

Article

# Design Thinking: Challenges for Software Requirements Elicitation

Hugo Ferreira Martins <sup>1</sup>, Antônio Carvalho de Oliveira Junior <sup>2</sup>, Edna Dias Canedo <sup>1,\*</sup> ,  
Ricardo Ajax Dias Kosloski <sup>2</sup>, Roberto Ávila Paldês  <sup>1</sup> and Edgard Costa Oliveira <sup>3</sup> 

<sup>1</sup> Department of Computer Science, University of Brasília (UnB), P.O. Box 4466, Brasília-DF, CEP 70910-900, Brazil; hugomartins013@gmail.com (H.F.M.); roberto.paldes@gmail.com (R.Á.P.);

<sup>2</sup> Faculty UnB Gama, University of Brasília (UnB), P.O. Box 4466, Brasília-DF, CEP 70910-900, Brazil; antoniojk7@gmail.com (A.C.d.O.J.); rikosdf@gmail.com (R.A.D.K.)

<sup>3</sup> Department of Production Engineering, University of Brasília (UnB), P.O. Box 4466, Brasília-DF, CEP 70910-900, Brazil; ecosta@unb.br

\* Correspondence: ednacanedo@unb.br; Tel.: +55-61-98114-0478

Received: 21 October 2019; Accepted: 30 October 2019; Published: 28 November 2019



**Abstract:** Agile methods fit well for software development teams in the requirements elicitation activities. It has brought challenges to organizations in adopting the existing traditional methods, as well as new ones. Design Thinking has been used as a requirements elicitation technique and immersion in the process areas, which brings the client closer to the software project team and enables the creation of better projects. With the use of data triangulation, this paper brings a literature review that collected the challenges in software requirements elicitation in agile methodologies and the use of Design Thinking. The result gave way to a case study in a Brazilian public organization project, via user workshop questionnaire with 20 items, applied during the study, in order to identify the practice of Design Thinking in this context. We propose here an overview of 13 studied challenges, from which eight presented strong evidence of contribution (stakeholders involvement, requirements definition and validation, schedule, planning, requirement details and prioritization, and interdependence), three presented partial evidence of contribution and two were not eligible for conclusions (non-functional requirements, use of artifacts, and change of requirements). The main output of this work is to present an analysis of the use of Design Thinking to see if it fits properly to be used as a means of solving the challenges of elicitation of software requirements when using agile methods.

**Keywords:** agile methodology; requirements elicitation; design thinking; software evaluation

## 1. Introduction

Proper requirements elicitation in a software project is considered one of the most important and difficult tasks during the software process. Shortcomings in the treatment of requirements has been identified as the main cause of failure of software projects [1,2]. The negative effects of a misconducted software requirements elicitation are well known: project delays, cancellations and deliveries of incomplete work due to the bureaucratization of the development of software established in the mid-2000s. Aware of this, a group of influencers of the Extreme Programming community gathered and discussed these problems. The agile methodology has been popular since 2001 due to the valorization of teams members and the intense interaction among them, constant software deliveries, more effective stakeholder collaboration and an openness to receive changes from the users [3].

The increasing stakeholders involvement in the project has helped obtain the requirements that satisfy the project needs. However, in practice, the relationship among stakeholders is more or less

unrealistic [4]. A systematic review of 21 studies showed that software requirements engineering practices helped to solve some challenges in the traditional requirements engineering. Nevertheless, it also introduced a number of constraints that showed limitations in the achievement of the appropriate balance between agility and the stability pursued by agile methodologies [5].

An empirical study based on the analysis of the data collected in 16 software development organizations in the U.S.A revealed seven challenges in the agile methodologies: (1) problems with cost estimation and schedule; (2) inadequate or inappropriate architecture; (3) neglecting non-functional requirements; (4) stakeholder access and participation; (5) prioritization in a single dimension, compromising scalability; (6) inadequate verification of requirements; and (7) minimal documentation [4]. Among these seven challenges, some are directly related to the lack of stakeholder involvement in the software requirements elicitation process, namely Items 1, 3, and 4.

Out of these seven challenges, three are directly related to the lack of stakeholder involvement in the software requirements elicitation process, namely Items 1, 3, and 4. This problem confirms the idea defended by Pressman [6]: he explained the importance of customer involvement in the specification, towards the success of any project. Linked to this and taking into account the current high competitiveness, the goal of the software industry is to increase productivity and quality of the end-products for its customers, thus more and more software has been experimented to develop different techniques and to elicit requirements in order to bypass this bottleneck.

Design Thinking (DT) emerges as a methodology that provides design-inspired practices for resolving and developing projects, using empathy, encouragement of creativity and rationality to meet the needs of users and converge into innovative solutions [7]. The mission of Design Thinking is to translate observations into insights and to generate products and services with the objective of improving people's life quality. It is based on how the user would like to be treated: with empathy, encouragement of collaboration and participation in the application of generated results. DT-related literature considers that when initiating a project the most important task is to understand and to question what people's needs are and to understand what impacts the product may cause. Software project teams involved delves into the user's universe in order to extract all the information [7]. Thus, DT methodology seems to be a mechanism that can help requirements elicitation in agile methodology-based software projects [8].

Adikari et al. [9] defined Design Thinking as: (1) an approach to creating new or better solutions based on customer needs that adds value to business; and (2) a design approach. The Design Thinking process is basically divided into three activities: (i) identification of the real problem (immersion) that an organization/customer faces; (ii) ideation run-through, which is the moment when ideas and concepts are generated; and (iii) prototyping, in order to generate innovations about problems identified in the initial phase [7,10–12].

Due to the importance of the requirement's elicitation, and with the aim of achieving success in the software projects, this paper intends to respond the following research questions: What are the Design Thinking techniques used? what are the challenges faced in the requirements elicitation of agile software development projects?

The scientific methodology adopted in this research helped to understand how design thinking can contribute to the requirements elicitation in agile software development. We also classify our research method as descriptive and focused on the subjective character of the question under analysis and its quantitative changes. To gather information, we used methods such as bibliographical research, based on the Systematic Literature Review (SLR) method, to answer research question [13]. In addition, we conducted a case study, a questionnaire application and interviews with the software requirements elicitation teams involved in the project, in order to answer RQ.2 and RQ.3. Such methods were used together according Trivinos approach [14] as means to making a triangulation of qualitative data.

The main contribution of this work is to present an analysis of the use of Design Thinking to verify if it fits properly to be used as a means of solving the challenges of software requirements elicitation

when using the agile methodology, as well as new profiles of users demanding an increased level of value in their acquired products.

## 2. Background

Agile methodologies emerged as an innovative proposal so that companies could deliver products expected by users on time and efficiently for the development team, due to the fact that software development processes should be extremely organized, wasting only a minimum of resources [14]. Agile methodologies for software development have emerged as an answer to the challenges faced by traditional methodologies in the document-oriented software development [3,15]. Even though agile methodologies have broken the paradigm of cascade development and other models, they do not replace the processes that already exist, but present an alternative to existing paradigms [16].

One of the main objectives of agile software development is to develop software more quickly by means of a series of iterations (short periods of time) that are feasible in relation to cost and time. Each iteration produces a version of the software for the client/users in a manner that ensures that the defined requirements were implemented [9]. According to Fadel and Teixeira [14], in agile methodologies, at each iteration, a sub-project (components, functionalities or modules of the software) is created, and all steps of the software development process are realized: planning, requirements, coding and testing. The iterations last for a few weeks, which leads to quick results, mainly for users. The stakeholder and part of the project development team make suggestions and improvements, participate in the scope planning of each iteration and approval of each delivery. In addition, agile development teams are generally small, which facilitates communication.

### 2.1. Requirements Elicitation

Software requirements reflect needs expressed by users and written in sentences that impose conditions to software quality. They describe which services, constraints and features a system must provide, must be compliant and represent, and also to specify the knowledge necessary for its development [17]. They are obtained by means of a systematic process that involves five activities: (1) elicitation; (2) analysis and negotiation; (3) documentation; (4) verification and validation; and (5) management. This process is known as Requirements Engineering (RE) [18,19]. A complete understanding of software requirements is fundamental to a software project, no matter how the project will be done. A poorly analyzed and specified software will disappoint the user and will cause annoyance to the developer [6].

In the requirements elicitation activity, teams work with stakeholders to learn more about the domain of the application, system services to be provided, the constraints of the operation and the required performance of the system (non-functional requirements) [20]. During agile software development, requirements elicitation demands an interactive face-to-face communication with users. Agile methodology recognizes that requirements change constantly, evolve over time and are discovered throughout the software development process [4].

In the context of agile methodologies, requirements engineering is performed iteratively throughout the development process instead of a single phase at the beginning of the project. Due to this iterative cycle, a just-in-time model is often used to refine the high-level requirements and derive them into low-level tasks that can be implemented by developers [21]. There are several techniques for elicitation by using agile methods: (1) interviews; (2) brainstorming; (3) observation; and (4) case analysis [20,22,23].

### 2.2. Design Thinking

Design Thinking (DT) is an approach to creating new solutions based on the needs of users and that adds value to the business, considered an approach towards conception [9]. The main idea of Design Thinking is to see how designers progress during the design process, with creative ideas and solutions designed to discover new opportunities [9]. DT is user-centered innovation which

requires collaboration, interaction and practical approaches to find the most appropriate ideas and, consequently, coherent final solutions [11].

The Design Thinking process is basically defined by three phases: immersion; ideation; and prototyping [7,10–12,24].

