


Design Thinking Use in Agile Software Projects: Software Developers' Perception

Edna Dias Canedo¹ ^a, Ana Carolina Dos Santos Pergentino¹, Angelica Toffano Seidel Calazans², Frederico Viana Almeida¹, Pedro Henrique Teixeira Costa¹ and Fernanda Lima¹

¹Department of Computer Science, University of Brasília (UnB), P.O. Box 4466, Brasília – DF, CEP 70910-900, Brazil

²University Center – UniCEUB, Brasília – DF, Brazil

Keywords: Design Thinking, Agile Teams, Requirement Engineering, Software Development Process, Software Developers' Perception.

Abstract: Only recently the application of the Design Thinking (DT) in agile development has begun to be researched. Thus, this research aims to analyze information collected from agile software developers concerning their perceptions about applying DT methods and tools in agile development. An online survey was submitted to agile teams from Brazilian software organizations and a total of 59 answers were obtained. Results reveal that the most commonly used techniques during the development process are brainstorming and prototyping. In requirement elicitation, the techniques most used by the practitioners are interviews with users and prototyping. The study concludes that practitioners are already using many DT techniques, tools and methods in software development activities. Furthermore, they acknowledge that DT practice in requirement elicitation could contribute to delivering product quality to the end-user since design thinking techniques could prevent failure to understand requirements prior to implementation.

1 INTRODUCTION

Requirement engineering (RE) has developed new approaches through recent years regarding system needs. It is acknowledged that there are many manuals and reference guides concerning how to manage requirements in the process of software development. Although, according to Femmer and Vogelsang (Femmer and Vogelsang, 2019) it is perceptible in the practical field that there are some problems in following normative rules, such as no contextualization, lack of solutions (incompleteness) or not being clear why certain criteria do not fit in the good quality ranking. Dealing with guidance insufficiencies, there are simple workarounds to manage a lack of applicability of overall normative rules. It is very common to solve problems quickly using face-to-face communication, which also facilitates the understanding of needs in real-time. But these are not possible methods in all software development processes (Femmer and Vogelsang, 2019).

According to Knauss (Knauss, 2019), in a wide

project, with multiple agile teams, it is noticeable that there are no means of integration and synchronization enough to suppress all problems that may appear during development. Iteration, although helped substantially the performance enhancement in product delivery, also causes constant changing features, which influences in original system requirement. System requirements are detailed and could be modified over development. Agile development is dynamic and demands flexibility during all process, which leads many developers to criticize or ignore requirements (Knauss, 2019), (Masood et al., 2017), (Kusters et al., 2017).

When requirements are not followed, and there are constant changes during development, it becomes difficult to maintain also the knowledge about the code. And so, the challenge of passing this knowledge enters the list of difficulties that may delay development (Knauss, 2019). But even recognizing that there are eventually problems through continuous development, how do programmers work and feel using the methodology? Are they satisfied with the hierarchy that was supposed to be more horizontal but can appear as the same vertical pressure as always? Which

^a  <https://orcid.org/0000-0002-2159-339X>

are the workarounds daily used to deal with the fast movement of needs in a project?

The Design Thinking practiced nowadays has begun in business to improve business innovative processes. DT is a multidisciplinary and human centered methodology that focuses on a target group's needs. It is possible because this approach is design oriented. Consequently, DT looks forward to the people's experience and well-being (Hehn et al., 2019). Just as innovation itself, the DT approach can be applied in multiple sectors because of its great effectiveness and adaptability and also its high potential. Nevertheless, only recently, this methodology is being widely employed in software products developing with a global scope (Hehn and Uebernickel, 2018). However the same fact is not a completely valid affirmation in Brazil since initiatives in this area have been slowly promoted in universities and market (de Carvalho Souza and Silva, 2015),(de Almeida et al., 2018). Design Thinking has high adaptability, and Software Engineering can seize it especially during software requirements elicitation, which is part of the software development process as well as the DT (de Carvalho Souza and Silva, 2015).

In this paper, we examined how agile software development teams use DT techniques during development process activities and which benefits and facilities were discovered over activity execution with DT. Besides, we collected the perception of agile teams' practitioners, regarding their satisfaction over DT use along stages of the development process. Thus, the main contributions of the paper are: 1. Identification of the Design Thinking phase most used by agile software development teams; 2. Identification of most used techniques in requirements elicitation; 3. Perception of agile teams regarding the advantages of using DT along with software development projects; 4. Evaluation of the use of the requirement specification document during functionalities implementation.

