Journal of Software Engineering Research and Development, 2021, 6:1, doi: 10.5753/jserd.2021.xxx

A Tool Proposal for Recommending Design Thinking Techniques in Software Development

Rafael Parizi [**Pontifícia Universidade Católica do RS** | ***rafael.parizi@edu.pucrs.br***]
Marina Moreira [**Pontifícia Universidade Católica do RS** | ***marina.moreira@edu.pucrs.br***]
Igor Couto [**Pontifícia Universidade Católica do RS** | ***igor.couto@edu.pucrs.br***]
Sabrina Marczak [**Pontifícia Universidade Católica do RS** | ***sabrina.marczak@pucrs.br***]
Tayana Conte [**Universidade Federal do Amazonas** | ***tayana@icomp.ufam.edu.br***]

Abstract

Design Thinking (DT) has been incorporated into software processes by teams from startups to large companies, used in problem exploration fostering innovative solutions, and integrated into agile methods to satisfy the real stakeholders' needs. DT places the customer needs up-front and helps to empathize with users, examining their behaviors, and producing outcomes focused on users' demand. A set of techniques, like personas and user journey, can be used to support this human-centered approach. Selecting which technique to use might be challenging since factors such as the application scenario, stakeholder engagement level and previous knowledge of the problem-to-be-solved may vary from case to case. In our previous work we presented a DT session conducted to better understand the need for a recommendation tool, followed by a requirements elicitation activity in which we defined how the tool should work, and an early evaluation of the initial tool low-fidelity prototypes. Inspired by the Design Science Research method, in this article we extend our previous work by furthering what we performed in the tool's definition process. We included in this work a requirements refining activity through high-level fidelity prototypes, and a requirements validation activity through a questionnaire-based feedback collection with professionals who have experience on the use of DT in software development. Our results show that our proposal offers an useful and easy to use tool to recommend DT techniques with the potential to support those professionals who apply DT in requirements engineering by suggesting techniques that are the best fit to the declared context.

Keywords: *Design Thinking, Technique, Recommendation System, Collaborative Tool, Design Science Research.*

1 Introduction

Design Thinking is a problem-solving approach that brings human-centered design principles at its core (Docherty, 2017). Driven by the search for innovation, it fits the development of new products, services, or processes, from startups to large and complex environments (Brown, 2008), thus being used by software development teams.

As a human-centered approach(Hehn et al., 2020), DT can be used in the early phases of software development process to identify what the customer needs are, providing better support for downstream development activities, mainly those related to identifying a proper solution for the problem-at-hand. The multidisciplinary view offers software teams a dynamic and collaborative problem-solving environment; bringing all involved parties closer together (Seidel and Fixson, 2013).

By bringing the user needs to the center of the discussion, DT also improves team communication and facilitates knowledge domain acquisition, which are well-known issues in software development (Lindberg et al., 2011). DT is also considered an easy-in integration with and a way to boost agile development (Przybilla et al., 2018). Despite the use, navigating in this new world might be challenging. Literature offers a plethora of DT tools (or techniques for simplification) that form the toolkit to perform DT activities. Moreover, there is a lack of studies mentioning strategies to support the decision process of which techniques to use and detailing which contextual factors (e.g., previous knowledge about the problem, customer engagement, etc.) affect such decision.

Therefore, considering the DT leans and how it can support the discovering of user needs and scoping of a solution (Hehn et al., 2020), well-known activities of the Elicitation phase of the Requirements Engineering discipline, our long-term research goal is to support software development professionals' decision in the selection of which DT techniques to use in a certain scenario (or context).

To achieve our goal, we adopted a long-term multi-phase research process, as follows: i) exploratory studies including a literature review (Prestes, 2020) and a survey (Prestes et al., 2020) to understand the use of DT in software development and to identify the difficult from industry professionals for selecting DT techniques, ii) proposal of a tool to initially support the DT techniques decision process (Parizi et al., 2020a) and to later establish a community of users that can retro-feed the efficiency of the recommendations, iii) exploratory study based on interviews to model the decision making of IT professionals on the selection of DT techniques, iv) refining of the DT techniques recommendation tool to reflect the decision-making model, v) empirical studies to validate the generated DT techniques decision model.

Inspired by the Design Science Research (DSR) method, this paper shows the activities we performed in phase (ii) a tool proposal[1]. We extended the 3 activities we presented in a previous work (Parizi et al., 2020b), by adding 2 new ac-

[1]This paper does not detail in depth the previous studies that we conducted as exploratory research activities for understanding the problem and for determining the technological rule. Such details can be found in Prestes (2020) and in Prestes et al. (2020).

tivities for solution's requirements refining and validation. More specifically, in Parizi et al. (2020b) we presented a) a DT session conducted to better understand the need for a recommendation tool, b) a requirements elicitation activity in which we defined how the tool should work, and c) an early evaluation of the initial tool low-fidelity prototypes. In the current work, we further our tool definition by presenting 2 new activities: a requirements refining activity through high-level fidelity prototypes, and a requirements validation activity through a questionnaire-based feedback collection with professionals who have experience on the use of DT in software development. Results show that the recommendation tool should consider the context in which the software product will be used and project characteristics (e.g., team expertise), feedback from others using the recommendation, and results from the tool usage itself (e.g., the most selected techniques). Also, as pointed out in the validation activity in this extended article, the tool seems to be useful and easy to use, exploring a sense of community that helps professionals establish some shortcuts in selecting DT techniques.

The remainder of this paper is outlined as follows: Section 2 introduces DT in software development in a nutshell. Section 3 presents the methodology followed in this study, while Section 4 illustrates the activities performed for the elicitation, refining and validation of the requirements of the DT techniques recommendation tool proposed in this study. Section 5 discusses the study results. Section 6 clarifies the threats to validity of our study. Finally, Section 7 presents the final remarks and the future directions for our research.

2 Background

Design Thinking is as a human-centered approach to problem-solving by exploring the users' and businesses' needs, transforming ideas into an acceptable and validated solution (Brown, 2008). DT can be understood as "*a way of describing a designer's methods that is integrated into an academic or practical management discourse*" (Sköldberg et al., 2013). Hiremath and Sathiyam (2013) argue that DT is increasingly used in software development companies as a tool for innovation. It offers iterative learning from the beginning of the development cycle, including continuous improvement (Hehn and Uebernickel, 2018).

Requirements Engineering (RE) literature has also increased its interest on the topic. For instance, Hehn and colleagues' work on approaches for tailoring and integrating DT and RE (Hehn et al., 2020). Hehn, Uebernickel, and Fernandez's (Hehn et al., 2018) proposed an integration of Design Thinking with software engineering processes aiming to improve the quality of requirements. These are examples of how RE can benefit from DT.

The DT approach is inherently multidisciplinary, involving stakeholders from different knowledge areas. It also uses empathy as the underlying concept to seek for what is technologically appropriate and strategically feasible to be built when proposing a solution (Brown, 2008).

Moreover, Brenner et al. (2016) consider that DT can be conceptualized in the light of three perspectives: as a process, as a mindset, and as a toolbox. The first perspective defends that DT is a set of distinct and iterative steps (e.g., empathize, define, ideate, prototype, test). These steps abstract working spaces to explore the problem, propose a solution and validate it, which can be adapted and executed non-sequentially according to the necessity. Literature (e.g., Lucena et al. (2016); Luma (2012); Stanford d.School (2019)) present different process models to abstract these working spaces. The second perspective argues that DT as a mindset inspires a human-centered view, placing the user as the center of attention, seeking to understand what the problem is, among others. Finally, the third perspective supports DT as a 'toolbox', i.e., a set of techniques (e.g., persona, user journey) that aid the achievement of the planned process steps. It acts as an 'operational layer' that makes it possible to obtain a solution to a problem. Our research assumes this third view as the focus of interest without disregarding the others.

Literature discussing DT as a set of techniques has been growing. Liedtka (2015) suggests a list of techniques that can be associated with DT working spaces. The authors summarize a wide span of DT techniques, grouping them into visualization, ethnography, collaborative sense-making, assumption surfacing, prototyping, co-creation, and field experiment techniques. In software development, Rozante de Paula et al. (2020) that argue the key for conducting DT sessions with no quality loss is by choosing the proper techniques. They also pointed out that these techniques allow the stakeholders to gather different points of view, boosting creativity and innovation. Dobrigkeit and de Paula (2019) argue that the use of appropriate techniques during DT workshops is a key success factor. To the authors, the perception of DT changes according to the professional's role (developers, managers), and application of DT might be different within a same team.

Despite the large number of studies reporting on the use of DT in software engineering (Souza et al., 2017), we still have no consolidated knowledge on how developers choose DT techniques, what criteria they consider for supporting such selection, and what sources they consult for their decisions. We identified in a previous literature review (Prestes, 2020) initiatives such as the Luma Institute's Innovating for People Toolkit (Luma, 2012), which explains the meaning of each technique and suggests related techniques for a certain purpose, or the IDEO Toolkit (IDEO, 2020), which recommends techniques based on a predefined set of questions that a professional might ask about on the use of DT. These examples that demonstrate that choosing a technique is not trivial. Although highly used by practitioners, both cases fail to consider the context in which DT is being used. For instance, if the team has no previous information about the stakeholders, selecting interviews might be of more interest to learn about their profile rather than a focus group session in which people might not get along and waste effort. DTA4RE - Design Thinking Assistant for Requirements Elicitation (Souza et al., 2020) is a first step towards aiming to tailor the recommendation process. However, DTA4RE is still limited in the number of contextual factors it considers to recommend a technique and on not taking into account any kind of feedback from those who use the recommendations.