1. **Immersion:** It can be divided into two stages: preliminary and in-depth. The preliminary immersion's objective is the re-framing and initial understanding of the problem. It aims to define the purpose of the design and its limitations, in addition to identifying the user and other key authors who should be considered. At this stage, it is also possible to raise issues which can be explored in the in-depth immersion. The stage of the in-depth immersion starts with the elaboration of a research plan with the protocol of the first research and the list of user's profiles and key authors who assist in the mapping of the investigated context. The communication with these users can take place with the help of certain techniques, such as interviews, generative sessions and awareness raising. These techniques can help understand the context related to the product and/or the service explored during the project [25].
2. **Ideation:** The intention here is to generate new ideas for the project with the use of tools and people to stimulate creativity and generate solutions within the context of the project. Generally, the ideation phase begins with a brainstorming around the subject in order to collect ideas, which are discussed, documented and validated constantly in meetings with the client during the prototyping phase [25].
3. **Prototyping:** It has the function of assisting in the validation of ideas. Even though presented as the ultimate stage of the DT process, it can be carried out in parallel with the immersion and ideation phases throughout the project [25]. The main outcome of this process is to learn about strengths and weaknesses of the idea as well as identifying new directions for the prototype, a reverse way of the traditional creative thinking, with the aim to visualize and imagine new alternatives and solutions. After the solutions are well defined and inspired by the user's needs, which is in fact the focus of the whole analysis process, the solution will then be implemented [11].

It is also important to emphasize that DT is compliant with the initial elicitation practices of requirements engineering, and that the use of rapid prototyping and customer involvement is consistent with the agile development methods. This provides a consistent methodology for both documentation, and team management, which is one of the main focuses of the agile development methods [26]. Activities such as brainstorming with multidisciplinary teams is an example that can be used to elicit the best ideas in order to do a team's evaluation. The ideas that are approved take shape with the rapid prototyping to generate data that will be useful for the continuity of ideas [11].

### 2.3. Related Works

Integrating Design Thinking into the Agile system development methodology means customers take part in every sprint, and more focus is put in the beginning to determine the customer needs, requirements, and environment [27]. Design Thinking would enable clearer focus of customers requirements—affecting the product vision and product backlog. There is less rework during the sprints, since there is a clearer vision of requirements and customer expectations at the start [27].

Schon et al. [21] presented a review with the aim of capturing the current state of the art literature related to agile requirements, focusing on stakeholder's involvement. They investigated which methodologies are often used that present the user's perspective on managing project requirements.

Higuchi et al. [12] developed a project management model that covers the whole process of digital game development and proposes Design Thinking approaches combined with agile methodology. The model has three phases: (i) project definition; (ii) the connection; and (iii) the scrum.

Inayat et al. [5] presented practices and challenges encountered during the requirements elicitation phase when using agile methodologies. Adikari et al. [9] developed a framework based on the Design Thinking methodology in order to support the activities of an agile software project in a real context.

Hehn and Uebernickel [28] conducted an exploratory case study to identify how DT and Requirements Engineering can work together in an agile development environment, covering since the conception until its implementation. These authors defined seven requirements for engineering challenges in agile methodologies: minimal documentation, customer or user issues, non-functional requirements neglected or improper architecture, knowledge of tacit requirements, inaccurate effort estimates, and difficulty in prioritizing requirements. The findings of their case study suggest that DT has the potential to support RE practices and vice versa.

Correa et al. [29] presented a case study report aiming to verify if the use of DT and its techniques allow a proper identification of a software solution understanding. It used the double diamond model and the Hasso Plattner model [30] with a set of 13 DT techniques and, based on the opinion of the stakeholders and other participants, the understanding was adequate and the proposed solution near the stakeholders, and suggests that the techniques are appropriate to support software development.

Carroll and Richardson [31] showed the importance of using Design Thinking to perform requirements elicitation process within software engineering and presented a case study that analyzed how Design Thinking can assist in requesting software requirements for an application in the context of health systems. Carroll and Richardson [31] claimed that Design Thinking complements traditional requirements engineering techniques in prototype-oriented requirements analysis by encouraging the generation of innovative results through the creative and exploratory interaction of people, processes, technology and business practices. In addition, Carroll and Richardson [31] provided an overview of Design Thinking phases (empathy, definition, idea, prototype, and testing) that promote a learning lifecycle that helps in understanding health issues and the software design process.

Corral and Fronza [32] discussed a methodology strategy that integrates Design Thinking into the software development process, in order to meet undergraduate software engineering principles by comparing experience in software engineering courses in two educational environments, one that meets the principles of agile and one that meets the principles of Design Thinking. In addition, they also compared the experience of teaching software engineering to educational environments with a similar work environment. Corral and Fronza [32] based their studies on the fact that software is a product that has a mission to meet user requirements and, therefore, in the software process, analysis–design–implementation–test can be reinterpreted as empathy–define–idea–prototype–test.

### 3. Methodology

In this study, we used data triangulation technique. According to Denzin [33], methodological triangulation involves multiple qualitative and/or quantitative methods for investigating a certain issue, while data triangulation uses dissimilar sources of data or different data from the same source to examine the same object. Triangulation is a process in which several methods are used in the study of one phenomenon, and might be used in four basic ways: (1) data triangulation; (2) researcher triangulation; (3) theory triangulation; and (4) methodological triangulation.

According to Trivinos [34], data triangulation technique is presented in three different aspects: (1) subject-centered product processes; (2) elements produced by the subject environment; and (3) processes and products originated by the socioeconomic and social structure of the social macro-organism of the subject [34]. Figure 1 shows the aspects of Data Triangulation we used.

In this research, data analysis was made by triangulation based on four sources: (i) bibliographic research; (ii) observation; (iii) questionnaire; and (iv) interview. They are contained in the first aspect (process and products centered on research subject). This aspect emphasizes the products prepared by the research, ascertaining the perceptions of the subject through interviews and questionnaires, and the behavior and actions of the subject by means of observations (free or directed) of the behavior and the processes and products built by the subject (autobiographies, daily intimate, confessions, etc). Thus, the main data source was bibliographical research, observation, questionnaire and interviews which work together in a linear way, namely the outcome of one is an input for the others [34], as shown in Figure 2.

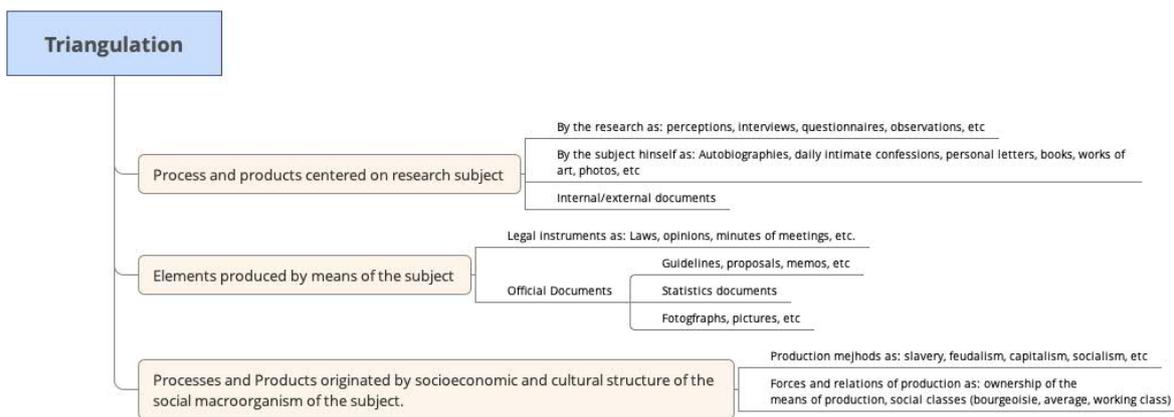


Figure 1. Aspects of Trivinos’ Data Triangulation Method, adapted from [34].

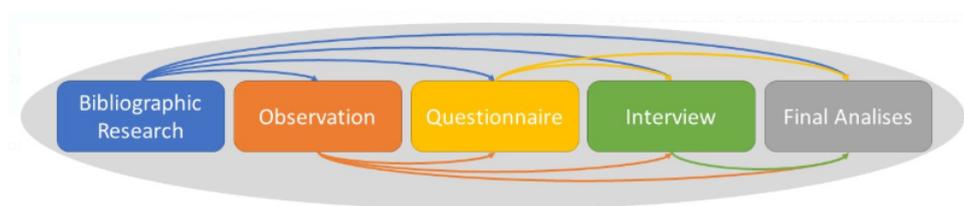


Figure 2. Data Triangulation Schema [34].

Seeking to reach the objective of this study, three research questions (RQ) were defined:

1. **RQ.1.** What are the Design Thinking techniques used and what are the challenges faced in the requirements elicitation of agile software development projects?
2. **RQ.2.** How do Design Thinking techniques work as a method of eliciting requirements in a real agile software development project?
3. **RQ.3.** What are the results obtained in the use of Design Thinking as requirements elicitation method in agile software development project and was there evidence of the use of Design Thinking regarding the identified challenges?

To answer RQ.1 we performed a systematic literature review. RQ.2 and RQ.3 were answered using case study, observation, questionnaire and interview.

#### 4. Systematic Literature Review

We present here our a Systematic Literature Review (SLR) of existing works that approach the challenges of using Design Thinking techniques faced by the software developing community, regarding requirements elicitation in the context of agile methodologies. The SLR contributed to answer the first research question, as mentioned in the Section 3, as means of primary studies [35]. During the SLR, the Planning, Conduct and Publication of Results were followed, as defined in the work presented by Kitchenham [35], together with the Manual Search and the Snowballing [36] in order to answer RQ.1.

##### 4.1. SLR Process Method

Our SLR used Automatic Search [37], which consists of applying the search string in the electronic databases, followed by the manual search [37], through which searches were performed for works in conferences, newspapers or magazines. The manual search activity was performed in the annals of the conferences and periodicals specific to the Software Engineering area. The automatic search was performed in the following databases:

- Digital library ACM;
- Digital Library IEEE Xplore; and
- DBLP-Computer Science Bibliography.

#### 4.1.1. Search String

The definition of a search *string* was according to the set of criteria PICO [38–40]: P (population) corresponds to a specific role within the lifecycle of the research focus, an area of application, or even a specific group of the industry; I (intervention) delimits the research focus within a broader scope; C (comparison) identifies alternatives and compares with the delimitation performed in the intervention; and O (outcomes) lists what is intended to be achieved, measured, improved or affected in relation to the population.

By defining the criteria, the following search string was created: (“agile software development” OR “agile methodologies”) AND (“design thinking”) AND ( requirements elicitation OR requirements elicitation OR requirements analysis).

