (user story) will be developed and delivered.

Developers, designers, Product Owner, quality analysts, and Scrum Master were mentioned as the essential profiles from each Scrum team. For the team, Vlaanderen *et al.* (2011) affirm that Vision is the starting point for the lifecycle of

most requirements and will generally move through a set of stages, during which is refined with details and specifications.

Each company adjusts its tasks in a block of activities, which can take from two to four weeks, breaking the project into smaller sprints. All the process is usually controlled in a framework, where the team can see the tasks that are under development, those that have been worked on but still need to be checked or tested, and those that are considered completed.

From the point-of-view of the professionals interviewed, different kind of metrics are used to estimate the improvement of software quality perceived by the customer: *software improvement, number of bugs, storypoints, NPS, feedback, histories accepted by the clients, KPI, software working producing expected results.*

The 13th annual State of Agile Report (2019) stated that the companies' objective for adopting Agile were more about to enhance software quality. When organizations were questioned of how success would be measured with Agile initiatives the result indicates *Customer/User satisfaction* as the main indicator of success.

The software was named as adequate to user needs and pains by three essential ways: *interviews with users, MVP validated, software released to the production environment and sprints backlogs completed.* As mentioned by Abrahamsson *et al.* (2002) the Sprint Backlog also includes the tasks of setting up the team and Scrum roles and building management practices in addition to the actual tasks of implementing the demo.

Once the team has developed a few prototypes, successfully tested them and thereby gained a profound understanding of what the perfect final solution should look like, the next step would be to build an MVP. Additionally, Vetterli *et al.* (2013) concluded that as prototype maturity increases, teams will also start to create proof-of-concept implementations for resulting technical challenges. Given the respective

elements of the architecture, it also becomes clearer, which necessary (due to legal reasons) and meaningful (due to company policy) standards are applicable to the project, continuously adding functionalities in new releases until to have fully developed the 'final' product, as envisioned with initial prototypes. Ratifying Vetterli *et al.* (2013), our single case study reflects this reasoning.

A cross-functional team, with different job title and roles, act from the product discovery and development until software validation. Majchrzak, More & Faraj (2012) observed how creative breakthroughs occur without creative tensions between individuals and how the knowledge transformation occurs between different perspectives.

From Chapter 6 the practitioners indicated that Design Thinking and Scrum emerge as a strategy in business projects to increase team productivity and develop valuable, customer-centric deliverables.

7.3 Case Study

The JMJ case study has been established as an example to show how strategy emerges in business projects to increase staff productivity and deliver valuable, customer-centric backlogs.

By engaging teams more and adjusting companies' mindset to the needs of this new consumer, these methodologies bring to light the relevance of collaborative behavior at work.

The insights produced using Design Thinking help to a better scope definition, that is which direction to go and develop the project. From there, the characters involved with Scrum complement the work to be done by building on multiple cycles - with fragmented closures and deliveries - and ensuring the product has value from the very beginning.

By working together, Design Thinking and Scrum ensured flexibility, adaptability, scalability, quality, productivity and improved communication. This facilitates and empowers all projects.

JMJ assists and give a metric when describing and improving the organization's internal processes using AS IS and TO BE Process Mapping as a management tool.

In the AS IS survey, the current process is modeled with key users. The TO BE scenario present the opportunity to make decisions more efficiently and standardize process, increase productivity, improve product and or service delivery quality, and achieve greater customer satisfaction.

In addition, the MJM's team can contribute to the optimization of processes to better adhere to practices, organizational objectives and support systems.

7.4 Discussion on Related Work

As a comparative model to our study we can propose the "Enterprise Design Thinking IBM Framework", created to solve the user's problems related to software development at the speed and scale of the modern enterprise.

The principles are described to focus on user outcomes, restless reinvention, and diverse empowered teams in an Agile environment. The loop that drives the process has the intention to understand the present and envision the future in a continuous cycle of discovering.

The keys to align the team are facilitated by Hills created to express the objectives which need to be achieved, and Playbacks to align by regularly exchanging feedback. The Sponsors Users monitor the development process with the responsibility to stay in touch with users' real-world needs throughout the project.

After the Visioning Phase, in a product team, Hills are owned by Offering Management – a more customer-centric specialist - and defined in collaboration with

Design and Engineering. Around each Hill are formed diverse empowered teams with the expertise and authority needed to deliver their outcome independently.

The real-world users are named Sponsor Users who are representative of the target user, personally invested in the outcome and available to collaborate. They are recruited after the team write the Hills and have a sense of the target users. The Sponsor Users are involved in Delivery Wave phase working together in the Playbacks as checkpoints to review the state of the project and plan the next steps.