Aiming to fill this gap, in Parizi et al. (2020b) we introduced an innovative DT techniques recommendation tool that uses the collaboration between IT professionals as the mechanism of recommendation. Here, in this work we further on the presentation of our recommendation system.The recommendation system aims to aid professionals who apply DT in software development by recommending techniques based on the collaboration of other experienced professional. In addition, by keeping the dynamic and flexible nature of DT, the tool also supports the use of DT for other software development activities beyond RE. The tool is aligned to Hehn et al. (2020) that integrated DT for RE indicating that DT can be revisited during the development process whenever the team perceives the need to understand the user's needs (e.g, from gathering feedback of a release delivery, etc.).

3 Research Methodology

This section presents the research methodology aiming to answering the following research question: *How can we support software development professionals to select DT (set of) techniques during requirement engineering?*

This study is inspired on the Design Science Research (DSR) method (Runeson et al., 2020; Hevner, 2007; Wieringa, 2014). DSR promotes problem-solving by exploring instances of the problem in practice, creating artifacts that result in better constructs (human-made designs) in specific contexts. Wieringa (2014) introduced DSR in SE arguing that "Design Science is the design and investigation of artifacts in context. The artifacts [...] are designed to interact with a problem context in order to improve something in that context".

Runeson et al. (2020) presented an iteration-based framework for DSR in SE research. The framework has 3 components (or activities): *Problem conceptualization activity* (problem understanding), which expresses the understanding of a general problem as instances (concrete problems); *Solution Design activity* (solution design approach): which represents a creative activity of problem-solving where solutions for the problem at hand are ideated, and; *Empirical validation activity* (validation approach): which allows to assess whether the solution proposal is feasible for the problem, providing a room to extend the acquired knowledge.

The DSR framework proposed by Runeson et al. (2020) also includes the concept of technological rules. A technological rule describe the desired effect of a proposed solution in a particular context. It expresses the scope of validity of the solution and it helps researchers to determine practical (problem-solving) and theoretical (generalization) contributions of the research. Novelty is other element of the DSR framework proposed in Runeson et al. (2020). Novelty represents a refinement of the technological rule to summarize the contributions that have been achieved with the research.

Runeson et al. (2020) inspired their proposal on Hevner (2007) original DSR model. Hevner's model for DSR is based on 3 cycles, namely: Relevance cycle, Design cycle, and Rigor cycle. The Relevance cycle seeks to identify and understand the application context, the research problem, and the acceptance criteria that will evaluate the research results. The Design cycle contemplates the research activities themselves that iteratively enable the construction of an artifact as well as its evaluation and collection of feedback to refine the design in the next iteration. Finally, the Rigor cycle aims to guarantee that artifacts being produced are valid and contribute both to practice and theory. The Rigor cycle suggests the use of empirical methods for validating the research results and its application in a particular context.

Therefore, in this study we followed the DSR framework proposed by Runeson et al. (2020). **Figure 1** illustrates our DSR-based research methodology. We conducted the Design Science Research framework iteratively, starting with the Problem Understanding activity followed by 2 iterations of the Solution Design and Validation activities.

3.1 Problem Understanding

For the Problem Understanding, we used as input the knowledge we have gained from previous exploratory studies: a systematic literature review and an exploratory survey with professionals from the agile software development industry.

- Systematic Literature Review (Prestes, 2020): We performed a systematic review of the literature in which we assessed 78 papers between 2010 and 2019. The study aimed to characterize the use of DT in software development. We learned that there is a large set of DT models, DT techniques, and different DT techniques' selection strategies reported in literature. We also identified tools (e.g., DTA4RE, IDEO DT) that support the identification and selection of DT techniques, but none of them takes the context in which the selection of the techniques takes place.
- Survey with IT professionals from the Agile community (Prestes et al., 2020): We collected data using a questionnaire with 158 participants from the Agile Software Development Community between 2019 and 2020. We learned that 1/3 of them had a difficulty of 6 or up points (scale 1 to 10) in selecting a technique. We also learned that 83% of them chose techniques based on the product context, 81% on their previous experience (which includes learning from others), and 67% on the fitness of a certain technique to a certain DT working space phase (e.g., interviews to support discovering). Results showed that selecting a set of techniques might a challenging endeavor and shed some lights on the techniques' selection criteria.

Motivated by the findings we obtained with the exploratory studies, we worked on the definition of a technological rule for our research problem: to support IT professionals' decision to select which DT techniques to use in a specific development scenario by recommending DT techniques based on the collaboration of the DT-experienced professionals. Then, aiming to further the problem understanding, we performed a DT session. We named it as "meta-DT" since we conducted a DT session to support DT activities and to better understand the need to propose a recommendation system to support the selection process of DT techniques. The meta-DT session was performed in a 3 hours-long session. A total of 10 people participated in the activity – 5 of them from industry and with an average of 3 years of experience working with DT, and 5 graduate students with DT as a research topic and 2 of them also with industry experience. The

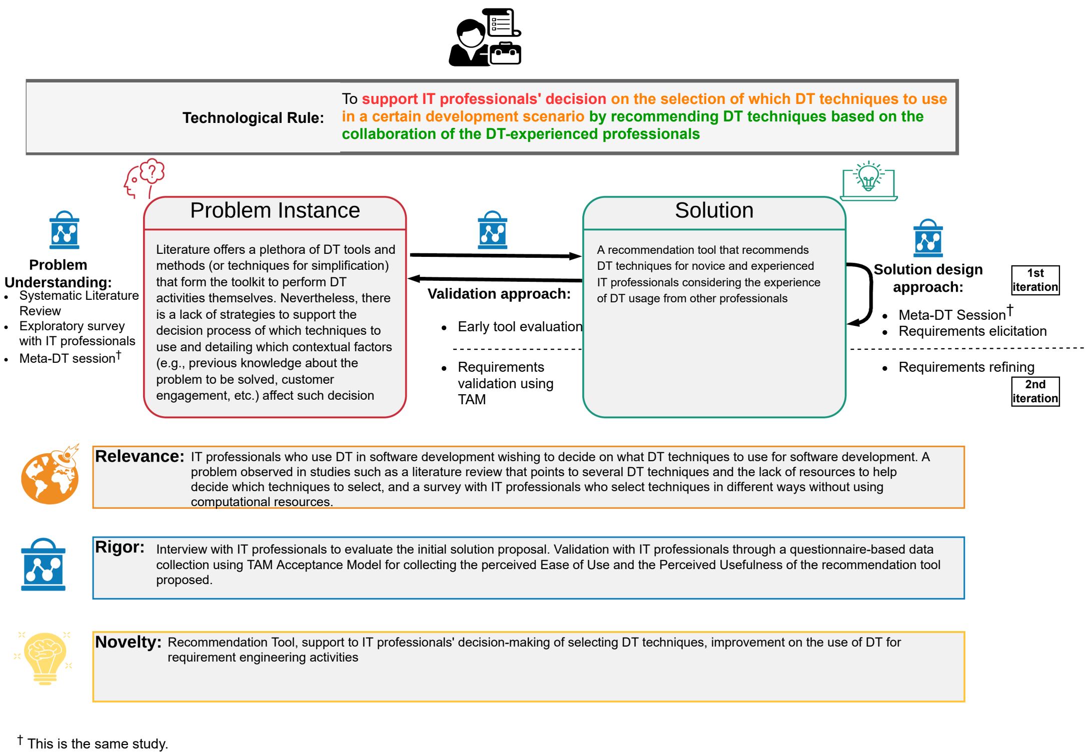


Figure 1. Overview of the Design Science Research methodology followed in this study based on Runeson et al. (2020).

meta-DT session served as a bridge between the Problem Understanding activity and the Solution Design activities of the DSR method. Therefore, in addition to help us on the problem understanding, the meta-DT session allowed us to define a first solution to the selection of DT techniques.

The session was conducted by a moderator who is a requirements analyst with 3 years of experience in Design Thinking. Considering her previous experience in conducting DT sessions, the moderator organized the problem understanding with the following steps: i) presentation of a working question, to explore in depth the problem-at-hand, and ii) problem definition, to further discuss the need for a tool.

After the problem understanding activity was completed, we were able to determine the relevance of the research (Runeson et al., 2020; Hevner, 2007). We identified that our research is relevant to IT professionals who use DT in software development wishing to decide on what DT techniques to use for software development. The problem observed in our SLR points to several DT techniques and lack of resources to help decide which techniques to select, and a survey that showed that IT professionals select techniques in different ways without considering the context information and the experience from other professionals. Since we defined the relevance of our research, we moved on to Solution Design and to Solution validation activities. We performed these activities in 2 iterations, as we describe next.

3.2 Iteration 1: Initial Solution Proposal and Early Tool Evaluation

The first iteration in the Solution Design activity was embedded within the meta-DT session. It took place in the second-half of the session, which referred to the solution space. Our goal was to propose an initial solution to the problem-at-hand. Thus, the moderator proposed the following steps: i) ideation using the Brainwriting DT technique to generate ideas (Lewrick et al., 2020); ii) convergence activity using the Affinity Diagram DT technique to find similar features (Vianna, 2012); iii) prototyping the solutions using paper-based prototypes defined by two mixed groups of 5 members; and iv) choice of solutions and presentation by each one of the groups through a voting activity. As a result, the participants concluded that the solution would be the development of a recommendation tool associated with a community building environment through feedback, and in the future, rely on gamification mechanisms to engage tool users and promote rich feedback to maintain the community.