#### 4.1.2. Selection Criteria (Inclusion and Exclusion)

The following selection criteria were defined for the selection of primary studies:

1. The paper must be available in the previously defined digital databases.
2. The year of publication of should be between 2007 and 2018. However, classical sources with definitions were also considered. The period of 10 years was suitable for work selection, according to important research guides in the area of software engineering [13,38,39,41–43].
3. The study must have been written in English or Portuguese.
4. The work is classified as gray literature, that is, it is technical reports, preliminary studies, technical specifications, or official documents of specific organs [38].

As criterion of exclusion of the studies was considered as the non-fulfillment of some of the inclusion criteria, as well as:

1. Incomplete works that were published as short papers.
2. Do not present sufficient information to extract the expected data, thus impairing the quality or relevance of the work [41].

#### 4.1.3. SLR Quality Criteria

The evaluation of the quality of the studies identified after the execution of the search process allowed the selection of the most relevant articles to compose the SLR, executed through the three stages defined by Kitchenham [35] and by Kitchenham et al. [13], which are presented in Figure 3.

The stages adopted in the search strategy were:

1. Execution of search strategy involving automatic and manual searches.
2. Identification of potentially relevant studies, based on reading the title, abstract and keywords. In this stage, it is possible to discard studies that are clearly irrelevant to the research. In case of doubt about the permanence of any study in the SLR, the next stage assists in this definition.
3. Reading the introduction, methodology and conclusion of the pre-selected works, again applying the inclusion and exclusion criteria.

The papers that were selected in Stage 3 were read in full and the volume of articles resulting from this stage were used to compose the SLR and support the answers to the research questions.

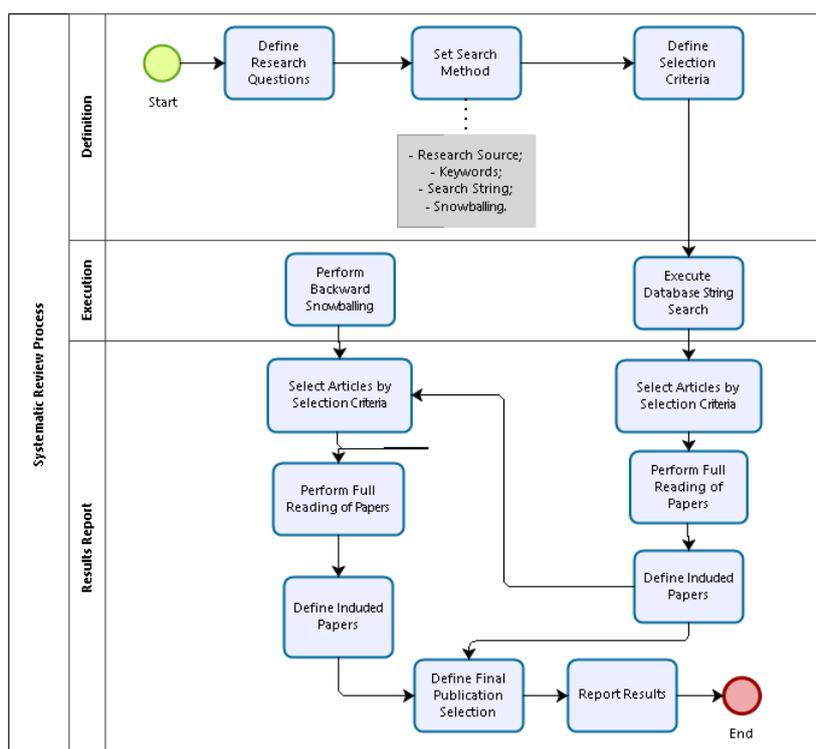


Figure 3. Stages of the paper selection process.

#### 4.2. Selection of Primary Studies

The SLR was conducted by using a search string, applied in some electronic databases. The outcome was a total of 24 publications which were filtered by the read of its title, abstracts and keywords. The application of this filter led to five publications being selected. Another filter was applied according to the inclusion and exclusion criteria, as defined in the research. After the application of these filters, four publications were classified to data extraction. Beyond this, the search strategy involved the use of manual search found that generated more 162 publications and, after resulting applying the previous filters including a complete reading, had as its outcomes 16 publications selected. Thus, after executing an automatic and manual search (Snowballing), **20 primary studies** were selected to answer the research questions. Figure 4 resumes the results obtained during data collection. The primary studies selected are presented in Table 1.

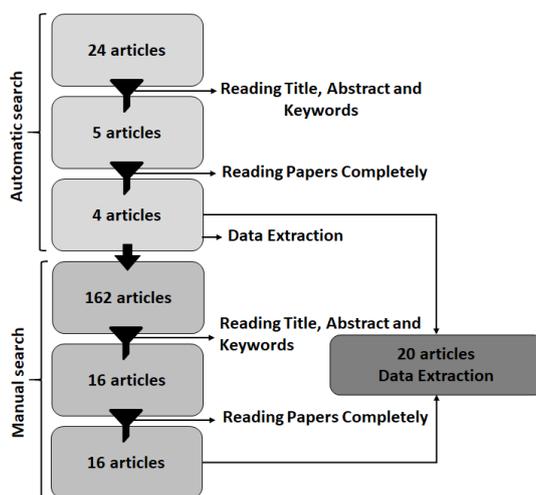


Figure 4. Articles selected from the Systematic Literature Review.

**Table 1.** SLR selected articles.

ID	Title	Reference
S1	Agile Requirements Engineering: A systematic literature review	[21]
S2	The profession of it beyond computational thinking	[44]
S3	LODPRO: learning objects development process	[45]
S4	Projeto ágil: Um Modelo Combinado com Base em Pensamentos de Desenho e Metodologias Ágeis para Projetos de Jogos Digitais	[12]
S5	A systematic literature review on agile requirements engineering practices and challenges	[5]
S6	Agile requirements engineering practices and challenges: an empirical study	[4]
S7	Reframed Contexts: Design Thinking for Agile User Experience Design	[9]
S8	A Systematic Literature Review for Agile Development Processes and User Centred Design Integration	[46]
S9	Investigating the Link Between User Stories and Documentation Debt on Software Projects Development	[47]
S10	O Modelo de Design Thinking como Indutor da Inovação nas Empresas: Um Estudo Empírico	[11]
S11	Change by Design	[7]
S12	Change by Design: How Design Thinking Transforms Organizations and Inspires Innovate	[10]
S13	From palaces to yurts: Why Requirements Engineering Needs Design Thinking	[26]
S14	Promoting the Elicitation of Usability and Accessibility Requirements in Design Thinking: Using a Designed Object as a Boundary Object	[48]
S15	The Use of Design Thinking for Requirements Engineering: An Ongoing Case Study in the Field of Innovative Software-Intensive Systems	[28]
S16	Aligning healthcare innovation and software requirements through design thinking	[31]
S17	Experimenting with design thinking in requirements refinement for a learning management system	[49]
S18	Design Thinking and Agile Practices for Software Engineering: An Opportunity for Innovation	[32]
S19	The role of design thinking and physical prototyping in social software engineering	[50]
S20	Design thinking: A fruitful concept for it development?	[51]

#### 4.3. SLR Results

##### 4.3.1. RQ.1. What Are the Challenges Faced in the Requirements Elicitation of Agile Software Development Projects and What Are the Design Thinking Techniques Used?

SLR allowed answering the first research question **RQ.1**. A case study carried out in 16 software development organizations in the United States revealed seven challenges found in the Requirements Elicitation (RE) by using software methodologies. Among them, three are directly related to the requirement elicitation activity [4].

1. **Estimation of costs and schedule:** It is more difficult to develop accurate estimates of costs and schedule during the initial stages of an agile project than those found in traditional ones. In traditional methods, the definition of the problem lies in the initial phase of the project and, in agile approaches, the definition of the problem is unstable in these stages, and the aim of the

project is subject to constant changes [4]. However, we noticed that short iterations and frequent feedback of the stakeholders help agile development teams to create better estimates of time and individual costs for each iteration [4].

2. **Attention to non-functional requirements:** A large concern of the ER in agile software development lies in the lack of attention regarding non-functional requirements, because they are poorly defined or ignored in the early stages of the project development. Stakeholders often focus on the main functionality and ignore technical issues related to the scalability, maintenance, portability, safety and performance [4].
3. **Customers' participation:** Communication with stakeholders will depend on many factors such as availability, consensus and confidence, especially in the early stages of the project. An American health software developer who participated in [4] empirical studies reported that the best users will always be occupied with their activities in the organization or their boss will not allow them to join the development team in full time. "In fact, in my experience, even part-time is a problem. We were fortunate to have about 3 to 5 hours per week with the product manager. When a system involves more than one group of stakeholders and each one is focused on different system features, reaching consensus or compromise in the iterations can be a challenge. In addition, each stakeholder may not have a complete understanding that his needs were fully understood by the development team because of the iterative nature of the process which are often not fully understood by the users"[4].

Software developers do not usually read all the artifacts produced by the project [21]. To solve this problem, it is important to find the right combination of artifacts which best fit into the context of the project, so that members can work better. In addition, users stories are not sufficient to define the question of usability, since they do not describe these aspects. To work better with the usability, even though user stories have enough focus on technique, usability needs to be treated on a higher level than that of the development of stories [21].

A systematic review of the practices and challenges encountered in the requirements engineering in agile software methodologies presented by Inayat et al. [5] reports that the practices of requirements elicitation used in the agile methodologies for software development can compensate some of the deficiencies found in traditional approaches. However, this flexible approach still faces some challenges such as lack of user involvement, negligence of non-functional requirements and problems with the estimation of cost and schedule, which are directly related to the phase of requirements elicitation.

Salah et al. [46] presented seven challenges that are integrated in the use of the user-centered design framework and agile software development, and two of these challenges are directly related to the activity of requirements elicitation.

1. **Planning of initial activities:** The initial phase is a pre-development stage divided into agile projects with the aim of assessing the initial requirements and understanding the users needs and goals. In the agile approach, the team focuses on results only in terms of functionality. Thus, the agile methods discourage the developer community to make better plans for the initial project activities, because they are sensitive to a change of requirements.
2. **Short amount of time in the iteration to realize usability tests:** Agile teams report a lack of time in the iteration for performing usability tests during development, leaving aside requirements verification and validation.