2 BACKGROUND AND RELATED WORKS

To examine difficulties related to the methodology, we must recapture the concept of Design Thinking (DT). DT was formulated to develop innovative solutions with a human-centered approach (Brown and Katz, 2009). DT has an iterative profile and intends to develop and test potential solutions, until attaining the ideal solution. DT is a methodology indicated when complex problems that cannot be properly defined need to be solved, seeking to understand human needs bonded to the problem, placing the ques-

tion from a human point of view, conceiving ideas through brainstorming and applying them in prototypes and tests. As a more practical approach, it is possible to save development resources, because the project is planned and tested before the implementation so that efficiently it will not result in unnecessary rework (Plattner et al., 2009).

Constant iteration in the Design Thinking process follows the agile development concept of acting in feedback cycles to increase efficiency. The delivered product after each phase will not be necessarily done, the team must comprehend the product impact and proceed to develop it cyclically. The intention is to reach the ideal, a product closest to the need of the final user.

In order to reach it there will be many changes in what was first defined: empathy will increase due to what is observed in interaction between user and system, problems will be reframed to include new discovers in empathic comprehension, ideas will adapt to solve those new problems, prototypes will readapt to verify if those new ideas indeed are functional, and tests will point out efficiency from all process that will renew themselves consecutively (Pereira and de FSM Russo, 2018). According to Liedtka and Ogilvie (Liedtka and Ogilvie, 2011), DT is associable with 4 basic questions: 1. What is? - explores reality in present; 2. What if? - looks toward the future; 3. What wows? - what impresses helps decision making; 4. What works? - what inserts us into the market. There are 10 essential tools to answer these questions, that are necessary to create new possibilities and reduce risks while having to handle uncertainty from growth and innovation (Liedtka and Ogilvie, 2011), (Moggridge, 2007):

1. **Visualization:** trying to set the situation from a visual perspective, to naturally materialize possibilities;
2. **Journey Mapping:** perform navigation from the consumer's point of view;
3. **Value Chain Analysis:** a chain value is a set of activities to be executed to deliver a valuable product to the market (Urbig and Verlage, 2003). It is essential to evaluate the user's chain value to define if the product is valuable;
4. **Mind Mapping:** a tool to envision thoughts, that can be the problem or ideas, and their connections. It starts with a core term and the association of ideas connected to that term (Moggridge, 2007);
5. **Brainstorming:** session of solution creation, it occurs in the ideation phase, where diverse ideas rise to solve the problem defined;

6. **Concept Development:** attach an alternative solution compatible with the innovation components.
7. **Assumption Testing:** isolating and testing the key assumptions that will drive the success or failure of a concept;
8. **Rapid Prototyping:** expressing a new concept in a tangible form for exploration, testing, and refinement;
9. **Customer Co-creation:** enrolling customers to participate in creating the solution that best meets their needs;
10. **Learning Launch:** creating an affordable experiment that lets customers experience the new solution over an extended period of time, to test key assumptions with market data.

Complementing the list, we present some of the tools most applied nowadays when employing the DT approach. The human-centered perspective is widely seen in all tools, it is noticeable the search to capture the whole story, creating a product that fits in the user's context, not only delivers functionality but brings facilities to the practical experience as a whole. The tools are (Schwaber, 2005; Bitner et al., 2008; Morelli and Tollestrup, 2006; Long, 2009; Bittner and Shoury, 2019; Bitner et al., 2008):

1. **Sprint:** originated in Scrum, an Agile Development methodology in which the project is separated into cycles. Each cycle is a Sprint, that represents a period throughout which a set of activities should be executed. It is also the time interval defined to iteration: at the end of each Sprint, a meeting occurs intending to set feedback and plan the next Sprint (Schwaber, 2005).
2. **Blueprint Service:** is a graphic way to describe the service processes. It is a tool composed of all sub-process that define interactions and it describes characteristics of the service so detailed that is possible to implement and maintain it (Bitner et al., 2008).
3. **Story Telling:** is about understanding all ways to execute an action or service to be built. It is an analytical phase that allows a realistic event narration, highlighting main problems, interaction errors and emotional factors regarding the system (Morelli and Tollestrup, 2006).
4. **Storyboard:** is about representing use cases through a sequential narrative, allowing better visualization of a story, as a visual tool to support storytelling.

5. **Personas:** refer to a technique of profiling potential users through observation. Fictional characters are created to represent an existing social group, and so their point of view builds a storytelling (Long, 2009).
6. **Empathy Map:** is a creativity method useful to summarize all staff considerations, recognize clients' needs and receive new customers perceptions (Bittner and Shoury, 2019).
7. **Insight Cards:** after immersion insights appear. Cards are used to describe data collected in the research, during these insights.
8. **Pitch:** is a fast presentation of the business proposition to the potential customer or partner, quickly made in 2 to 3 minutes. It is a common way to connect investors and entrepreneurs.
9. **CSD Matrix:** composed by Certainties, Supposition, and Doubts, the CSD matrix gathers all different team members' perspectives, by organizing in these three sections important data to develop, avoiding discussion and centering all possible obstacles.
10. **Prototype:** is not only a tool but considered a whole stage in the Design Thinking process. Projecting a visual preview of systems and their interactions allows visualization, essential to the Design Thinking process.