At the end of each sprint, on Delivery Playback meeting, the team decides whether to release the project to real users to identify significant user experience gaps they need to prioritize.

Value proposition and tools to measure success as we propose in our thesis were not part of Lucena et al. (2017) analysis of IBM's Framework. That was based on a survey to know how teams used Design Thinking for user experience improvement in software development, and the result was the improvement in terms of a productivity boost to time and resources savings.

Another difference from our research is the fact that in the Visioning Phase, that is the product discovery stage, happens without the participation of users.

8 Share

During our research, a paper was submitted to a journal in order to share and communicate our findings:

- International Journal of Agile Systems and Management (Rank Q1 - Scimago).

The dissertation final report will be subject of discussion and evaluation with a qualified jury.

9 Conclusion

Analyze quality in software production has many benefits with short iterations during scope initiate phase of Scrum combined with Design Thinking methodology. For this attempt, "How" and "Why" questions are more explanatory and likely to lead to the use of case studies, histories, and experiments as the preferred research strategies. This is because such questions deal with operational links needing to be traced over time, rather than mere frequencies or incidence (Yin, 2003).

Design Thinking and Scrum frameworks can be united to create and develop software with quality. The strategy presupposes a greater tangibilization of ideas and concepts guiding the company to make it viable, technologically, and economically, which can be interesting both for the corporation and for the end user.

Design Thinking and Scrum together ensure flexibility, adaptability, scalability, quality, productivity, and improved communication. This association facilitates and empowers all software projects, to provide more transparency with continuous customer involvement in deliveries and minor risks of failure.

As the same time, the members of the team learn from each other how to improve releases with Epics and user stories since the definition of scope in the initiate phase of a Scrum project, defining the degree to which a customer or user perceives that software meets their composite expectations.

Associate Design Thinking and Scrum with a great variety of tools, platforms, and practices are a prove of how those frameworks are adaptable to meet the users' expectations while identifying the problems and to give solutions.

Value perception has a direct relationship with customer satisfaction, and different kind of metrics were mentioned to be used to estimate the improvement of software quality.

As random variables (not under control), such as waste of resources, a misunderstanding due to miscommunication, the absence of customer feedback, brute-forced solutions, perfect product for the wrong problem, team frustration, law of instrument bias, are some of the limitations of the research that were not encompassed by our investigation.

We also limited our interviews and survey target audience to Brazilian IT practitioners. The positive aspect is the fact that a relevant part of audience were women from product management of all stages of software lifecycle.

A single case study is a limitation of our study and an opportunity not to generalize but for the disclosure of valuable insights.

For the future work, will be possible to show software development as value-creating allowing calculate clear ROI (Return of Investment) from the capital expenditure, turning customer experience from a cost center to a profit center, aligned with organization's strategy.

References

- Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). Agile software development methods: review and analysis. VTT Technical Research Centre of Finland. VTT Publications, 478.
- Baxter, P., & Jack, S. (2008). Qualitative Case Study Methodology: Study Design and Implementation for Novice Researchers. *The Qualitative Report*, 13(4), 544-559.
- Benbasat, I., Goldstein, D.K., & Meead, M. (1987). The Case Research Strategy in Studies of Information Systems. *MIS Quarterly*, Vol 11(3) pp. 369–386.
- Cardinal, Mario. (2014). Executable specifications with Scrum. First Edition. Massachusetts, Pearson.
- Chasanidou, D., Gasparini, A. A., & Lee, E. (2015). Design Thinking Methods and Tools for Innovation. *Lecture Notes in Computer Science*, 12–23. Springer International Publishing Switzerland.
- Cohn, M. (2010). *Software Development Using Scrum*. Addison-Wesley.
- Deemer, P., Benefield, G., Larman, C., & Vodde, B. (2012). *The Scrum Primer. A Lightweight Guide of the Theory and Practice of Scrum*. InfoQ.
- Denning, P.J. (2016). The Profession of IT Software Quality. *Communication of the ACM*, vol 59, no. 9.
- Dhir, S., Kumar, D., & Singh, V. B. (2019). Success and Failure Factors that Impact on Project Implementation Using Agile Software Development Methodology. *Software Engineering* (pp. 647-654). Springer Nature Singapore Pte. Ltd.
- Diebold, P., Ostberg, J.P., Wagner, S., & Zender, U. (2015). What Do Practitioners Vary in Using Scrum? *Agile Processes in Software Engineering and Extreme Programming*, 40–51, Springer International Publishing Switzerland.