Soares et al. [47] indicated nine challenges met during software requirements elicitation in agile methodologies, which are presented in Table 2.

**Table 2.** Challenges met during requirements elicitation [47].

ID	Challenge	Description
1	Prioritization of requirements	The prioritization of requirements based only on the value they have for the user poses a risk for the project.
2	Identification of non-functional requirements	The lack of specification of non-functional requirements can provoke future problems.
3	Itemization of requirements	The agile methodology usually approaches users' stories as a means of representing requirements. Very often they only present a low-level of detail causing difficulties for other development activities.
4	Change of requirements	One needs to pay attention to the constant changes of requirements in agile projects by evaluating its impacts and risks.
5	Definition of requirements	The difficulties that users face in figuring out what is to be described by the development team.
6	Interconnection between requirements	Difficulty in dealing with the interconnection between requirements, as in the agile development, user stories can be developed individually.
7	Involvement of the stakeholders	Users are not always available to answer questions about software requirements.
8	Communication with stakeholders	It very often proves difficult to communicate with stakeholders in an efficient manner. This can lead to undesired consequences, as the agile requirements are based on the continuous communication and cooperation with the user.
9	Validation of requirements	The validation of requirements can be flawed due to the low level of itemization of the requirements when using agile methodology.

Table 3 presents the results of all the challenges met during software requirements elicitation by using agile methodologies and some referenced primary studies.

**Table 3.** Challenges met in software requirements elicitation.

ID	Challenge	Reference
1	Cost estimate and schedule	[4,5]
2	Attention to no-functional requirements	[4,5,21,47]
3	Users participation	[4,5,47]
4	Correct combination of artifacts and their respective use	[21,31]
5	Planning of initial activities	[21,46]
6	Time for usability tests	[46]
7	Prioritization of requirements	[5,44,47]
8	Itemization of requirements	[12,47]
9	Change of requirements	[45,47,51]
10	Definition of requirements	[47–49]
11	Interconnection between requirements	[4,9,47]
12	Validation of requirements	[28,32,47,50]

## 5. Design Thinking Techniques Used in Requirements Elicitation of Agile Software Development

Requirements engineering involves the stages of data acquisition, analysis and negotiation, specification and validation, and it is not usually enough to satisfy this demand, except by immersing into the user's context [26].

There is a certain conflict in the way that requirements engineering and agile development define user's participation. The agile development tends to have a greater effort designing the code itself, rather than a more rigorous documentation, and it requires greater involvement of the user during the development process. The requirements engineering activities tend to reduce user's participation after the initial mapping. However, even though agile development is guided by user descriptions, its outputs are obtained by functional and non-functional requirements' perspective, with guidance towards user and a good professional team, it is still complicated to capture what is necessary in a whole. However, it became necessary to search for a way to bring the agile characteristics into the realm of requirements engineering, which includes the elicitation phase as well as the management of the challenges in the method [26].

Some works indicate a solution to this problem of using Design Thinking approach as an initial practice of requirements engineering, elicitation and prototyping and support user involvement, as defended by the agile methodology [26]. The relationship between DT and agile methodologies is so obvious that the literature points out the similarity between them, as presented in Table 4.

**Table 4.** Relationship between Design Thinking and agile methods' concepts [12].

Characteristics	Design Thinking	Agile Method
Solution of unstructured problems	Mainly designed to solve complex issues	Developers deal with uncertainties, hence unstructured problems are part of their daily work
Users needs are important for the development of the project	Most of the Design Thinking approaches are in essence focused on the users and their experiences	To guarantee a better quality and aptitude of the final product, it is necessary to involve users and project team in an active form during the product development
Iterative productive process	Design Thinking defines that interaction is fundamental for the development of solutions	Agile processes are mainly iterative
Team Cooperation (interdisciplinary and multidisciplinary)	Practitioners of Design Thinking work in teams, via a multidisciplinary interaction	Developers who use agile methodology are basically formed by multidisciplinary teams, consisting of programmers, project managers, testers, etc.
Rapid prototyping	Rapid prototyping is important in the phase of ideation, the use of prototypes helps identify new creative and innovative ideas	Support a more efficient project development, enabling high quality and cost efficiency products

A project management model covers the whole process of developing digital games through the combination of two approaches to Design Thinking and agile methodology. This model can be used as a method for requirements elicitation in agile methodology, mainly in applications which require creativity [12]. Throughout the history of creative solutions for solving problems, designers have developed a set of techniques to help them control the three phases of Design Thinking: immersion, ideation and prototyping [7]. Design Thinking as well as the agile software development methodologies offer competitive advantages, in the differentiation of the product and in the efficiency of cost estimation [12]. Additionally, both approaches have many similarities, which strengthen their joint use [12].

Table 5 shows the DT techniques used together with the agile methodologies brought up in research performed by the authors of [9,12,25,44].

**Table 5.** DT techniques according to their phases [12,25,44].

Phase	Related Technique	Description
Immersion	Strategic challenge	Ask questions like: “How can we...?” [12].
	Challenge selection	Evaluate and select challenges to be worked on by the team [12].
	Knowledge Sharing	What is known and what is not known to solve the problem, and what needs to be studied to solve it [12].
	Research Planning	Plan the survey considering users, stakeholders, experts, contexts and benchmarks [12].
	Questionnaire	Research references and problematic context [12].
	Research Exploring	Research references and problematic context which helps the team to understand the context to be worked on [12,25].
	Interviews	Method which, in a conversation with a respondent, helps obtain related information about the central subject with specific questions [9,25,44].
	Reframing	Examine user issues from different perspectives to enable deconstruction of beliefs and assumptions of stakeholders [25].
	Research desk	Information research related to the project subject through diverse sources, such as websites, blogs, interviews, books and articles [25].
	Awareness notes	A way to obtain information about the users with a minimum of interference with the actions. What makes this technique different is that the user formally reports his activities [25].
Ideation	Generative sessions	Meet with stakeholders to enable them to share their experiences and plan activities around the subject of the project to present their ideas [25].
	A day in life	Members of the project team assume the paper of the user for a certain time in order to interact with the challenges that the latter faces on a daily basis [25].
	Shade	Follow-up users by project team member for a certain period, which includes the interaction with the analyzed product or service. [25].
	Sharing	Sharing all information and identifies perceptions in the ideation phase [12].
	People	Create fictitious characters to use them as a refinement for the solution [9,12,25].
	Empathy map	Develop understanding of people involved, what they feel, see, hear, speak, do, strengths and weaknesses [12,25].
	Synthesis	Synthesize the learning process [9,12].
	Brainstorming	Discuss ideas for possible solutions [12,25].
	Common creation workshop	Organize meeting with a series of activities to stimulate creativity and cooperation of the stakeholders by promoting innovative solutions [25].
	Map of ideas	Present all ideas generated during the project in order to discuss, display and identify business opportunities [25].
Ideation	Positioning matrix	Tool for strategic analysis of ideas generated in the project. It is used to validate the ideas in relation to the guiding criteria of the project in order to support the decision-making process starting from the efficient communication of the benefits and challenges of each solution, so that the most strategic ideas can be selected for prototyping [25].
	Customer Journey	Allow stakeholders to propose solutions emanating from user’s description, including thoughts and feelings [12].
	Blueprint	Visual presentation of the solution process as well as characters and activities [12].

Table 5. Cont.

Phase	Related Technique	Description
Implementation	Prototyping on paper	Creation of a prototype that expresses the ideas of participants on paper to schematically represent project screens [12,25].
	Model of the volume	Product presentations in levels of fidelity, from low, with few details, to high, with the presentation of the final product [25].
	Staging	Improvised simulation of a situation that represents an interaction between the user and the product [9,25].
	Storyboard	Present a story with pictures and drawings, photographs and collages in order to visualize the implementation of the solution [25].
	Prototype of the services	Simulation of artifacts, environments or interpersonal relationships which represent aspects of the product in order to engage the user and simulate the delivery of the proposed solution [25,44].
	Hypothesis and tests	Hypothesis for the solution and testing of the hypothesis [12].
	Pivoting	Changing of the solution based on the obtained results [12].
	Considerations	Take into consideration the team that participated in the development of the solution [12].

Queiros et al. [45] presented a study on how the use of technology in education has attracted the attention of researchers in the last few years. It shows that brainstorming is mentioned as well as three other techniques that can be used in the process of DT: introspection cards, conceptual maps and mental maps. Design Thinking can be used from the view of strategic management of innovation using techniques such as: (1) ethnography, a practice of observing how people behave in their day-to-day life or how they exercise a particular activity in organizations; (2) storytelling, a technique which helps create a common understanding of the challenge that is being explored; and (3) brainstorming, i.e., obtaining and evaluating ideas [11]. DT can transform organizations and inspire innovation through the use of mental map, prototypes of paper, scenarios and high fidelity prototypes which are DT techniques that can help stakeholders improve and discover new requirements [10].

In our systematic literature review, we identified 12 challenges that organizations confronted within the elicitation of requirements in agile projects and 31 Design Thinking techniques: 13 techniques are related to the immersion phase, 10 techniques are related to the ideation phase and 8 techniques are related to the implementation phase.

## 6. Requirements Elicitation with the Use Design Thinking: A Project Case Study at the Brazilian Federal Court of Accounts (TCU)

To answer **RQ.2** and **RQ.3**, a practical case study was carried out in an institution that used requirements elicitation by using Design Thinking. This study was conducted from December 2017 to August 2018. The implementation stages of the case study were guided by a timeline organized by the research team.

The institution is the [Federal Court of Accounts \(TCU\)](#), the Brazilian authority responsible for the evaluation of accounts of public administrators and is also responsible for money, federal public funds as well as the accounts of any person who causes loss, theft, corruption or other irregularities that results in damage to the national treasury. It is a federal autonomous and independent body whose main basic functions can be grouped as follows: judicial, advisory, informative, sanctioning, corrective, normative and as ombudsman.