2.1 Design Thinking Methods in Requirement Elicitation

The software community noticed over the past years how the world is constantly changing, quickly developing and demanding a growing value in acquisitions, and how this scenario applies to the software field. Traditional requirement engineering, that in its majority includes requirements elicitation, analyses and negotiation, specification and validation stages, is already not enough to suppress these demands. Now a context immersion is required (Vetterli et al., 2013a). A conflict exists regarding the way Requirement Engineering and agile software development define the involvement among the user.

The agile software development tends to a significant effort in the software implementation phase, demands user involvement along the development process instead of having rigorous documentation, that is how requirement engineering intends to decrease this involvement after the initial mapping. Despite the agile development oriented by users' descriptions, its capture is from a perspective of functional and non-functional requirements, even with user's orientation

and qualified professionals in team, it is, however, complex to capture what is necessary entirely. Confronted with this dilemma, the search to discover a form to bring agile features inside the requirement engineering universe, which includes the elicitation phase, besides suppressing the own challenges that already exist in the methodology (Vetterli et al., 2013a).

Some researches denote ways to solve this methodology use problem: Design Thinking with an initial practice in requirement engineering, elicitation and prototyping, and support user engagement, as agile methodology defends (Vetterli et al., 2013a). The connection between DT and agile methodology is so evident, that literature highlights their similarity, as in (Higuchi and Nakano, 2017). A project management model that covers the whole software development process, through the Design Thinking and agile methodology approach combination can be used as a requirement elicitation method in agile methodology, mainly in applications that require creativity (Higuchi and Nakano, 2017). Both Design Thinking and agile software development methodology offer competitive advantages, in product differentiation and cost estimative efficiency (Higuchi and Nakano, 2017). Besides, both approaches contain commonalities, that reinforce their use jointly (Higuchi and Nakano, 2017).

Hehn et al. (Hehn et al., 2018) present a tutorial to create an understanding about what Design Thinking is and how it can be integrated with existing Requirements Engineering practices. Hehn et al. (Hehn et al., 2019) suggest three approaches for tailoring and integrating Design Thinking and Requirements Engineering with complementary synergies. Canedo and da Costa (Canedo and da Costa, 2018) present an experience report using the technique of Design Thinking applied in the modernization of two real systems.

3 STUDY SETTINGS

The main goal of this research is to build a broad comprehension of how practitioners use the design thinking tools, as well as to characterize how they notice their significance along the software development process. Here we focus on both public and private organizations. We conducted a study to investigate the following research questions:

RQ.1: What are the Design Thinking techniques used by the organization during the software development process?

RQ.2: What are the Design Thinking techniques used in requirement elicitation in a software development project?

RQ.3: What phase in Design Thinking is widely adopted in organizations along the development process?

RQ.4: What tools are mostly used by practitioners in the software development process?

RQ.5: What are the obtained results with design thinking use inside organizations as a support method in real software development projects?

3.1 Developer Survey

To answer these research questions, we used a survey approach, composed of 33 closed questions and 01 open question, aimed to collect information from practitioners concerning their perceptions about applying Design Thinking methods and tools in their work environment. Besides, we collected data regarding the practitioner's time of experience, what function they executed inside the development process using Design Thinking methodology, and which DT phases they actively participated.

The survey also contained a question section addressed to developers to collect data related to Design Thinking employed in requirement elicitation and how the developer keeps up with methodology's iteration. The questionnaire was available through the Google Forms tool, within the period of 01/08/2019 to 20/11/2019. It was distributed to both public and private Information Technology (IT) companies. A total of 59 answers was obtained. And that was considered the research scope. Section 4 presents the results obtained with the survey and a discussion thereof.

4 SURVEY RESULTS AND DISCUSSIONS

In this section we present the main findings of our research, answering the general research questions we investigate and we expose a discussion of the obtained results. Regarding the practitioners' profile in the organizations, including age, degree, and experience in software development projects, 25.4% of practitioners are in the age range of 31 to 36 years old, 18.6% are between 26 to 30 years old, 22% are within 37 to 42 years old, 15.3% between 42 to 47 years old, and 8.5% are between 42 to 47 years old. Beyond this, 22% of the respondents are graduated, 25.4% have a specialization degree, 28.8% are master students, 10.2% are PhD students, 5.1% are masters and 6.8% are undergraduate students.