- Dobrigkeit, F., Wilson, M., & Nicolai, C. (2018). Adding Scrum-style project management to an advanced Design Thinking class. Hasso-Plattner Institute. Nord Design Society.
- Fenton, N., & Bieman, J. (2015). Software Metrics. A Rigorous and Practical Approach. CRC Press. Taylor & Francis Group, Chapter 4.
- Fleming, I. (2016). Software Quality Assurance, Morgan Kaufmann, 47-61.
- Gannon, M. (2013). An Agile Implementation of SCRUM. IEEE Software.
- Grashiller, M., Luedeke, T., & Vielhaber, M. (2017). Integrated approach to the agile development with design thinking in an industrial environment. Proceedings of the 21st International Conference on Engineering Design (ICED17), Vol. 2: Design Processes | Design Organization and Management, Vancouver, Canada.
- Häger, F., Kowark, T., Krüger, T., Vetterli, C., Übernickel, F., & Uflacker, M. (2015). DT@Scrum: Integrating Design Thinking with Software Development Processes. In H. Plattner, C. Meinel, & L. Leifner (Eds.), Design Thinking Research (pp. 263-289). Basel, Switzerland: Springer International.
- Harrison, H., Birks, M., Franklin, R., & Mills, J. (2017). Case Study Research: Foundations and Methodological Orientations. Forum Qualitative Sozialforschung / Forum: Qualitative Social Research, 18(1), Art. 19.
- Hokkanen, L. (2017). From Minimum Viable to Maximum Lovable: Developing a User Experience Strategy Model for Software Startups. Tampere University of Technology. Publication 1483.
- Jones, A., & Thomas, V. (2019). Determinants for Successful Agile Collaboration between UX Designers and Software Developers in a Complex Organization. Taylor and Francis Group.
- Jones, C., & Bonsignour, O. (2011). Defining Software Quality and Economic Value. In O. Bonsignour & C. Jones (Eds.), The Economics of software value (pp. 1-33). Boston, MA: Addison-Wesley Professional.

- Karout, R., & Awashti, A. (2017). Improving software quality using Six Sigma DMAIC - bases approach: a case study. *Business Process Management Journal*, 23(4), 842-846.
- Khan, A.S., & Kajko-Mattsson, M. (2012). Core Handover Problems. *ACM New York*
- Krefting, L. (1991). Rigor in qualitative research: The assessment of trustworthiness. *American Journal of Occupational Therapy*, 45, 214-222.
- Lárusdóttir, M., Cajander, Å., & Gulliksen, J. (2013). Informal feedback rather than performance measurements – user-centred evaluation in Scrum projects. *Behaviour & Information Technology*, 33(11), 1118–1135. Taylor and Francis Group.
- Lee, A. S. (1991). Integrating Positivist and Interpretive Approaches to Organizational Research. *Organization Science*, (2), pp. 342-365.
- Lietz, P. (2010). Research into questionnaire design – A summary of the literature. *International Journal of Market Research*, Vol. 52 (2).
- Lindberg, T., Köppen, E., Rauth, I., & Meinel, C. (2012). On the Perception, Adoption and Implementation of Design Thinking in the IT Industry. *Design Thinking Research* (pp. 229-240). Springer-Verlag Berlin Heidelberg.
- Lindberg, T., Meinel, C., & Wagner, R. (2011). Design Thinking: A Fruitful Concept for IT Development? *Design Thinking: Understand – Improve – Apply, Understanding Innovation* (pp. 3-18). Springer-Verlag Berlin Heidelberg.
- Lucena, P., Braz, A., Chicoria, A., & Tizzei, L. (2017). IBM Design Thinking Software Development Framework. *Communications in Computer and Information Science*, 98–109.
- Madill, A. (2011). Interaction in the Semi-Structured Interview: A Comparative Analysis of the Use of and Response to Indirect Complaints. Taylor & Francis Group, LLC.
- Majchrzak, A., More, P. H. B., & Faraj, S. (2012). Transcending Knowledge Differences in Cross-Functional Teams. *Organization Science*, 23(4), 951–970.