One of the main projects executed by the Information Technology Secretary (STI) covers the appreciation of legality of granting pensions, reforms and admission of staff of the federal public service. The motivation for choosing this project came up by verdicts which have identified structural and operational difficulties in the determination and instruction of these acts. Among them, a low quality of information reaching the TCU by means of filled out forms, with errors and omissions,

which, even with great effort for clarification, created issues for more precise automated analysis. This raised the amount of acts for all those involved in the process, by consequently ignoring deadlines and complicating the detection of acts which were not forwarded to the TCU.

In addition, there was a need to review all the legislation that conducted the acts subject to registration. Thus, a presentation of a detailed schedule for the implementation of improvements in the treatment of the acts was determined to facilitate the execution of the activities with greater efficiency, extending the automated analysis for information and, consequently, reducing the need for allocating providers for the manual and individual analysis of acts, optimizing the time needed to analyze these documents.

TCU/STI expected a system that could focus on the quality of the data with automatic verifications of the data input with a differentiated treatment of the roles played by the manager of staff. The roles are as issuer and as registrar of the act, allowing also that the personnel manager and the internal controller to organize their work by creating internal divisions as well as providing a user-friendly and adjustable interface within the context of each type of act. Due to its size and importance, this project took more than two years to be delivered.

Figure 5 shows the levels of the system and how the actors play a role in each of them. Each unit level has a profile associated with their respective functions.

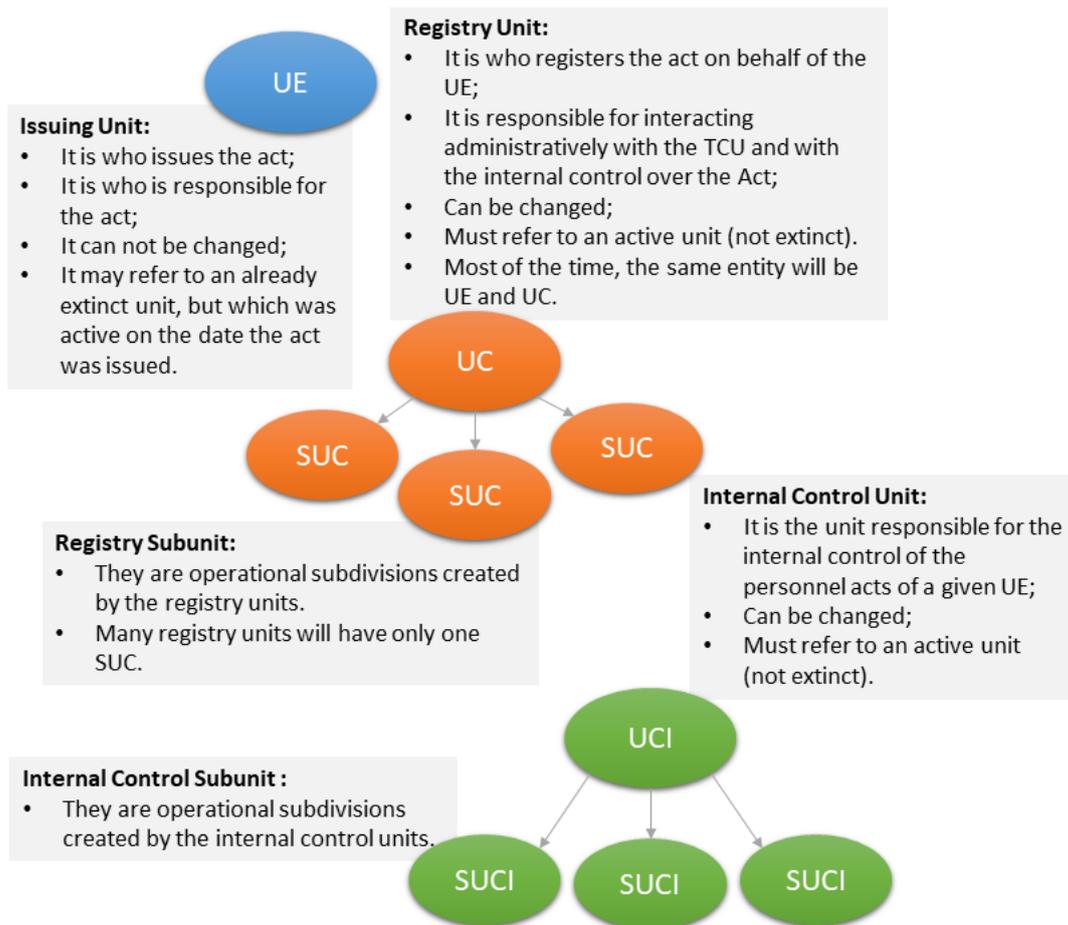


Figure 5. e-Pessoal System Actors. Source: TCU.

6.1. RQ.2. Do Design Thinking Techniques Contribute to Overcome Challenges of Requirements Elicitation in a Real Agile Software Development Project and What is the Input DT Gives to the Challenges Identified in the Case Study?

The project was led by two development teams: a team responsible for the elicitation of requirements and another with 6 stakeholders from different institutions. It took a 4-h requirements gathering workshop involving a total of 40 participants. Its objective was to identify the needs, wishes and difficulties of the users concerning the information about the acts of the staff, as shown in Table 6. The first activity performed in the workshop aimed at integrating users, because many of them were from different institutions. Interaction was necessary for a good development of the workshop as a whole. After the interaction was presented to the conception of Design Thinking, a method that would be adopted for the requirements elicitation, by making the collaborative aspect of the approach evident. The idea was to work together with the users and not for them, by seeking solutions based on understanding and experience of those who work with the system on a daily basis.

**Table 6.** Quantity of stakeholders of the case study participants and some of their profiles.

Stakeholders	Total
<b>Internal Team Profile</b>	
Developer	7
Product Manager (PM)	1
Product Owner (PO)	2
User Experience Design (UX)	3
<b>Users (Clients)</b>	
TCU	2
Ministry of Planning	2
Superior Court of Justice	2
Brazilian Federal Court of Accounts	2
Brazilian Post and Telegraph Corporation	1
Federal Savings Bank	4
Federal Reserve of Brazil	4
Bank of Brazil	2
Brazilian Petroleum Corporation	1
Brazilian Company of Hospital Services	3
Regional Federal Courts	1
Brazilian Army	3
<b>Total</b>	<b>40</b>

The requirements elicitation workshop obtained information about the desires and needs of the users of the e-staff system, which was compiled and subdivided into groups, in order to facilitate the work of the team in defining the scope of the project. The workshop results were used as an input for the definition of the project scope, after a preliminary evaluation of all the data collected with the aim of grouping them according to their characteristics. Thus, items that dealt with the same subject or could in some way contribute to the subject by a defined need were gathered in order to organize the information in a better way and consequently provide a better treatment. During the workshop, Design Thinking techniques were used, as shown in Table 7.

After the scope definition, the software development was initiated. In this first moment, several functionalities that had to be implemented were defined, a task which would take an estimated four months to be completed. The development of these functionalities was based on the generated prototypes. The solution was made available for all users compliant to the planning; some adjustments were carried out after delivery, in accordance with the users validation. Thus, the use of Design Thinking techniques was compliant to the development of the actual project adopted in this case study.

As mentioned above, we performed requirements elicitation in a software project of the Brazilian Federal Court of Accounts (TCU) by using Design Thinking techniques. The objective was to observe how the practitioners applied the techniques of DT in their daily activities. During the case study, nine DT techniques were observed by TCU practitioners: strategic challenge, challenge selection, re-framing, generative sessions, co-creation Workshop, brainstorms, ideas menu, hypotheses and tests, and paper prototyping.

**Table 7.** Design Thinking techniques used in the scope definition meetings.

Phase	Technique	Description
Ideation	<i>Brainstorming</i>	Discussion of ideas for possible solutions.
Implementation	Prototype on paper	Prototype design expressing the ideas of participants in small amounts of paper to present the screens of the project schematically.
	Hypothesis and tests	Hypothesis proposed for the solution and hypothesis testing.

### 6.2. RQ.3. What Were the Results Obtained in the Use of Design Thinking in an Agile Software Development Project and Was there Evidence of the Use of Design Thinking Regarding the Identified Challenges?

We proposed an approach with the intention of evaluating the use of Design Thinking in the requirements elicitation via agile methodologies. The approach was developed in parallel with the execution of the case study, identifying potential evaluation criteria, the potential interviewees and how to make assessment. The resulting evaluation process applied in the case study was subsequently generalized for use in similar contexts. The goal of the evaluation approach was to evaluate whether the use of Design Thinking techniques can contribute positively to the challenges identified in the requirements elicitation, in projects developed with agile methodologies.

DT is focused on the user/stakeholder to obtain collaboration, interaction and practical approaches in order to find the most appropriate ideas and, consequently, coherent final solutions [11]. In addition, as the level of satisfaction with the product is obtained by the user/stakeholder and the level of satisfaction with the methodology adopted in the development of the software is obtained by the project team, we considered that the participants of the evaluation could be the user/stakeholder and the project team. Bearing in mind that the evaluation method would be conducted by people, two methods were chosen: questionnaires and mixed interviews. A questionnaire is a data collection instrument, consisting of a sorted series of questions that should be answered in writing and without the presence of the interviewer. An interview combines pre-defined questions with the possibility of creating new questions based on the topics of the interviews [52].

The choice of two different evaluation methods gives more variety to the approach, allowing it to be applied in any project context. The decision of which method to choose depended on several factors, such as the size of the team to be evaluated, the size of the evaluation team and the time available for data collection and analysis. Therefore, taking into account the advantages and disadvantages of the two methods, the use of questionnaire is recommended for the case when the team that is being evaluated is large and the evaluating team, data collection and analysis are small. The interview is recommended when the evaluated team is small and the quantity of members of the evaluation team, the data collection and analysis are considered adequate, for example. The questions were based on DT techniques and also based on the explanation of challenges observed in the elicitation of the identified requirements.