Next, still in the Solution Design activity, we focused on eliciting requirements for the elected recommendation tool solution. Upon a 1.5 hour-long session, we conducted a requirements specification activity. This activity was conducted by the 5 graduate students who refined the tool's requirements. The chosen techniques were: i) User Journeys (Lewrick et al., 2020), to represent the needed steps for a user to achieve their (business) goals, ii) Service Blueprints (Lewrick et al., 2020) to describe how the 'service' (tool features) may be offered to the user (touch-points), covering the entire journey, identifying points for improvement and business opportunities, and iii) Low-Fidelity Prototypes (Lewrick et al., 2020), to define how users will interact with the tool . A sample of those are presented in Section 4.1.2[2]. Specifications were defined by 3 of the students and reviewed by other 2 until a consensus was reached.

[2]The full report can be found at `http://bit.do/CIBSE2020DesignThinking`

Moving towards the Validation activity in the DSR framework, in the first iteration we performed an early evaluation of the initial artifact proposed. This evaluation aimed to evaluate our tool specification with industry practitioners as a means to identify whether we were missing any relevant feature into the artifact. We interviewed 5 DT practitioners (a product designer, a service designer, a product owner, a business analyst, and an IT manager) of 2 multinational IT companies: of a large Brazilian TV broadcasting company, of a cooperative bank and of an IT provider with an average of 3.5 years of experience with DT in software development. We first openly asked them how they select the techniques they use to later present them with our recommendation tool idea and a sample of printouts of low-fidelity prototypes to discuss in details. Four of the interviews took place during coffee breaks of the DT track of an industry-based developers' conference. Each interview lasted an average of 20 min. We sent the prototypes to the interviewee by e-mail and then we discussed our proposal.

3.3 Iteration 2: Requirements Refining and Tool Validation steps

Motivated by the results of Iteration 1 and seeking for improvement to the artifact, we performed a second iteration in the DSR framework. We refined the requirements of the proposed solution and then we validated the solution with industry professionals as described next.

We used the results of the early evaluation with DT practitioners (Iteration 1 - Validation Approach) and our defined artifacts, such as user journeys, service blueprints, and low-level prototypes (Iteration 1 - Solution Design Approach) as input to the Refining requirements activity. Performed by 2 graduate students and reviewed by 2 other researchers (an assistant researcher and a senior researcher), the requirements refining activity was based on the use of the Figma prototyping tool (Figma, 2015). Figma is a prototyping web-based tool that allows the definition of transitions between prototype screens and allows the creation of content into a multiple layers' structure. Using Figma, we created high-level prototypes to refine our understanding of the elicited requirements. We transformed the low-level prototypes generated in Solution Design cycle into high-fidelity prototypes.

Next, moving to the Validation activity, we aimed for a requirements validation activity of our DT techniques recommendation tool requirements. We started by setting up a validation environment using the Quant-UX tool (Quant-UX, 2020). Quant-UX is a free prototyping and validation tool that allows to design and measure the users' interactions with a solution. Thus, we created the screens interactions aiming to collect experiences from the DT practitioners by using our DT techniques recommendation tool prototypes. Once the validation environment was set up, we collected the feedback with DT practitioners[3]. We sent invitations for 80 out of 158 respondents who have authorized us to contact them for follow-up studies. Seven of them (8.75%) accepted our invitation and participated in this validation study.

[3]These are distinct individuals from those who participated in the early evaluation study.

The validation activity was performed through a questionnaire data collection based on the Technology Acceptance Model (TAM) (Davis, 1989), which uses the Likert evaluation scale (Albaum, 1997). TAM is a behavior model that allows to evaluate users' perceptions on the acceptance of a technology. The model considers the Perceived Ease of Use (PEU) and the Perceived Usefulness (PU) factors to assess this acceptance. We created the questionnaire based on other studies such as (Dias et al., 2011; Laitenberger and Dreyer, 1998). Therefore, we carried out a 3-tasks procedure: initially, we presented our research goals using a pre-recorded video; then, we asked the DT practitioners to access the validation setup for simulating the tool's prototypes; and finally, we had them answering the questionnaire. We present the results of these studies next.

Last but not least, motivated by the results of our research, we were able to complete the novelty element of the DSR. Our research contributes to the knowledge base brought by this study are: the recommendation tool itself, the support to IT professionals' on the decision-making for selecting DT techniques, and an improvement on the use of DT for requirement engineering activities.

4 DT Techniques Recommendation Tool

This section presents in detail the 2 iterations of the DSR method that we followed in this study. We describe here the activities that we performed in Iteration 1: Meta-DT session, the requirements elicitation, the early evaluation, followed by the activities that we performed in Iteration 2: the requirements refining and validation activities[4].

4.1 Iteration 1: Initial Solution Proposal and Early Tool Evaluation

4.1.1 Solution Design: Meta-DT Session

Our meta-DT session was guided by the the following question: *In what ways would we as Requirements Analysts/Software Engineers/Designers be able to choose techniques when using DT to support software requirements?*

In a two-minutes time slot and using post-its, each participant described what she thought was important to provide relevant information to interested parties about the problem. This ideation activity resulted in 24 proposed insight cards to the established problem. Then, to gain a deeper understanding of this scenario, an Affinity Diagram was organized (see **Figure 2**). This diagram allows the organization and grouping of results (insight cards) according to their similarities, or affinities, generating a diagram that results in macro features that delimit the addressed theme (Vianna, 2012).

The 4 resulting identified macro features (in Portuguese, from left to right) are: i) qualified decision making, ii) prediction-based recommendation, iii) techniques presentation, and iv) dynamic visualization of recommendations.

[4]In this paper, we do not bring information about how the recommendation mechanisms will work (e.g. recommendation algorithm, dataset, and so on). This technical information will be included in future work.

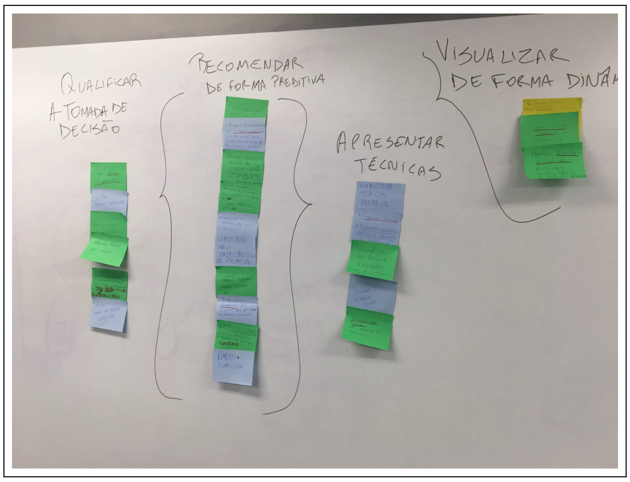


Figure 2. Affinity Diagram identifying the DT tool macro features

Next, the participants were separated into two mixed groups composed each of practitioners and graduate students to carry out a prototyping activity. These groups are named Group 1 and Group 2 from now on. The moderator introduced two Personas, Joano and Sindi, presented in **Figure 3** and previously created for the session. Personas is a technique for user modeling that helps to create fictitious characters representing the users needs, goals and desires (Ferreira et al., 2015; Tonkinwise, 2011). Each Persona was assigned to each of the groups to work on. Group 1 was assigned to the Joano persona and Group 2 to Sindi.

Through prototyping, for about 30 minutes, each participant individually designed a first round of paper-based low-fidelity prototypes to address the 4 identified macro features. **Figure 4** shows the prototypes related to the macro feature iii) present the techniques. Subsequently, each group was instructed to review the designed prototypes, group them by features and vote by posting a yellow post-it on the grouped set of prototypes by macro feature that most represented what the tool should offer.

Table 1 summarizes the identified features per persona grouped by macro feature and the respective amount of votes attributed to them (columns (V_1) and (V_2)). The voting aimed to highlight whether certain macro features stood out and deserved priority attention or a deeper discussion. Each participant could choose between 1 to 3 features. Features with zero voting indicate that none of the participants considered it a priority. The groups differ on what they consider to be most relevant. While Group 1 considered that the visualization of techniques was the most relevant feature (6 out of 10 votes), Group 2 considered that the qualified decision making was the most relevant feature (4 out of 8 votes). Given the distinct Personas' needs, this divergent scenario was not considered an issue. The table also maps the prototyped features in round 2 as described in Section 4.1.2 (e.g., (7a)) to the discussed features in the meta-DT session (prototype round 1) as indicated in the referred table (e.g., Feature i)).

To conclude, each group presented their results to one another and explained their voting and motivations behind it. This discussion promoted common ground among the meta-DT session participants and made possible to the group to conclude that they have reached a tool vision.

4.1.2 Solution Design: Requirements Elicitation

The second activity includes User Journeys, Service Blueprints and Prototyping of the screen's flow. This article presents a sample of the identified features.