- Moe, N. B., Dingsøyr, T., & Dybå, T. (2009). A teamwork model for understanding an agile team: A case study of a Scrum project. *Information and Software Technology*, 52(5), 480–491, Elsevier.
- Pinheiro, E.G., Conte, T.U., Lopes, L.A., & Zaina, L.A.M. (2018). The contribution of non-technical stakeholders on the specification of UX requirements: an experimental study using the proto-persona technique. ACM New York.
- PMI and Agile Alliance (2017) - A Guide to the Project Management Body of Knowledge (PMBOK® Guide), Project Management Institute, Sixth Edition.
- Razavian, M., Tang, A., Capilla, R., & Lago, P. (2016). In two minds: how reflections influence software design thinking. *Journal of Software: Evolution and Process*, 28(6), 394–426.
- Rigby, D. K., Sutherland, J., & Takeuchi, H. (2018). Embracing Agile. [online] Harvard Business Review, <https://hbr.org/2016/05/embracing-agile> [Accessed on 21 Jul. 2019].
- Runeson, P., & Höst, M. (2008). Guidelines for conducting and reporting case study research in software engineering. Springer.
- Stopher, P. (2012). Collecting, managing, and assessing data using sample surveys. Cambridge University Press, pp. 177-198.
- Sutherland, J., & Schwaber, K. (2017). The Scrum Guide - The Definitive Guide to Scrum: The Rules of the Game. Scrum.org.
- Tarkkanen, K., Harkke, V., & Reijonen, P. (2015). Are We Testing Utility? Analysis of Usability Problem Types. DUXU 2015: Design, User Experience, and Usability: Design Discourse (pp. 269–280). Springer International Publishing Switzerland.
- Vetterli, Christophe; Uebernicket, Falk; Brenner, Walter; Haeger, Franziska; Kowark, Thomas; Krueger, Jens; Mueller, Juergen; Plattner, Hasso; Stortz, Barbara & Sikkha, Vishal. (2013). Jumpstarting Scrum with Design Thinking. University of St.Gallen.

- Vlaanderen, K., Jansen, S., Brinkkemper, S., & Jaspers, E. (2011). The agile requirements refinery: Applying SCRUM principles to software product management. *Information and Software Technology*, 53(1), 58–70.
- Wallach, D., & Scholz, S. C. (2012). *User-Centered Design: Why and How to Put Users First in Software Development*. Software for People, 11–38. Springer-Verlag Berlin Heidelberg.
- Yin, R.K. (2003). *Case study research. Design and methods*, 3rd edn. London, Sage Publications.
- Yoshida, T. (2018). Try Design Thinking + Scrum: A Powerful Hybrid Agile Approach | Lifecycle. [online] Lifecycle, <https://lifecycle.management/2018/09/18/design-thinking-plus-scrum/> [Accessed on 21 Jul. 2019].
- Zaman, S., Adams, B., & Hassan, A.E. (2012). *A Qualitative Study on Performance Bugs*. IEEE Press Piscataway.
- 13th Annual State of Agile Report. (2019) CollabNet, Inc., <https://www.stateofagile.com/#ufh-c-473508-state-of-agile-report> [Accessed on 21 Jul. 2019].

Appendix I Interview Guide

1. Quais são os problemas de qualidade em softwares mais recorrentes? Além disso, como costumam corrigi-los?
2. A empresa usa Scrum com Design Thinking no desenvolvimento de software? Comente a experiência, por favor.
3. Se você reconhecer problemas com o uso do Design Thinking na fase inicial do Scrum, por favor, nomeie-os.
4. Usam outras ferramentas, plataformas e práticas associadas? Poderia mencionar quais?
5. O cliente e a equipe trabalham juntos para melhorar a qualidade do software? Como funciona?
6. Se houver mudanças de escopo durante o projeto Scrum, quais práticas adaptativas são aplicadas?
7. Quantas *sprints* são necessárias para entregar uma *release*, em média?
8. Qual tipo de métrica é usada para estimar a melhoria da qualidade do software percebida pelo cliente?

Appendix II Survey

Design Thinking e Scrum - qualidade de software

Este questionário é parte da Tese "Melhorando a Qualidade do Software usando o Design Thinking com o Scrum", cujo objetivo é comprovar que as técnicas associadas revertem em mais valor na solução criada para o cliente.

Agradeço pela colaboração.

1. Qual seu cargo/habilidade? *

Texto de resposta curta

2. Quais problemas de qualidade de softwares mais citados pelos clientes? *

- bugs
- usabilidade
- falha em requisitos de projeto
- ferramenta/tecnologia
- código inacessível/fechado
- performance
- documentação
- Outros...