The best moment to do the evaluation was based on two points: to verify whether the application of DT technique had been perceived and whether its use contributed positively to the challenges of requirements elicitation. The first evaluation phase took place shortly after the collection of requirements and involved final users and other stakeholders. The second evaluation phase happened right after the delivery of the product and also involved the users/stakeholder with the objective of capturing the compliance of the defined requirements with the delivered product, the positive

contribution of the use of DT regarding the challenges and the satisfaction with the product and the used methodology.

The questionnaire is composed of 11 questions which were answered by the participants of the requirements gathering. It has two parts: the first part was applied shortly after the workshop and the second right after the delivery of the product. Tables 8 and 9 present the result of the workshop questionnaire and the results obtained from the participants' responses. The user assigned a grade from 1 to 5 to indicate the level of agreement with the presented statements: the closer to one (1), the lower the degree of agreement/satisfaction, while, the closer to (5), the higher the degree of agreement/satisfaction.

**Table 8.** Results of the workshop questionnaire.

Question	Average
1. There was an incentive to present ideas for the system during the workshop.	5.00
2. There was cooperation among participants in the presentation of ideas.	4.85
3. The team discussed solutions for the presented ideas.	4.54
4. The main needs of the system were identified during the workshop.	4.23
5. By the end of the workshop, the team reached consensus over the system requirements.	4.08
6. Questions related to the response time of the operations were considered.	3.92
7. Questions related to the security of the system were considered.	3.77
8. Questions related to the aesthetics of the system were considered.	3.69
9. The length of the workshop was sufficient to fulfill its intentions.	4.31
10. The methodology used in the workshop can be applied in future projects dealing with a similar approach.	4.54
11. What is your level of satisfaction with the workshop?	4.54

**Table 9.** Results of the questionnaire after the product delivery.

Question	Average
1. The identified needs of the workshop are now present in the system.	4.29
2. The interaction with the system works intuitively.	4.00
3. The system offers adequate security utilities.	4.14
4. The response time of the system is adequate.	3.57
5. What is your level of satisfaction with the information panel?	3.71
6. My satisfaction with the final product is partially due to the use of the applied methodology in the workshop for the assessment of requirements.	4.14

The interviews were conducted with the members of the project development team who were grouped into four different profiles:

1. **Project Manager/Requirements (PM):** Responsible for the project planning, definition of goals and monitoring its implementation as well as for the collection and analysis of all the software requirements information.
2. **Product Owner (PO):** Responsible for the requirements, assisting in prioritizing them and validating the implementation in order to ensure the quality of the final product.
3. **Developer (DV)** Responsible for the development of the system.
4. **User experience/interface UX/UI design:** Responsible for planning the best practice and interaction with the system.

The interview with the team was conducted by taking into account the main aspects obtained from the questionnaire with clients/users. The intention was to conflict the answers obtained in the questionnaires with the information that was given by the team. The questions used as the baseline of the interview were the same for all areas, but the answers took different directions according to the interviewed team. For instance, a conversation with a developer is more technical than an interview with a product owner, turning the interview to a more specific level in each area. Thus, after all interviews, it was possible to merge the information obtained and realize that they complemented each other and gave more support to the conclusions.

The interviews, in general, lasted a 30 min and, to synthesize the answers of the members, we considered the common views and divergences, by choosing to cite the individual points of view of each interviewee. Table 10 shows the results obtained and the questions can be found at: [Questions](#).

**Table 10.** Questions and sample answers of the interviews with the project's participants.

ID	Question	Answer
1.	How did it work with cost estimation and timeline for the information panel of the e-staff?	A cost estimation was not made, since it was a project developed by the internal staff. The focus was on the timeline. The schedule estimation was carried out on the basis of the evaluation of the needs assessed during the workshop.
2.	Did the methodology used in the survey workshop for the assessment of requirements to some extent help estimating cost and schedule? If yes, how?	Yes. From the needs raised from the workshop, it was possible to estimate how complex it would be to obtain and present user's requested information. Thus, it was possible to establish a timetable for the project delivery.
3.	Was there discussion about the non-functional requirements during the workshop and were they considered in the project development phase? If so, how was that?	The non-functional requirements were discussed superficially during the workshop, with some considerations. Only in the development stages the usability and performance requirements were considered, due to the volume of information that was presented.
4.	The users and stakeholders' participation happened in an effective manner throughout the project development? If not, what is the reason?	Yes. The users and stakeholders actively participated in each stage of the prioritization, testing, revalidation and assessment of requirements activities, in addition to indicating and suggesting how information should be organized and displayed in the panel.
5.	The information generated during the workshop for the requirements assessment consisted in the elaboration of artifacts that supported the project development?	Yes, because the purpose of the workshop was very specifically designed to develop an information panel. Hence, the question was to discover, together with the users, which information they would like to have and how it should be arranged. Thus, it was possible to gather an initial understanding of the project scope and to plan its releases.

Table 10. Cont.

ID	Question	Answer
6.	Compared to other projects, was there a greater effort in planning the initial activities? If yes, has this contributed to a better quality of the requirements?	There was already a formerly previous version of the developed system, with all the information already structured, thus we already had an idea of the data needed to assess and to meet users' requirements. There was an effort to plan the workshop activities well in order to focus on what we had to extract from the users/stakeholders. We knew that by planning the workshop led us to obtain the most specified—and therefore higher quality—requirements for the users and project needs.
7.	Did the workshop enable the team to obtain the requirements which precisely expressed the real users' needs? Was this due to the methodology used?	Yes. I believe that it is due to the methodology used, because users had the freedom to express their needs. The workshop was entirely focused on the expression of the users' needs.
8.	The information obtained from the workshop enabled the definition of more detailed requirements?	Yes. In the workshop, users from different authorities were present, each with a vision, work context and proper needs. This not only added value to the discussion, it also helped the requirements team to gather more detailed higher quality information so that all users were taken into account.
9.	After the requirements assessment workshop, were there meetings with the project team to discuss the information obtained to define the work scope?	Yes, there were several meetings afterwards.
10.	After the requirements assessment workshop, were there meetings with the project team in order to prioritize the requirements?	Yes. A meeting was held to prioritize the requirements.
11.	After the requirements assessment workshop, were there meetings with the project team to discuss the interdependence between the requirements?	Yes.
12.	Were there prototypes on paper created to express the workshop gathered ideas in order to present the system screens?	Yes. Several prototypes on paper were created.
13.	When an idea of solution was identified, was it tested?	No, they were not tested but were only discussed.
14.	What is your level of satisfaction with the methodology used at the workshop?	I am partially satisfied with the methodology adopted at the workshop.
15.	Would you adopt or suggest the adoption of this methodology for the assessment of requirements in future projects with a similar approach?	Yes. The used methodology facilitated the requirements elicitation.
16.	What is your level of satisfaction with the final product?	I am satisfied with the final product.
17.	Usability testing was made before the actual development of the system?	It was not. Usability tests would be best performed at the end of the development.
18.	The details of the project requirements were sufficient for the development of the activities which you performed? If not, why?	Yes. The stakeholders were very precise and provided a lot of information. In addition, all graphics were prototyped before the start of the development.
19.	Was the time to perform the usability tests adequate?	No usability testing was performed.
20.	Usability testing began to be made before the actual development of the system?	No usability testing was performed.

In this paper, we present an approach designed to evaluate the use of Design Thinking in the requirements elicitation in software development agile methodologies. The approach was elaborated during the case study, as observed elements served as input to identify the best evaluation method. From the nine techniques observed in the project, we concluded after the evaluation that eight of them were used and only one, Hypotheses and Tests, was partially considered. Additionally, out of the thirteen challenges perceived in the workshop, there was evidence of the use of Design Thinking in eight of them, evidence of a partial contribution in three of them and no contribution in two.

### 6.3. Discussion of the Results RQ.2. and RQ.3

Through the analysis of the answers obtained from questionnaires and interviews, we can see in the case study that there is evidence of an input to eight of the challenges identified through the use of DT.

1. **Participation of users/stakeholders:** From the user's point of view, they realized that there was a proper incentive to assist the project team in assessing the initial requirements, through the exposition of ideas and the underpinning during its construction, the discussing of needs and the reaching of consensus regarding the requirements. This incentive was caused by means of DT technics used in the workshop. From the point of view of the project team, the stakeholder participated actively in the course of the project, contributing to the prioritization and validation of requirements, testing and improvements. This contribution was due to the dynamics adopted in the project meetings, which used DT techniques and required the participation of the stakeholder.
2. **Requirements Definition:** From the users point of view, we may affirm that the use of DT techniques in the workshop helped both users and the team to define the requirements and deliver the product, and the defined requirements were reflected in the final product. From the point of view of the project team, the DT techniques used in the workshop gave users the opportunity to express their wishes and needs and indeed fulfilled their main purpose, which was to define the requirements of the system.
3. **Requirement Validation:** From the users point of view, it may be asserted from their responses that the use of DT techniques enabled an environment for discussing requirements, thus allowing to validate them. From the team's point of view, the techniques used together with the diversity of the users present in the workshop with their different points of view, contexts of work and needs enriched the discussion and forced the project team to meet the most relevant requirements so that everyone could feel included and taken into account.
4. **Project Schedule Estimation:** According to the project team, the needs identified during the workshop allowed to estimate the complexity of the project in a way that the elaboration of an estimation of a more appropriate schedule could be enabled. As a result, a chain effect could be observed: the use of DT in the workshop provided a better identification of the project needs and allowed the preparation of a more reliable schedule.
5. **Initial Activities Planning:** According to the project team, there was a great effort in planning the initial activities, via the study of the DT and an analysis of its practical implementation, so that the main focus of the workshop was the stakeholder's involvement in order to allow the extraction of system key features.
6. **Details of the requirements:** In accordance to the project team and considering the participation of the stakeholder, the system requirements were well detailed regarding the prototyping, the DT approach and the regular presence of the stakeholder in the course of the project, which was also fostered by DT.

**Prioritization of requirements:** In accordance with the project team, the dynamics applied in the meetings during the project, which were based on DT techniques, were important to prioritize the requirements.

7. **Interdependence between requirements:** In accordance with the project team, the dynamics applied in meetings in the course of the project, which were based on DT techniques, were important to identify the interdependence the requirements.