User Journeys: Proceeding with the requirements elicitation for the tool proposed in this article, we started with the definition of 5 User Journeys. **Figure 5** shows a User Journey developed for Persona Sindi, highlighting her main actions when looking for a technique and aiming to use it in her product development. The identified functional requirements derived from this user journey are highlighted in **Figure 5**, and described next:

- Access the system: user access the tool;
- Recommend techniques based on a self-updated visualization graph: user can receive technique recommendations and collaborate with other users evaluating techniques, retro-feeding the recommendation graph;
- View techniques details: user can access detailed information about a certain technique, like when, how, and why to use it;
- Attach/Add techniques from the visualization graph to a new project: user have the opportunity to choose a technique and associate it to a new project.
- Create a new project: user can create a project and manage the used techniques and previous experiences, updating the visualization graph.

Service Blueprints: Service Blueprints (SB's) are used to visually present the detailed specification of aspects of a service (business feature), from the user's perspective, and other relevant parts that may be involved.

Figure 6 presents a SB for the persona Sindi, related to the User Journey showed by **Figure 5**, aiming to get new insights and a deeper understanding of the problem. In this service blueprint, the user initially accesses system (A) and selects from the suggestion graph some technique for his DT project (B). The user can also see information about the selected technique (C) as well as linking to the running project (D). This figure also presents other expected functionalities through the relationships shown in each of the horizontal lanes: physical evidence (i), customer actions (ii), contact with backstage actions (iii), and support processes (iv).

Low-Fidelity Prototypes: Considering the insights gathered from the Users Journeys and the Service Blueprints, we further detailed the features through their prototypes. We designed a second round of paper-based prototypes mapping the results from the two referred DT techniques used for requirements elicitation to consolidate our understanding.

The Techniques Recommendation Graph Screen (**Figure 7-(a)**), which represents one of the core values of our application, shows the techniques within a graph according to a specific filter. Filters represent the recommendation mechanisms implemented on the tool. These will be further and better defined later on as part of our future activities. By default, the graph is set to generate the vertices and edges according

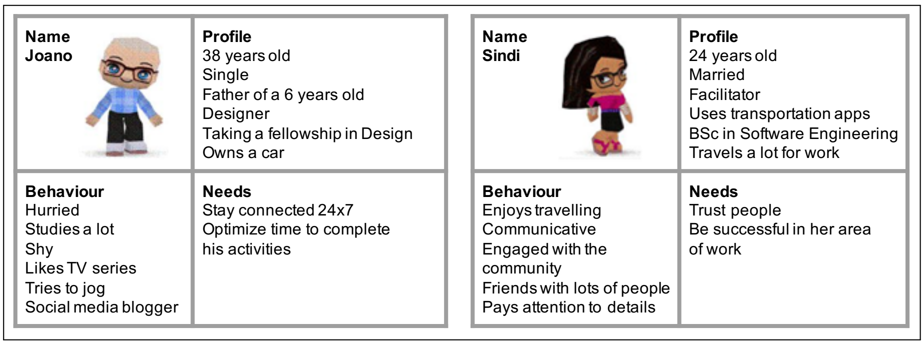


Figure 3. Personas Sindi and Joano

Table 1. Identified features per persona grouped by macro feature

Macro Feature	Joano Persona (Group 1)	V_1	Sindi Persona (Group 2)	V_2
i) Qualified decision making	- Organize Techniques by Category (7a) - Qualify Decision Making (7a, 7b, 7d)	0	- Suggest Complementary Technique (7a) - Organize Techniques by Category (7a) - Qualify Decision Making (7a, 7b, 7d)	4
ii) Prediction-based recommendation	- Get to know similar cases (7b, 7c, 7d) - Consider the context of use of DT (7a, 7c) - Provide feedback (7d)	1	- Suggest tools predictively (7a) - Receive examples of the application of the technique (7b, 7c, 7d) - Consider the context of using DT (7a, 7c) - Provide feedback (7d)	2
iii) Techniques presentation	-Visualize techniques and models (7a, 7b, 7c) - Search for technique recommendations in a systematic way (7a, 7c)	3	- Visualize the techniques and models (7a, 7b, 7c)	0
iv) Dynamic visualization	- Simulate the use of combined techniques for a particular purpose (7c)	6	- Dynamic and interactive visualization (7a, 7b)	2

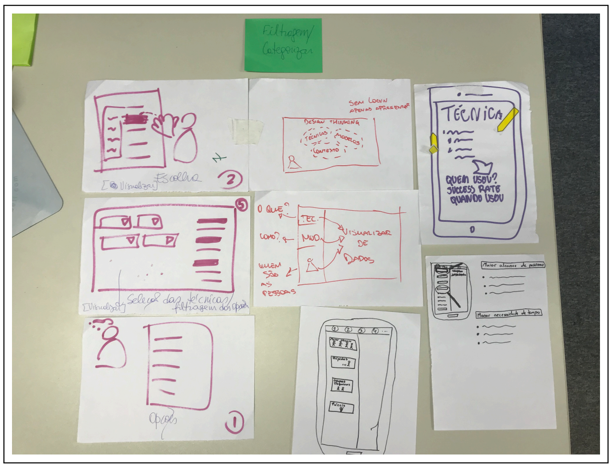
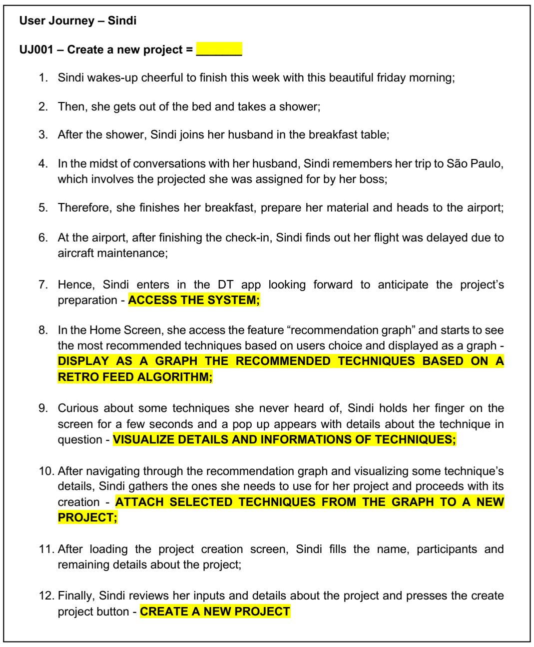


Figure 4. First round low-fidelity prototypes for the macro feature iii)

to the most used technique. Therefore, the starting and central node from this graph presents the most recommended technique, followed by the next most recommended nodes that relate to the first (e.g., Brainstorming followed by Journey Map and Personas). The central node is the most recommended technique, and the other nodes are subsequent ones. The edges are built according to the combined use of techniques by users. Thus, our tool presents a network of interconnected DT techniques. Yet, this Techniques Recommendation Graph should also consider aspects, such as a set of techniques previously applied by the software professionals community, feedback from the users about the experience of the use of a technique or a DT model, and experience from the combination of different techniques in specific scenarios of requirements engineering. These are part of our long-term research view and will be discussed only once the tool is made available.


User Journey – Sindi

UJ001 – Create a new project =

1. Sindi wakes-up cheerful to finish this week with this beautiful friday morning;
2. Then, she gets out of the bed and takes a shower;
3. After the shower, Sindi joins her husband in the breakfast table;
4. In the midst of conversations with her husband, Sindi remembers her trip to São Paulo, which involves the projected she was assigned for by her boss;
5. Therefore, she finishes her breakfast, prepare her material and heads to the airport;
6. At the airport, after finishing the check-in, Sindi finds out her flight was delayed due to aircraft maintenance;
7. Hence, Sindi enters in the DT app looking forward to anticipate the project's preparation - **ACCESS THE SYSTEM;**
8. In the Home Screen, she access the feature "recommendation graph" and starts to see the most recommended techniques based on users choice and displayed as a graph - **DISPLAY AS A GRAPH THE RECOMMENDED TECHNIQUES BASED ON A RETRO FEED ALGORITHM;**
9. Curious about some techniques she never heard of, Sindi holds her finger on the screen for a few seconds and a pop up appears with details about the technique in question - **VISUALIZE DETAILS AND INFORMATIONS OF TECHNIQUES;**
10. After navigating through the recommendation graph and visualizing some technique's details, Sindi gathers the ones she needs to use for her project and proceeds with its creation - **ATTACH SELECTED TECHNIQUES FROM THE GRAPH TO A NEW PROJECT;**
11. After loading the project creation screen, Sindi fills the name, participants and remaining details about the project;
12. Finally, Sindi reviews her inputs and details about the project and presses the create project button - **CREATE A NEW PROJECT**

Figure 5. User Journey – View techniques in detail – Persona Sindi

In addition, if a user does not know any DT techniques, she can find descriptive information about the techniques, e.g.: definition, case scenarios to use, and others (see **Figure 7-(b)**). Exactly as a haptic, when the selected technique is selected, its respective data will pop up inside a little rounded square, delivering the users dexterity and continuous use of the feature, despite having to go back and search manually about the given technique.