3. As equipes (times ou squads) são compostas por quais perfis de profissionais? *

Texto de resposta longa

4. Quais etapas da solução do problema são parte do processo de melhoria: *

- Definir
- Medir
- Analisar
- Aperfeiçoar
- Monitorar/Controlar

Design Thinking

Descrição (opcional)

5. A empresa usa o Design Thinking no desenvolvimento de software? *

- Sim
- Não. Vá para a pergunta 11

6. Em que fase(s) do projeto?

Texto de resposta longa

7. Selecione as práticas de DT mais utilizadas para elaborar a hipótese de solução?

- epics
- blueprints
- mapa de jornada
- double diamond
- how might we
- crazy 8
- personas
- Outros...

8. Quais são as desvantagens do uso do Design Thinking?

Texto de resposta longa

9. Comente, por favor, a frase: "O erro na primeira versão gera ajustes incrementais no produto para criar um software de valor agregado. Os Epics são refinados e depois priorizados para criar um Backlog do Produto para o projeto."

Texto de resposta longa

10. Complete, por favor: Transformar insights em questões de Design envolve

Texto de resposta longa

Scrum

Descrição (opcional)

11. Sua equipe adota o Scrum como metodologia ágil? *

- Sim
- Não. Vá para a pergunta 17

12. Como pode contribuir para a melhoria do software associar o Scrum com o Design Thinking?

Texto de resposta longa

13. A equipe responde rápido a mudanças de escopo durante o projeto Scrum?

- Sim
- Não

14. Quantas sprints, em média, são necessárias para entregar uma release?

Texto de resposta curta

15. As etapas de Planning e Daily Meeting, e Retrospectiva ajudam a que todos colaborem para um mesmo objetivo?

- Sim
- Não

16. Esta afirmativa é verdadeira?: "O objetivo é que alguns dos trabalhos realizados em cada sprint resultem em recursos que os usuários possam ver e opinar."

- Sim
- Não

Demais práticas

Descrição (opcional)

17. Cite outras ferramentas, plataformas ou práticas que costumam ser usadas pelas equipes. ^{*}
Por favor, justifique as escolhas.

Texto de resposta longa

Clientes

Os utilizadores finais do software

18. Em que etapa(s) e como o cliente e a equipe trabalham juntos para melhorar a qualidade do software? ^{*}

Texto de resposta longa

19. Comente, por favor, a frase: "Capturar o sentido da qualidade desejada pelos clientes é o principal objetivo a ser atingido, no sentido de estabelecer a diferença entre percepções e expectativas, do começo ao fim do ciclo de vida do software." ^{*}

Texto de resposta longa

20. No final do processo, a equipe tem condições de afirmar se o software é adequado ou não à necessidade do cliente por meio de: ^{*}

- sprints backlogs
- lead time
- entrada de melhoria em produção
- entrevista com o cliente
- MVP validado
- Outros...

Fique à vontade para colocar comentários ao questionário:

Texto de resposta longa

Source: Elaborated by the author

Appendix III Case Study – Consent letter

Pedido de colaboração e consentimento

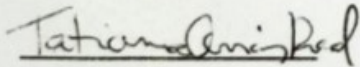
No âmbito da tese de Mestrado em Informação e Sistemas Empresariais, que está a ser realizada na Universidade Aberta (UAb), em parceria com o Instituto Superior Técnico (IST) da Universidade de Lisboa, pela aluna Tatianna Arrais Rosal, sob a orientação do Professor Doutor Miguel Mira da Silva, com o tema "Melhorando a Qualidade do Software Usando o Design Thinking com o Scrum", venho por este meio solicitar a vossa colaboração.

Este projeto tem por objetivo comprovar que as técnicas do Design Thinking e do Scrum associadas revertem em mais valor na solução criada para o cliente.

Caso a presente proposta venha a ser aceite pela JMJ Sistemas e Consultoria, será oportunamente enviado um questionário com as questões modelo a serem analisadas no âmbito do tema da tese. De seguida, uma entrevista e solicitação de evidências comporão o estudo de caso.

Com o conhecimento e autorização prévia da vossa empresa os resultados dos dados recolhidos serão analisados e, mais tarde, apresentados para vossa análise e reflexão. Somente com a aprovação da JMJ Sistemas e Consultoria a síntese das informações servirão como base para a presente investigação.

Com os melhores cumprimentos,

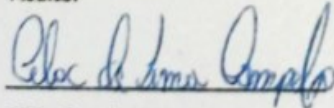


Tatianna Arrais Rosal

Aluna n.º 1701391 da Universidade Aberta

Rio de Janeiro, Brasil, 05 de maio de 2019.

Aceite:



CEO Alex de Lima Campelo

JMJ Sistemas e Consultoria

Mato Grosso, Brasil, 05 de maio de 2019.