There is partial evidence of the contribution of this project to three specified challenges through the use of DT, following an analysis of the questionnaire and of the interviews.

1. **Attention to non-functional requirements:** In accordance to the users' vision, questions related to non-functional requirements were addressed during the the workshop, but not so efficiently, so that the DT techniques used did not promote a discussion about this. Taking all the questions asked in the after-workshop questionnaire, those questions related to the non-functional requirements were the ones which obtained a lower average. Evaluating the view of the users after the delivery of the product, they observed the presence of some resources related to non-functional requirements, mainly concerning safety and the question of system usability. From the point of view of the team, they considered that the non-functional requirements were addressed superficially during the workshop, and that they were only considered more effectively during the development phase mainly related to the usability, which is due to the use of prototyping and performance which is due to the type of system they were developed. Due to the characteristics of the applied system, it is difficult to come to a conclusion on the influence of the DT on non-functional requirements. However, it can be asserted that DT fosters the discussion of some items related to non-functional requirements, such as usability.
2. **A correct combination of artifacts and their proper use:** In accordance with the vision of the team and because the workshop had well defined goals, they succeeded together with the users, identified in a well-defined way which were the key needs and characteristics of the system and, starting from this, elaborated the artifacts which would be necessary for each member of the team. However, it can be seen that, in the decision making of what artifacts should be elaborated in a project, the DT does not influence the choice. The methodology only provides the input needed to elaborate the artifacts, with the team being responsible for the definition of which artifacts will be used.
3. **Change of requirements:** According to the vision of the team, due to the regular participation of the stakeholder to discuss ideas and to create prototypes, items which are related to DT, the atmosphere seemed favorable for the dealing with changes of requirements in a way that it would not cause much impact. The only mistake perceived in the development of the system in the study was the lack of testing when discussing and prototyping these new ideas. This made it difficult to analyze the real need of change in the requirements. Hence, there is partial evidence for the contribution of a challenge related to the changes of the requirements, but this is more due to the negligence of the team than to the methodology itself.

Finally, it can be seen, through the analysis of the questionnaires and interviews, that there were no indications of a contribution in two specified challenges through the use of DT:

1. **Cost estimate:** According to the perspective of the team and because the project was carried out within a public institution by internal staff, the cost estimate was not dealt with in this project, only the timeline was relevant to the context. Therefore, the influence of DT on the resolution of the challenge of the cost estimate cannot be evaluated.
2. **Time available to perform usability tests:** In accordance with the team's point of view, a usability testing was not realized. The only tests performed in the project were functional. Therefore, the influence of DT in the resolution of challenges related to the time to perform the usability tests cannot be evaluated.

In the questionnaires and interviews which were carried out, in addition to the collection of data to answer if DT techniques helped the team solve challenges arising from activities to elicit requirements

in an agile software environment, the staff and users were asked about their level of satisfaction with the applied workshop/methodology and the final product, on a scale from 1 to 5.

The feedback obtained from the use of the workshop/methodology, for both the user's development side and the team's side, was generally satisfactory, since the average of responses was 4.46 from the perspective of users and 4.25 from the view of the team. It can be asserted that the users have been partially satisfied with the final product and the development team has been fully satisfied, since the average of the responses here are 3.71 and 5.0, respectively.

## 7. Limitations and Threats to Validity

The findings presented in this review study have the following limitations and threats to validity.

We considered all the studies which provided empirical, case study, experimental, practices, industrial and survey related information about challenges faced in the requirements elicitation of agile software development projects.

The findings of this systematic literature review cannot be generalized because the results are based on a specific set of keywords and the research repositories that were used for the data collection. Therefore, our results could be limited and cannot be applied to every projects and organizational setup.

We cannot guarantee that all relevant primary studies were selected in systematic literature review. It is possible that relevant papers were not chosen. To mitigate it, we performed the automatic search, and complemented it by performing manual search to try to collect all primary studies in this field. During the data extraction process, the primary studies were classified based on our judgment. To mitigate this threat, the classification process was performed using peer review.

Internal validity is mainly related to the capability of replicating similar findings. We addressed this aspect by defining and later following the systematic literature review procedure, described in Section 4. Four researchers were involved in the review process, who, over a period of time, worked together to avoid duplications and achieved consensus in the acceptance of the identified primary studies. However, it could be possible that, if this study were replicated by other researchers, minor variations in the identified studies would be observed due to differences in personal aptitude and thinking. Regardless of this fact, the findings presented in this review will enable readers to obtain a clear picture of challenges for software requirements elicitation using Design Thinking.

Another threat is that perceptions of the interviewees could be biased towards their own beliefs. These beliefs could cause some distortions when interpreting the reality. To reduce this threat, the chosen practitioners were those who had more experience within organization.

## 8. Conclusions

The systematic review of the literature identified 12 challenges that organizations were confronted with in the elicitation of requirements in agile projects and 31 Design Thinking techniques, 13 of which are related to the immersion phase, 10 are related to the phase of ideation and 8 are related to the implementation phase (prototyping). These challenges and techniques served as an input to assess whether the use of DT helps the internal design team of the practical case study to meet the recurrent challenges of eliciting requirements in agile methodologies identified by systematic review in the project studied in this work.

To carry out this evaluation, a process of executing the case study was developed, in which the use of DT was observed during the development of a module belonging to the development of software by the TCU, e-staff (e-pessoal in Portuguese). During this observation, it was assessed which techniques of DT were used by stakeholders, both in the DT workshop and during the development of the software module. During the execution of the case study, a more generic approach was elaborated which can be used in other contexts in which DT is used with the same objective of assessing whether it helps staff face existing challenges in eliciting requirements via agile methodologies.

During the implementation of the process, stakeholders were asked via questionnaires and interviews whether they actually perceived the use of the observed techniques and whether they

contributed positively to challenges identified in the systematic literature review. In addition to that, interested parties gave their opinion of the application of DT (workshop) and the final product, after the development of the elicited requirements.

Of the 31 techniques, the application of nine techniques was observed: Strategic challenge, Challenge selection, reframing, generative sections, brainstorming, cooperation workshop, list of ideas, prototype of paper and hypotheses and test. It can be concluded that, through the obtained results in the questionnaires and interviews, they were perceived by people interested in the use of these techniques, except for the technique of hypotheses and test which was partly used because, according to the interested parties, the hypotheses raised during the workshop were prototyped, but not tested.

With regards to the 13 challenges identified in the systematic review, evidence was found that DT strengthens stakeholders participation in the definition, detailing, validation, interdependence and prioritization of requirements, mainly in the prototyping; in the estimation of the schedule; and in the planning of the initial activities. What was not perceived in stakeholders/users view was an incentive to discuss non-functional requirements during the workshop for the assessment of requirements; only during the development of the project was a major concern with usability verified, due to the use of prototyping and performance, and also due to the characteristics of the software being developed.

In addition to that, as the system in the study is part of a larger system, several questions related to non-functional requirements were extended to the module, for instance, reliability, security and support. Due to this, the project team dealt less with non-functional requirements.

About the correct combination of artifacts and their use, the project team believes that DT provides sufficient input for the elaboration of coherent artifacts, however it cannot confirm that every project team using DT may choose these artifacts and use them correctly. This is in fact the task of a team rather than the use of DT. Finally, although DT provides support for teams to identify new requirements through prototyping, there was a negligence of the team which did not execute a DT technique correctly, which prevented this point from being evaluated. This is partially because, despite creating prototypes of these new requirements, they were never actually tested before their implementation.

Regarding the challenges of cost estimation and the time to perform usability testing, it was not possible to identify evidence of a contribution. It can be concluded through the application of the evaluation approach in the studied software project that there is evidence of a contribution, by the use of Design Thinking, for most of the challenges of the elicitation of requirements obtained from an agile methodology.

For future studies, we can propose the realization of other practical studies for a better consolidation of the obtained results, with different types of systems and diverse teams, in a way that allows for an application of all the challenges found in the literature.

**Author Contributions:** Design Thinking: challenges for software requirements elicitation was made by H.F.M.; A.C.d.O.J.; E.D.C.; R.A.D.K.; R.A.P.; and E.C.O. All authors contributed to Writing, reviewing and editing the manuscript.

**Funding:** This publication was funded by PPCA/Postgraduate Program on Applied Computing from the Dept. of Computer Science CIC/UnB, University of Brasília, Brasil.

**Acknowledgments:** This research work has the support of the [Research Support Foundation of the Federal District \(FAPDF\)](#) research grant 05/2018.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Hofmann, H.F.; Lehner, F. Requirements engineering as a success factor in software projects. *IEEE Softw.* **2001**, *18*, 58. [[CrossRef](#)]
2. Brooks, F.P. No Silver Bullet. *IEEE Comput.* **1987**, *20*, 10–19. [[CrossRef](#)]
3. Beck, K.; Grenning, J.; Martin, R.C. Manifesto ágil. Capturado em. 2011. Available online: <http://agilemanifesto.org/> (accessed on 31 October 2019).