Figure 8-(a) provides the concept of a project to store related information held together, whereas having to seek them separately all the time. Therefore, the screen exhibits blank

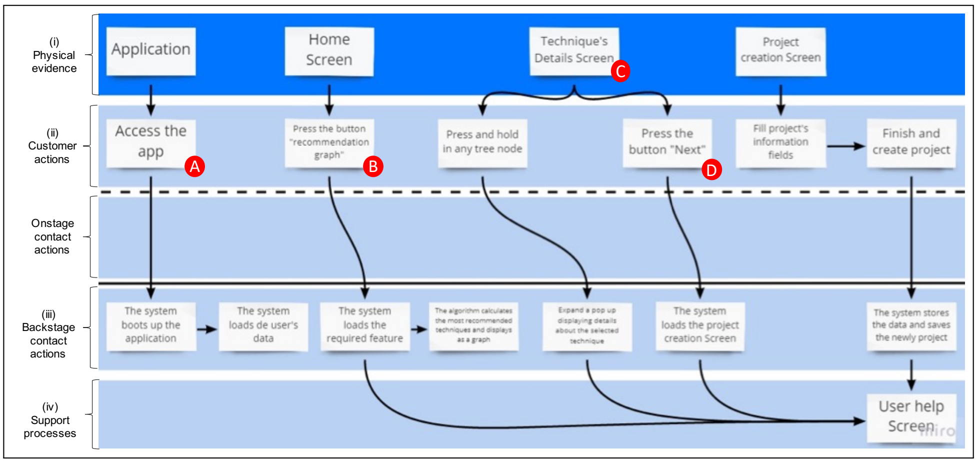


Figure 6. Service Blueprint - Persona Sindi

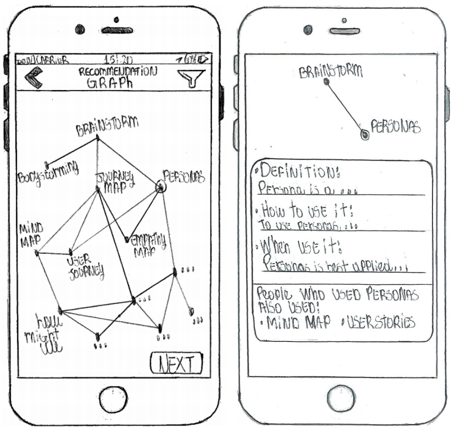


Figure 7. (a) Graph (b) Technique detail

fields to be filled about the project's crucial details (e.g.: name, description, purpose and participants), and a list of the previous selected techniques with the graph.

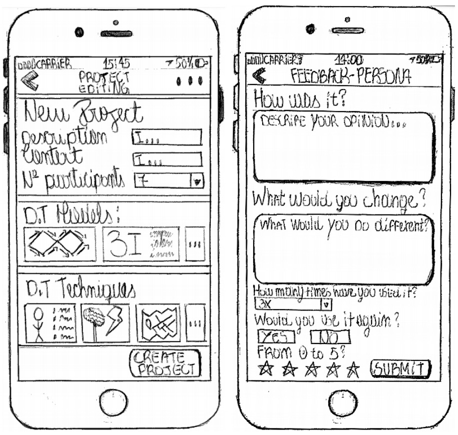
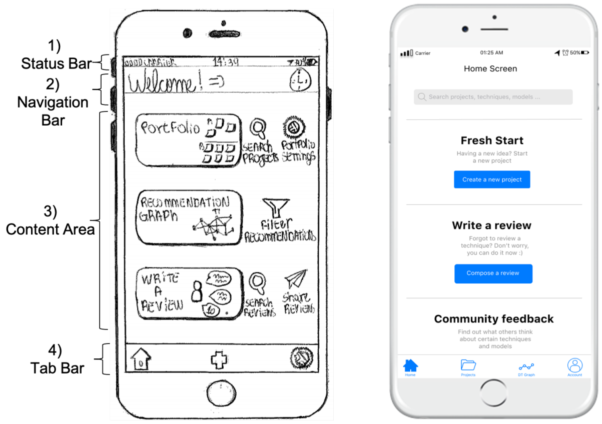


Figure 8. (a) Project creation (b) Feedback

Finally, **Figure 8-(b)** illustrates a screen for capturing user feedback from the use of DT techniques in their software development projects, chosen through the recommendations made by the tool we are proposing in our work. Thus, the user can inform the result of the application of a specific technique (e.g., Persona), filling in fields like what was the experience of use, what techniques they used in combination, how many times they used it, and how to rate it (on a level of stars ranging from 1 (not suitable) to 5 (very appropriate)). This last



(a) Home Screen Low-fidelity prototype
(b) Home Screen High-fidelity prototype

Figure 9. Low Fidelity to High Fidelity Home screen

screen represents another core of our tool: exchange of information and experience among DT professionals, creating an effective and consolidated communication channel and establishing a community environment, especially for those who apply DT in software engineering. Thus, our proposal is not limited to an information guide but defines it as a collaborative environment that enables the exchange of experiences between DT users.

4.1.3 Validation: Early Tool Evaluation

To identify whether we were missing any relevant feature in our tool proposal, we conducted an early evaluation interviewing 5 industry practitioners. Here, to maintain the confidentiality of both the practitioners, we identify them as P1, P2, P3, P4, and P5. We asked: i) how do you select DT techniques? ii) how do you deal with changes in the technique selection during a DT session, if any?, and iii) how would you welcome a tool that recommend you techniques considering your product context and feedback from others?

Regarding the question (i) choice of DT techniques, the participants reported that they select considering the customer's knowledge and feeling, according to P2; that they learn to choose from the experience gained from years of application, and that ends up creating a particular set of techniques by the results that have already been obtained in previous applications, as mentioned by P3. In addition, the techniques can be chosen and determined according to the DT working space, either for understanding the problem, for ideation or the construction of the solution. P4 unveiled it:

- *"We learn to choose techniques over time because each technique has a result. If one wants to understand the problem, one uses certain techniques. One defines ones' own toolkit overtime. Also, if one has to think about solutions, one has these (certain techniques) here that help one get there"*–P4.

On the need to look for other techniques while conducting a DT session (question ii), interviewees responded that situations arise when users/participants are not engaged or do not understand how the technique works and need to change or adapt the technique for more meaningful results. This situation is illustrated by the answers of P3 and P4:

- *"There are situations in DT Workshops that the use of some techniques does not work, so the moderator/designer must choose another technique from their experience. There are also instances when participants find it challenging to use a particular technique, so one needs to use others"*.–P3
- *"It happens that one has to change in the middle of a workshop because the customer does not respond well to a certain activity. In this situation, empathy must be used to understand what is happening and get around the situation. Changing the technique is often helpful in such situation."*–P4

Still, P5 and P2 suggest making combinations between different techniques, observing that with the applied variations, going beyond the pre-established models and keeping the freedom for exploring the Designer's mindset, one gets a more efficient result.

- "*Experiment variations between techniques (a mix of techniques), because this ensures the bias of Designer and not just replicating methods [...]*"–P5
- "*Ah, you (might) test if this technique combines with this (another) technique.*"–P2

Regarding the need for a recommendation tool (question iii), the interviews showed that it will be useful and of great help to professionals who use DT. They suggested that the DT user might input some data such as: *"Do you already have the problem defined?"* to know if the person (client) already has the scope of the problem defined; *"Do you know users?"* – then suggest a particular set of tools.

For P1, the tool should be geared to the DT community by fostering information exchange (feedback, more effective techniques); should have explanations of concepts, such as whether the solution to the problem is developing an mobile app, providing useful Minimum Viable Product (MVP) tips; and should clearly present the techniques to understanding the problem, for ideation, and for creating the solution. Also, it should establish a sense of community, with a forum for users to interact among themselves, going beyond a simple guide, and provide metrics for evaluation (number of users who accessed in a period, user satisfaction when using).

Therefore, we realized through the early evaluation task with 5 professionals that the selection of DT techniques is made considering customer's knowledge and feeling, previous experience, or according to the working space of DT. We have also learned that professionals might change a DT session's selected techniques due to a lack of participant engagement. In addition, the professionals mentioned that a tool that considers the previous experience of the professionals to recommend techniques seems to help with the decision-making of the selection of DT techniques for software development.

4.2 Iteration 2: Requirements Refining and Tool Validation Activities

4.2.1 Solution Design: Requirements Refining

After the early evaluation activity, inspired by the feedback gathered with the Design Thinking practitioners, and based on our defined artifacts (user journeys, blueprints, and low-level prototypes), we performed a requirements refining activity. We transformed the low-level fidelity prototypes into high-level prototypes to further detail each of the previously identified features. For this transformation, we used Figma as a prototyping tool.

Figure 9 presents an example of our low-level (a) to high-level (b) fidelity prototype transformation. We structured all the high-prototypes into 4 basic areas: (1) a status bar, a thin line containing system information; (2) a navigation bar, to show information about the user's current location within the App; (3) a content area, to represent the content, where the general buttons, inputs and other components are positioned in, and finally; (4) a tab bar, located at the bottom of the canvas–used area to draw–, used to show all tool sections, which in our example are: the home screen itself, the projects, the recommendation graph, and the user profile.

As a result, we transformed the 19 low-level into 62 high-level prototypes, as illustrated in **Figure 10**. We started the high-fidelity prototyping process with the creation of an initial screen, drawing the 4 main areas of the screen structure as previously mentioned–the status bar, the navigation bar, the content area, and the tab bar. We used the initial screen as a standard prototype to facilitate the creation of the other screens of the solution. After that, using the initial prototype, we built all the other high level prototypes in the solution. For each defined feature, we created one or more high level prototypes. During this process, we peer-reviewed the drafted high prototypes among the authors, resulting in improved versions. This process allowed us for a better and refined understanding of the specified user requirements.

4.2.2 Validation: Requirements Validation

After the requirements refining activity, we moved to the validation activity in the second iteration. We used the high-level fidelity prototypes generated with Figma as input for designing and running a requirements validation activity. Thus, we organized the validation activity into 2 major tasks: i) validation environment setup using Quant-UX, aiming to design and prepare the environment for validating the specified requirements through prototyping, and ii) feedback col-

Figure 10. Overview of the screens built using Figma

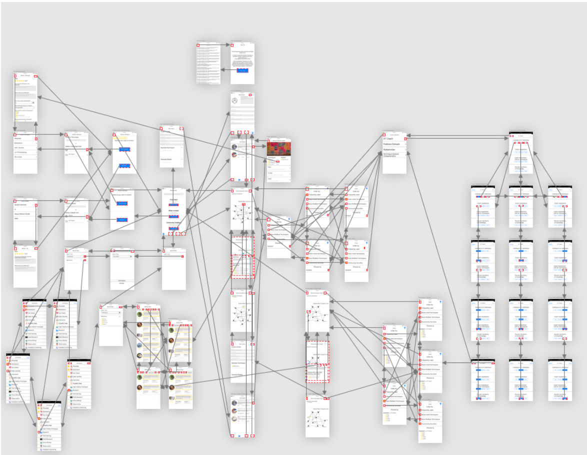


Figure 11. Overview of the screens flow built using Quant-UX

lection with DT practitioners, through a questionnaire-based data collection, aiming to validate our DT techniques recommendation tool for future implementation. As mentioned, the questionnaire was based on Technology Acceptance Model (TAM) considering the Perceived Ease of Use and Perceived Usefulness factors.

Validation environment setup using Quant-UX

We used Quant-UX to simulate the interaction of our high-level fidelity prototypes as a means to validate the requirements with potential users, i.e, DT practitioners. For validation purposes, Quant-UX includes features such as screen recordings, A/B testing, and QR Codes to share the solution with users and test the solution. Quant-UX also allows the creation of user interface flows, transforming the prototypes into interactive ones, providing a functional perspective of the solution. It also includes features like: (a) heatmaps, highlighting the points of user interaction on the screen; (b) user journey, showing which prototypes are executed by the user and in what flow, helping to discover usability problems, and (c) test evaluation, a statistical report describing useful data about the user interactions, like scroll visibility, number of views, and dwell time.

Using Quant-UX, we firstly imported the 62 high-level fidelity prototypes from Figma as images. Next, we created the interactions between the prototypes, defining the logical entries for generating user flows. **Figure 11** illustrates an

Table 2. Participants of the tool's validation step

Participant ID	Role	DT Years*
V1	Head of Innovation	10
V2	Innovation Manager	3
V3	Engineer Support	5
V4	UX Designer	5
V5	Project Manager	5
V6	Software Engineer	3
V7	Designer	4

*Years of Experience using DT in software development.

overview from the logic flow. The arrows portray the paths between an element and the screen that is summoned when that element is pressed.

Feedback collection with DT practitioners

Once the setup of the requirements' validation activity was completed, we conducted the study with the 7 participants that agreed to contribute. **Table 2** shows the participants (named validators here), identified as V1 to V7, their roles and their DT's years of experiences.

We sent to the practitioners a 4 minutes pre-recorded video explaining our research goals and introducing in general words how the tool will work. The video did not contain explanations about how the practitioners should use the features of the proposed tool, aiming to avoid feedback bias. We also sent the link for the prototypes created using Quant-UX, and the TAM-based questionnaire link.

Table 3 shows the questionnaire we created for collecting validation feedback based on TAM model and Likert scale (Totally agree, Strongly agree, Partially agree, Partially disagree, Strongly disagree, Totally disagree). We posed statements considering Perceived Ease of Use (PEU) and Perceived Usefulness (PU) factors. Also, we posed 3 open questions (OP) for collecting if the practitioners would use our recommendation tool, which are the positive features they perceived, and which are the features we should improve in our tool proposal. The questionnaire was peer reviewed by a senior researcher who has conducted several studies using TAM over the years, and piloted with a graduate student who also works in industry.

Figure 12 shows the results of the TAM model, based on the Perceived Ease of Use (left) and the Perceived Usefulness (right) factors of the recommendation tool, respectively.

- **Perceived Ease of Use:** According to the DT practitioners who participated in our validation activity, the DT techniques recommendation tool can be considered an easy-to-use tool. With different agreement levels, the participants considered it easy to learn how to use the recommendation tool (PEU #1), acquire the tool's ability (PEU #3), remember how to request DT techniques recommendation (PEU #4), and in general, the tool was easy to use (PEU #5). Only for the statement related to how easy it was to request recommendations for DT techniques (PEU #2), 1 DT practitioner partially disagree with the ease of using the tool.
- **Perceived Usefulness:** Regarding to the usefulness of the recommendation tool, most of the DT practitioners totally agreed that the tool allows: to create projects and

Table 3. Validation Questionnaire based on TAM model

Perceived Ease of Use statements for the recommendation tool:	
PEU #1	It was easy to learn how to use the DT techniques recommendation tool
PEU #2	I was able to use the recommendation tool to request recommendations for DT techniques
PEU #3	It was easy to acquire the ability to use the recommendation tool
PEU #4	It was easy to remember how to request DT techniques recommendation
PEU #5	I think the recommendation tool is an easy to use tool
Perceived Usefulness statements for the recommendation tool:	
PU #1	The tool allows to create projects and include DT techniques
PU #2	The tool allows to require recommendations of DT techniques
PU #3	The tool allows to evaluate the DT techniques used
PU #4	The tool allows to view information about the related techniques in the graph
PU #5	The tool allows to view other users' comments on DT techniques
PU #6	I consider the recommendation tool useful for selecting DT techniques
Open questions about the use of the recommendation tool prototypes:	
OQ1	Would you use this tool for selecting DT techniques in software development? Why?
OQ2	Which features of the tool do you consider positive for the selection of DT techniques?
OQ3	Which features of the tool do you think need to be improved for the selection of DT techniques?

Figure 12. Perceived Ease of Use (left) and Perceived Usefulness (right) of the recommendation tool

include DT techniques (PU #1), to require recommendations of techniques (PU #2), to evaluate the DT techniques used (PU #3), to view information about the related techniques in the graph (PU #4), and to view users' comments on DT techniques (PU #5). The DT practitioners also agree that the recommendation tool is useful for selecting DT techniques (PU #6).

Regarding to the questions about the features provided by the tool, the DT practitioners answered the following:

OQ1: Would you use this tool for selecting DT techniques for software development? Why?, the DT practitioners pointed out they would used the recommendation tool because it provides data about the experiences from professionals who used the tool, allowing to better know the practices around DT techniques:

- *"The tool will become a data pool with several experiments and combined models of tools. This is cool to bring inspiration and best practices."*–V1;
- *"We can create shortcuts through the experience of other users"*–V2;
- *"This tool will help my choice of techniques to the point that I can learn about other users' experience with certain techniques"*–V7.

The participants considered the tool appropriate for selecting Design Thinking techniques and for engaging DT practitioners during DT sessions:

- *"The tool would be very useful to assist when I need to think about DT techniques to engage participants in the DT session"*–V5.

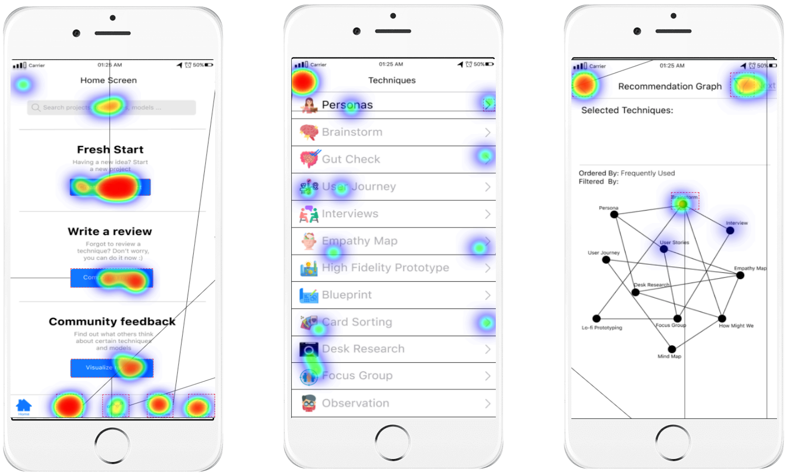
They also considered the tool useful since it provides information to select a DT technique according to how much time is required to use it, and also it was considered useful to that professionals who do not have any DT experience:

- *"(The tool) allows me to see how much time is required to apply a DT technique, because in order to select the techniques I always consider the time available I have to apply DT."*–V4;
- *"I found it very useful for those who have no previous experience with DT"*–V6.

Regarding to the question *OQ2: Which features of the tool do you consider positive for the selection of DT techniques?*, 2 participants considered the recommendation graph as a positive feature:

- *"The Recommendation graph [...] for a quick consultation (of the techniques) is amazing, especially for more experienced professionals"*–V1.
- *"The suggested graph is an interesting feature. By clicking on it, it is possible to know what other techniques are related. The technique recommendation is great, it's like we have someone helping us to choose techniques"*–V7.

Other participants (V2, V3, V4, and V5) mentioned the reviews of the techniques as a positive feature since it provides



(a) Home screen (b) Techniques selection (c) Graph recommendation

Figure 13. Heatmaps of the use of the tool's prototypes

information to the professional for choosing the appropriate DT technique. For instance, one participant mentioned:

- *"(The tool) provides a view of the comments of other participants, which allows me to see what the experience with the techniques was like"*–V4.