4. Ramesh, B.; Cao, L.; Baskerville, R. Agile requirements engineering practices and challenges: An empirical study. *Inf. Syst. J.* **2010**, *20*, 449–480. [[CrossRef](#)]
5. Inayat, I.; Salim, S.S.; Marczak, S.; Daneva, M.; Shamshirband, S. A systematic literature review on agile requirements engineering practices and challenges. *Comput. Hum. Behav.* **2015**, *51*, 915–929. [[CrossRef](#)]
6. Pressman, R.S. *Software Engineering: A Practitioner's Approach*; Palgrave Macmillan: New York, NY, USA, 2005.
7. Brown, T. Change by design. *J. Prod. Innov. Manag.* **2009**, *28*, 381–383. [[CrossRef](#)]
8. Sun, W.N.; Schmidt, C. Practitioners' Agile-Methodology Use and Job Perceptions. *IEEE Softw.* **2018**, *35*, 52–61. [[CrossRef](#)]
9. Adikari, S.; McDonald, C.; Campbell, J. Reframed Contexts: Design Thinking for Agile User Experience Design. HCI (9). In *Lecture Notes in Computer Science*; Springer: Berlin, Germany, 2013; Volume 8012, pp. 3–12.
10. Brown, T.; Katz, B. *Change by Design: How Design Thinking Can Transform Organizations and Inspire Innovation*, 1st ed.; Harper Collins: New York, NY, USA, 2009.
11. Bonini, L.A.; Sbragia, R. O modelo de design thinking como indutor da inovação nas empresas: Um estudo empírico. *Rev. De Gest Ao E Proj.-GeP* **2011**, *2*, 3–25. [[CrossRef](#)]
12. Higuchi, M.M.; Nakano, D.N. Agile Design: A Combined Model Based on Design Thinking and Agile Methodologies for Digital Games Projects. *Rev. De Gest Ao E Proj.* **2017**, *8*, 109. [[CrossRef](#)]
13. Kitchenham, B.; Pretorius, R.; Budgen, D.; Brereton, O.P.; Turner, M.; Niazi, M.; Linkman, S. Systematic literature reviews in software engineering—a tertiary study. *Inf. Softw. Technol.* **2010**, *52*, 792–805. [[CrossRef](#)]
14. Fadel, A.C.; Silveira, H.d.M. Metodologias ágeis no contexto de desenvolvimento de software: XP, Scrum e Lean. *Monogr. Do Curso De Mestr. FT-027 Ao De Proj. E Qual. Da Fac. De Tecnol.* **2010**, *98*, 101.
15. dos Santos Soares, M. Metodologias ágeis extreme programming e scrum para o desenvolvimento de software. *Rev. Eletrônica De Sist. De Informaç Ao* **2004**, *3*. [[CrossRef](#)]
16. López-Alcarria, A.; Olivares-Vicente, A.; Poza-Vilches, F. A Systematic Review of the Use of Agile Methodologies in Education to Foster Sustainability Competencies. *Sustainability* **2019**, *11*, 2915. [[CrossRef](#)]
17. Nardi, J.C.; de Almeida Falbo, R. *Uma Ontologia de Requisitos de Software*; CibSE: London, UK, 2006; pp. 111–124.
18. Kotonya, G.; Sommerville, I. *Requirements Engineering: Processes and Techniques*; Wiley Publishing: Hoboken, NJ, USA, 1998.
19. Aurum, A.; Wohlin, C. Requirements engineering: Setting the context. In *Engineering and Managing Software Requirements*; Springer: Berlin, Germany, 2005; pp. 1–15.
20. De Lucia, A.; Qusef, A. Requirements engineering in agile software development. *J. Emerg. Technol. Web Intell.* **2010**, *2*, 212–220. [[CrossRef](#)]
21. Schön, E.M.; Thomaschewski, J.; Escalona, M.J. Agile requirements engineering: A systematic literature review. *Comput. Stand. Interfaces* **2017**, *49*, 79–91. [[CrossRef](#)]
22. Horkoff, J.; Ersare, J.; Kahler, J.; Jorundsson, T.D.; Hammouda, I. Efficiency and Effectiveness of Requirements Elicitation Techniques for Children. In Proceedings of the 2018 IEEE 26th International Requirements Engineering Conference (RE), Banff, AB, Canada, 20–24 August 2018; pp. 194–204.
23. Tiwari, S.; Rathore, S.S. A Methodology for the Selection of Requirement Elicitation Techniques. *arXiv* **2017**, arXiv:1709.08481.
24. Du, J.; Jing, S.; Liu, J. Creating shared design thinking process for collaborative design. *J. Netw. Comput. Appl.* **2012**, *35*, 111–120. [[CrossRef](#)]
25. Vianna, M.; Vianna, Y.; Adler, I.; Lucena, B.; Russo, B. *Design Thinking: Inovação em negócios*; MJV Press: Atlanta, GA, USA, 2012.
26. Vetterli, C.; Brenner, W.; Uebernickel, F.; Petrie, C.J. From Palaces to Yurts: Why Requirements Engineering Needs Design Thinking. *IEEE Internet Comput.* **2013**, *17*, 91–94. [[CrossRef](#)]
27. Roach, T. How to Combine Design Thinking and Agile in Practice. 2015. Available online: <https://medium.com/startup-frontier/how-to-combine-design-thinking-and-agile-in-practice-36c9fc75c6e6> (accessed on 31 October 2019).
28. Hehn, J.; Uebernickel, F. The Use of Design Thinking for Requirements Engineering: An Ongoing Case Study in the Field of Innovative Software-Intensive Systems. In Proceedings of the 26th IEEE International Requirements Engineering Conference, Banff, AB, Canada, 11 July–31 October 2018; pp. 400–405.

29. Correa, L.; Maria, D.; Bellio, J.C.; Marczak, S.; Conte, T. O Uso de Design Thinking no Apoio ao Desenvolvimento de Software: Um Estudo de Caso no Contexto de Academias de Musculação. In Proceedings of the Anais do WER18—Workshop em Engenharia de Requisitos, Rio de Janeiro, Brasil, 5–6 September 2018; doi:10.17771/PUCRio.wer.inf2018-25. [[CrossRef](#)]
30. Plattner, H.; Meinel, C.; Weinberg, U. *Design-Thinking*; Springer: Berlin, Germany, 2009.
31. Carroll, N.; Richardson, I. Aligning healthcare innovation and software requirements through design thinking. In Proceedings of the International Workshop on Software Engineering in Healthcare Systems, Austin, TX, USA, 14–22 May 2016; pp. 1–7.
32. Corral, L.; Fronza, I. Design Thinking and Agile Practices for Software Engineering: An Opportunity for Innovation. In Proceedings of the 19th Annual SIG Conference on Information Technology Education, Fort Lauderdale, FL, USA, 3–6 October 2018; pp. 26–31.
33. Denzin, N.K. *The Research Act: A Theoretical Introduction to Sociological Methods*; Routledge: London, UK, 2017.
34. triviños, A.N.S. Introdução à pesquisa em ciências sociais: A pesquisa qualitativa em educação. São Paulo: Atlas, 1987. *Outros Números Do Inf. Rural ETENE ANO 2009*, 3, 25.
35. Kitchenham, B. Procedures for performing systematic reviews. *Keele UK, Keele Univ.* **2004**, 33, 1–26.
36. Wohlin, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, London, UK, 13–14 May 2014; p. 38.
37. Silva, F.S.; Soares, F.S.F.; Peres, A.L.; de Azevedo, I.M.; Vasconcelos, A.P.L.; Kamei, F.K.; de Lemos Meira, S.R. Using CMMI together with agile software development: A systematic review. *Inf. Softw. Technol.* **2015**, 58, 20–43. [[CrossRef](#)]
38. Felizardo, K.R.; Nakagawa, E.Y.; Fabbri, S.C.P.F.; Ferrari, F.C. *Revisão Sistemática da Literatura em Engenharia de Software: Teoria e Prática*; Elsevier: Amsterdam, The Netherlands, 2017.
39. Pai, M.; McCulloch, M.; Gorman, J.D.; Pai, N.; Enanoria, W.; Kennedy, G.; Tharyan, P.; Colford, J.J. Systematic reviews and meta-analyses: An illustrated, step-by-step guide. *Natl. Med J. India* **2004**, 17, 86–95. [[PubMed](#)]
40. Biolchini, J.; Mian, P.G.; Natali, A.C.C.; Travassos, G.H. Systematic review in software engineering. *Syst. Eng. Comput. Sci. Dep. COPPE/UFRJ, Tech. Rep. ES* **2005**, 679, 45.
41. Petersen, K.; Vakkalanka, S.; Kuzniarz, L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Inf. Softw. Technol.* **2015**, 64, 1–18. [[CrossRef](#)]
42. Zhang, H.; Babar, M.A. Systematic reviews in software engineering: An empirical investigation. *Inf. Softw. Technol.* **2013**, 55, 1341–1354. [[CrossRef](#)]
43. Wohlin, C.; Prikladniki, R. Systematic literature reviews in software engineering. *Inf. Softw. Technol.* **2013**, 55, 919–920. [[CrossRef](#)]
44. Denning, P.J. The profession of IT Beyond computational thinking. *Commun. ACM* **2009**, 52, 28–30.
45. Queiros, L.M.; Da Silveira, D.S.; da Silva Correia-Neto, J.; Vilar, G. LODPRO: Learning objects development process. *J. Braz. Comput. Soc.* **2016**, 22, 3. [[CrossRef](#)]
46. Salah, D.; Paige, R.F.; Cairns, P. A systematic literature review for agile development processes and user centred design integration. In Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, London, UK, 13–14 May 2014; p. 5.
47. Soares, H.F.; Alves, N.S.; Mendes, T.S.; Mendonça, M.; Spínola, R.O. Investigating the link between user stories and documentation debt on software projects. In Proceedings of the 2015 12th International Conference on Information Technology-New Generations (ITNG), Las Vegas, NV, USA, 13–15 April 2015; pp. 385–390.
48. Levy, M. Promoting the Elicitation of Usability and Accessibility Requirements in Design Thinking: Using a Designed Object as a Boundary Object. In Proceedings of the 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW), Lisbon, Portugal, 4–8 September 2017; pp. 156–159.
49. Freitas, R.; Peres, S.; Fantinato, M.; Steinbeck, R.; Araújo, U. Experimenting with design thinking in requirements refinement for a learning management system. In *Anais do IX Simpósio Brasileiro de Sistemas de Informação*; SBC: Porto Alegre, Brazil, 2013; pp. 182–193.

50. Newman, P.; Ferrario, M.A.; Simm, W.; Forshaw, S.; Friday, A.; Whittle, J. The Role of Design Thinking and Physical Prototyping in Social Software Engineering. In Proceedings of the 37th International Conference on Software Engineering, Florence, Italy, 16–24 May 2015; pp. 487–496.
51. Lindberg, T.; Meinel, C.; Wagner, R. Design thinking: A fruitful concept for it development? In *Design Thinking*; Springer: Berlin, Germany, 2011; pp. 3–18.
52. Marconi, M.d.A.; Lakatos, E.M. *Fundamentos de Metodologia Científica*, 5th ed.; Atlas: São Paulo, Brazil, 2003.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).