In question *OQ3: Which features of the tool do you think need to be improved for the selection of DT techniques?*, the DT practitioners pointed out improvements for some features of the proposed tool, such as:

i) techniques visualization: *"To improve visualization of techniques and details of techniques."*–V5;
ii) reviews presentation: *"The organization by tabs could be changed by a combo box"*–V1;
iii) project creation including DT techniques: *"I think that when we describe a project, we have to start with the selected techniques. I thought the prototype didn't give me this option to see if it worked"*–V3.

Still in the validation activity, using Quant-UX we captured heatmaps of the prototypes' areas that the users clicked. These heatmaps helped us to analyze the user experience and to identify improvements to the proposed tool's screens. **Figure 13** shows examples of the heatmaps for the Home screen (a), for the technique selection screen (b), and for the graph recommendation screen (c). In the Home screen (a), the participants clicked on the main areas, such as buttons and bottom tab bar, which allowed them to access the features of the tool. In the Graph recommendation screen (c), the participants were able to access the information of the technique showed in the graph, and also they were able to access the filtering features. On the other hand, the heatmap of the Techniques selection screen (b) shows that some users had some difficulty to select a technique to see its information since there are clicks over the technique's name, but the area on the screen for this purpose is the icon placed on the right side of the techniques' name. Thus, this result indicates that its necessary to highlight the area to be clicked.

5 Discussion

The tool proposed in this paper seeks to recommend Design Thinking techniques to professionals who are integrating DT into software development, specially due the increase of the use of DT in requirements engineering activities. Such proposal aims to attend a gap we found through a previous systematic review and a survey study regarding the challenge for selecting DT techniques. The tool is an artifact proposed as result from DSR-based methodology for what we posed the following research question–*How can we support software development professionals to select DT (set of) techniques during requirement engineering?*

We list some insights as a set of initial takeaways, obtained through a requirement elicitation and an early evaluation activities with DT practitioners:

- The tool seeks to provide relevant information on DT techniques, going beyond the existing user's toolkit. The tool must allow the selection of other similar techniques through its recommendation system, using a technique relationship graph that should take into account a set of items for the recommendation such as previous use of a certain technique, users feedback, product context, and project characteristics.
- The tool should be valuable and able to assist both onboarding novice users as well as expert ones during their DT sessions in software development. Both profiles found the tool idea useful.
- The tool should represent an innovative solution presenting a recommendation tool associated with a community building environment through feedback. Thus, the industry practitioners might collaborate with their community fostering the improvement of DT in software development.

Based on these insights, we conducted 2 Solution Design and Solution Validation iterations following the DSR framework presented in Runeson et al. (2020). As an extension of our previous study Parizi et al. (2020b), we included a requirement refining activity using high-fidelity prototypes and a requirement validation activity with DT practitioners.

We validated with industry professionals that our proposal is easy to use, it is useful, and has potential to support the professionals through the recommendation features. We also collected with the validation study the need of explore the sense of community as a future direction of our tool, allowing the users to retro-feed the recommendation system with their experiences on the use of DT techniques, fostering an opportunity to learn with other professionals and to establish some shortcuts when selecting DT techniques.

The results of the early tool evaluation and the requirements validation activities also indicated that our DSR-based research presents theoretical contribution (Relevance constituent). We figured out that IT professionals select DT techniques based on the investigation of the customer's knowledge and feeling, on the learning with previous experience on the application of DT tools, on the creation of an own set of DT techniques, or based on the analysis of the DT working space and techniques related to it. We also discovered that that the lack of participant's engagement, the lack of participant's understanding of how the technique works, or because of low meaningful results the professionals might change the selected techniques during a DT session.

Our research also presents practical contributions. Starting with the proposal of a DT techniques recommendation system, the professionals argued that it helps professionals on the selection of DT techniques and to experiment variations between a mix of techniques. The tool (the artifact produced) also keeps the freedom of choice of DT techniques according to the dynamic nature of DT, provides inspirations and knowledge of the best practices about DT techniques and creates shortcuts through the experience of other users. The tool also provides information about the relation of techniques.

We confirmed the contributions of our research by assessing the innovation potential of the proposed tool in (Parizi et al., 2020a). The study presents a comparison of our proposed tool with other tools listed from the literature. We used the DESMET feature analysis method. The comparison results showed that our recommendation system advances the state of practice by providing the collaboration among IT professionals, embracing their experience as an element to recommend techniques. Likewise, our work contributes to the state-of-the-art. The recommendation of DT techniques from collaborative experiences and the understanding of the decision-making of professionals who select the techniques can contribute to Software Engineering area.

However, it is possible to see that there is a universe not yet explored, that is, we can still consider a lot of other features and technologies to apply in this tool, such as decision support methods, recommendation systems approaches, artificial intelligence algorithms, multi-platform system, and others. In addition, once the tool is working, other application opportunities could raise up and open new rooms of research, such as its use in other phases of software development beyond RE itself, keeping the dynamic and flexible nature of DT. Thus, we seek to ensure that the software industry, from those responsible for applying DT sessions, perceive in this tool a guide of recommendations that enables effective gains in the software development process.

6 Threats to Validity

This section clarifies the threats inherent to our study that is qualitative by nature, and it also shows our actions aiming to mitigate them.

Problem understanding and solution definition using DT: the use of DT as a approach to understand the problem and define a solution is an activity that explores the diversity of ideas and creativity. The solutions proposed in a DT session consider the participants' worldview. Although we have invited professionals who have used DT in software development, the proposed solution represents the ideas of those who accepted to participate in the session. Therefore, the proposed solution could not be represent the needs of whole universe of professionals. To mitigate this threat, we based our research on the DSR method and through 2 iterations we empirically evaluated the proposed solution.

Defined RE Artifacts: The DT session's moderator introduced only 2 Personas (Joano and Sindi) to represent stereotypes of potential users. However, each persona aims to group users who face the challenge of selecting DT techniques from different perspectives. Literature has shown that Personas serve to connect users with similar profiles. We also generated other RE artifacts with the elicitation and refining of requirements (Blueprint, User Journeys, and Prototypes). They do not represent the full set of existing artifacts that we could use for requirements elicitation and specification (e.g., scenarios, storyboards, etc.). However, we were able to collect and specify the requirements, moving forward in proposing a DT techniques recommendation system.

Feedback from professionals: We collected data with different professionals on 2 different activities. The feedback given by professionals represents the individual point of view for our proposal and might not be generalized. To mitigate this, we interviewed 5 professionals in the early evaluation step, and we sent the invitation to 80 professionals for the tool's requirement validation step. Seven professionals accepted to participate in the requirements validation. Thus, although a more significant number of participants allows a more expressive capture of professionals' needs, our sample also represents professionals with different needs. Another threat to validity regarding the professionals' feedback is the introduction of our research goals and tool's features in the Tool's requirements validation step. Although we have sent a pre-recorded video to introduce the goals and features, which may have biased the feedback given by the participants, we mitigate this threat by not showing to the participants how they should use the features proposed by the tool.

Results: The results represent our interpretation from the feedback collected in the Validation activity (early tool evaluation in iteration 1 and tool's requirements validation in iteration 2). To mitigate the interpretation bias, we held meetings among the authors to discuss the artifacts produced and to analyze the feedback captured from professionals.

7 Concluding Remarks

This paper proposes a tool for recommendations of Design Thinking Techniques in the context of software requirements.

Inspired on the results of previous exploratory studies and aiming to further the understanding of the need to propose such a tool and identify a solution, we designed and conducted a study based on DSR methodology. Moving from a problem understanding activity to 2 iterations of Solution Design and Validation Approach, we performed a meta-DT session, followed by a requirements specification step, an interview-based early evaluation with industry practitioners, a requirements refining step through prototyping and, finally, a requirements evaluation step.

As a result of the first iteration, through a Meta-DT session we identified 4 macro features to be present in our recommendation tool, namely: i) qualified decision making, ii) prediction-based recommendation, iii) techniques presentation, and iv) dynamic visualization of recommendations. These macro features were broke-down into smaller features and detailed in their use (User Journeys) and functionality (Service Blueprints). In addition, they were prototyped in a second round to refine their understanding and serve as guidance for the tool early evaluation with industry practitioners.

Results of the early evaluation with 5 professionals revealed that: i) previous experience is key to select techniques, ii) new techniques might be needed during a DT session, and iii) a recommendation tool could be useful to help with the decision process and addressing changes, mainly when considering context and historical use from others.

After the early evaluation, moving to the next iteration of the DSR framework, we refined the requirements transforming the low-level prototypes into high-level prototypes. This step allowed us to generate a refined set of requirements and a refined version of the tool screens. Then, we prepared and run a validation study with 7 DT practitioners with experience on the use of DT in software development. The results of the validation step showed that the tool was accepted by DT practitioners with high experience on the use of DT in software development, considering the Perceived Ease of Use and the Perceived Usefulness factors based on TAM model. Also, through the validation study using Quant-UX, we confirmed what features to to implement in the tool and we were able to identify improvements to the tool according to the DT practitioners' perceptions.

Highlighting the extension activities (2 iterations of Design and Validations activities of the DSR framework), we validated the potential of our tool and captured insights to explore the sense of community, supporting the professionals to learn from the experiences of other professionals on the use of DT techniques in software development. To explore this mechanism, we planned as our next research phases the characterization of the professionals' decision-making process for selecting DT techniques, and the proposition of a decision model from the use of the recommendation system for selecting DT techniques (see Parizi and Marczak (2020)).

We aim to deepen this evaluation within the own use of the tool after it has been implemented. We are currently developing a working prototype that should soon be made available for experimental use by industry practitioners. We intend to use the tool to collect data on how people are selecting the techniques and, alongside with the feedback and the project context features, will retro-feed the recommendation algorithm until its stabilization. Other items might also be taken into account for defining how the recommendation should work. For instance, the work by Ignacio Crispim and Benitti (2020) propose a set of characteristics for recommending a requirements elicitation technique. This will complete this first phase of the tool proposal before we move to the refinement of the community building to engage users, which will both compose the tool design second phase.

Acknowledgements

We thank PUCRS BPA 2019, 2020 and 2021 projects (Programa de Bolsas Pesquisa Alunos da PUCRS/Chamada Geral 1/2019, 1/2020 and 1/2021) and CNPq PIBIC 2019/2020 project (Programa Institucional de Bolsas de Iniciação Científica do CNPq) for the undergraduate research assistant scholarships. Tayana Conte thanks CNPq (Grant 314174/2020-6) and Sabrina Marczak thanks CNPq (Grant 307177/2018-1). Rafael Parizi thanks CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil) for the financial support (Code 001).

References

Albaum, G. (1997). The Likert Scale Revisited. *Market Research Society Journal*, 39(2):1–21.

Brenner, W., Uebernickel, F., and Abrell, T. (2016). Design Thinking as Mindset, Process, and Toolbox. In *Design Thinking for Innovation*, pages 3–21. Springer.

Brown, T. (2008). Design Thinking. *Harvard Business Review*, 86:84–95.

Crispim, R. and Benitti, F. (2020). Improving the Selection of Requirements Elicitation Techniques: a Faceted Guide. In *Proc. of the Workshop on Requirements Engineering*, pages 1–14, São José dos Campos, Brazil. PUC-RIO.

Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly*, 13(3):319–340.

Dias, G. A., Silva, P. M. d., Delfino Jr., J. B., and Almeida, J. R. d. (2011). Technology Acceptance Model (TAM): Avaliando a Aceitação Tecnológica do Open Journal Systems (OJS). *Informação & Sociedade: Estudos*, 21(2).

Dobrigkeit, F. and de Paula, D. (2019). Design Thinking in Practice: Understanding Manifestations of Design Thinking in Software Engineering. In *Proceedings of the European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, page 1059–1069, Tallinn, Estonia. ACM.

Docherty, C. (2017). Perspectives on Design Thinking for Social Innovation. *Design Journal*, 20(6):719–724.

Ferreira, B., Silva, W., Oliveira, E., and Conte, T. (2015). Designing Personas with Empathy Map. In *Proc. of the Int'l Conference on Software Engineering and Knowledge Engineering*, pages 501–505, Pittsburgh, USA. Seke.

Figma (2015). Figma prototyping Tool. `https://www.figma.com`. Last checked on Nov 20, 2020.

Hehn, J., Mendez, D., Uebernickel, F., Brenner, W., and Broy, M. (2020). On Integrating Design Thinking for Human-Centered Requirements Engineering. *IEEE Software*, 37(2):25–31.

Hehn, J. and Uebernickel, F. (2018). The Use of Design Thinking for Requirements Engineering: An Ongoing Case Study in the Field of Innovative Software-Intensive Systems. In *Proc. of the Int'l Requirements Engineering Conference*, pages 400–405, Banff, Canada. IEEE.

Hehn, J., Uebernickel, F., and Fernández, D. M. (2018). DT4RE: Design Thinking for Requirements Engineering: A Tutorial on Human-Centered and Structured Requirements Elicitation. In *Proc. of Int'l Requirements Engineering Conference*, pages 504–505, Alberta, Canada. IEEE.

Hevner, A. R. (2007). A Three Cycle View of Design Science Research. *Scandinavian Journal of Information Systems*, 19(2):87–92.

Hiremath, M. and Sathiyam, V. (2013). Fast Train to DT: a Practical Guide to Coach Design Thinking in Software Industry. In *Proceedings of the International Federation for Information Processing*, pages 780–787, Cape Town, South Africa. Springer.

IDEO (2020). IDEO Design Kit. `https://designkit.org/methods`. Last checked on Nov 20, 2020.

Laitenberger, O. and Dreyer, H. M. (1998). Evaluating the Usefulness and the Ease of Use of a Web-Based Inspection Data Collection Tool. In *Proc. of the Int'l Software Metrics Symposium*, pages 122–131, Bethesda, USA. IEEE.

Lewrick, M., Link, P., and Leifer, L. (2020). *The Design Thinking Toolbox: A Guide to Mastering the Most Popular and Valuable Innovation Methods*. John Wiley & Sons.

Liedtka, J. (2015). Perspective: Linking Design Thinking With Innovation Outcomes through Cognitive Bias Reduction. *Journal of Product Innovation Management*, 32(6):925–938.

Lindberg, T., Meinel, C., and Wagner, R. (2011). *Design Thinking: A Fruitful Concept for IT Development?*, volume 1, pages 3–18. Springer, Heidelberg, Germany.

Lucena, P., Braz, A., Chicoria, A., and Tizzei, L. (2016). IBM Design Thinking Software Development Framework. In *Proceedings of the Brazilian Workshop on Agile Methods*, pages 98–109, Curitiba, Brazil. Springer.

Luma, I. (2012). *Innovating for People: Handbook of Human-Centered Design Methods*, volume 1. LUMA Institute, LLC.

Parizi, R. and Marczak, S. (2020). A Context-based Recommendation Model for Design Thinking: Techniques Selection in Software Development. In *Proc. of the Brazilian Symposium on Software Engineering*, pages 114–122, Natal, Brazil. SBC.

Parizi, R., Moreira, M., Couto, I., Marczak, S., and Conte, T. (2020a). A Design Thinking Techniques Recommendation Tool: An Initial and On-Going Proposal. In *Proceedings of the Brazilian Symposium on Software Quality*, São Luis, Brazil. ACM.

Parizi, R., Moreira da Silva, M., de Souza Couto, I., Pavin Trindade, K., Plautz, M., Marczak, S., Conte, T., and Candello, H. (2020b). Design Thinking in Software Requirements: What Techniques to Use? A Proposal for a Recommendation Tool. In *Proceedings of the Ibero-American Conference-American on Software-American Engineering*, page 14, Curitiba, Brazil. Curran Associates.

Prestes, M. (2020). Estudo Exploratório sobre Design Thinking no Desenvolvimento de Software. Master's thesis, Escola Politécnica, PUCRS. Porto Alegre, Brazil.

Prestes, M., Parizi, R., Marczak, S., and Conte, T. (2020). On the Use of Design Thinking: A Survey of the Brazilian Agile Software Development Community. In *Proceedings of the International Conference on Agile Software Development*, pages 73–86, Copenhagen, Denmark. Springer.

Przybilla, L., Schreieck, M., Klinker, K., Pflügler, C., Wiesche, M., and Krcmar, H. (2018). Combining Design Thinking and Agile Development to Master Highly Innovative IT Projects. In *Proceedings of the Project Management and Procedure Models conference*, pages 113–124. Gesellschaft für Informatik, Bonn.

Quant-UX (2020). QuantUX prototyping Tool. `https://www.quant-ux.com`. Last checked on Nov 20, 2020.

Rozante de Paula, T., Santana Amancio, T., and Nonato Flores, J. A. (2020). Design Thinking in Industry. *IEEE Software*, 37(2):49–51.

Runeson, P., Engström, E., and Storey, M.-A. (2020). *The Design Science Paradigm as a Frame for Empirical Software Engineering*, pages 127–147. Springer.

Seidel, V. P. and Fixson, S. K. (2013). Adopting Design Thinking in Novice Multidisciplinary Teams: The Application and Limits of Design Methods and Reflexive Practices. *Journal of Product Innov. Management*, 30:19–33.

Sköldberg, U., Woodilla, J., and Çetinkaya, M. (2013). Design Thinking: Past, Present and Possible Futures. *Creativity and Innovation Management*, 22(2):121–146.

Souza, A. F., Ferreira, B., Valentim, N. M. C., Correa, L., Marczak, S., and Conte, T. (2020). Supporting the Teaching of Design Thinking Techniques for Requirements Elicitation Through a Recommendation Tool. *IET Software*, 14(6):693–701.

Souza, A. F. B. d., Ferreira, B. M., and Conte, T. (2017). Applying Design Thinking in Software Engineering: A Systematic Mapping. In *Proc. of the Ibero-American Conference on Software Engineering*, pages 1–14, Buenos Aires, Argentina. Curran Associates.

Stanford d.School (2019). Design Thinking - Bootcamp Bootleg. `https://dschool.stanford.edu/resources/the-bootcamp-bootleg`. Last checked on Nov 20, 2020.

Tonkinwise, C. (2011). A Taste for Practices: Unrepressing Style in Design Thinking. *Design Studies*, 32(6):533–545. Interpreting Design Thinking.

Vianna, M. (2012). *Design Thinking: Business Inovation*. MJV Technology & Innovation.

Wieringa, R. (2014). *Design Science Methodology for Information Systems and Software Engineering*. Springer.