Regarding the professional experience time in the Information Technology field, 23.7% of the practi-

tioners have between 7 and 10 years of experience, 18.6% work longer than 21 years in the area, 15.3% have between 11 and 15 years of practice, 13.6% between 16 and 20 years, 11.9% have between four and six years and just 8.5% have less than a year of experience. Among these results, it is possible to conclude that the majority of the responders that use the Design Thinking methodology is composed of experienced professionals, relatively young, mostly under 50 years old.

Regarding participation in software development projects with an agile team, 76.3% from the practitioners affirmed to be currently in a project, 15.3% asserted that are not participating currently, but already joined, and only 8.5% declared to know the agile methods but never been in a project using agile methodology. Concerning the organization's performance context in which the professional participated/participates in software development projects in agile teams, 42.1% from practitioners works in private organizations, 43.9% in research project, 31.6% work in Federal Public Administration (FPA), 19.3% work in public companies (State-owned enterprises), 8.8% in open-source software projects.

4.1 RQ.1

Related to the question **RQ.1** we identified that 80.8% of practitioners applied prototyping, 53.8% Brainstorming, and 42.3% interviews. With lower participation are the methods CSD matrix (7.7%) and Moodboard (3.8%), as presented in Table 1. This ratifies partial findings from authors as Souza et al (Souza et al., 2017) that recognized Brainstorming as the most cited technique. Correia et al. (Correa et al., 2018) classified prototyping, brainstorming, interview and other methods used in the development process. From these results, it is perceptible that the DT techniques applied during the software development process are used as a group in the same DT stage.

4.2 RQ.2

Concerning the question **RQ.2**, it was observed that 73.1% utilize prototyping, 61.5% Interviews, and 50% Brainstorming. The lower use rates belong to CSD Matrix (7.7%) and Blueprint Service (7.7%), as represented in the table's second column 1. Authors as Lira and Silva (Lira de Carvalho Souza and Silva, 2015) and Vetterli et al. (Vetterli et al., 2013a) identified prototyping as a main Design Thinking method in requirement elicitation. Canedo and da Costa (Canedo and da Costa, 2018) presents an experience

report about prototyping use in two software development projects and effectiveness in using the prototyping technique in requirements elicitation and systems interface elaboration.

4.3 RQ.3

Related to the question **RQ.3** were presented in the survey, the phases as presented by Plattner et al. (Plattner et al., 2009) and it was observed that all identified phases (Immersion, Empathy, Definition, Ideation, Prototyping, and Tests) are utilized in a representative way at organizations responding. Figure 1 presents the identified percentages, which shows that Prototyping (75%) and Empathy (64.3%) are the most representative. Important to reinforce that the other phases were also adopted: Immersion (57.1%), Definition (60.7%), Ideation (57.1%), and Tests (53.6%).



Figure 1: Design Thinking Phases Adopted in Software Development Projects.

4.4 RQ.4

There were few answers to question **RQ.4**. We present in the survey a list of tools used in the software development process (Schwaber, 2005),(Bitner et al., 2008),(Morelli and Tollestrup, 2006),(Long, 2009),(Bittner and Shoury, 2019),(Bitner et al., 2008). Only 6 from the 59 respondents identified any tools used. These are the ones mentioned: Adobe XD (2 respondent), Quant-UX, Kanban Virtual, Mind Map Free, Ms-Word, Bizagi, Sketch, Axure, BonitaSoft, and Pencil, as shown in Figure 2. From the tools mentioned only Axure is cited in Souza et al. (Souza et al., 2017) systematic review. This allows us to infer that industrial practice identifies different tools than the academy.

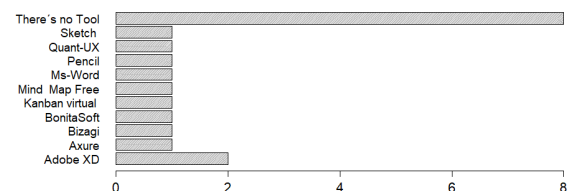


Figure 2: Design Thinking Tools Used by Practitioners in Software Development Projects.

Table 1: Design Thinking Methods and Tools Applied in Software Development Process versus Applied in Requirement Elicitation.

Techniques	Percentage in Software Development	Percentage in Requirement Elicitation
Prototyping	80.8%	73.1%
Brainstorming	53.8%	50.0%
Interview	42.3%	61.5%
Mind Map	38.5%	38.5%
Personas	38.5%	38.5%
Story Telling	34.6%	15.5%
Storyboard	30.8%	30.8%
Customer Journey Map	19.2%	26.9%
Insight Cards	15.4%	19.2%
Role Sprint	15.4%	11.5%
Empathy Map	15.4%	23.1%
Pitch	15.4%	15.4%
Blueprint Service	11.5%	7.7%
CSD Matrix	7.7%	7.7%
Moodboard	3.8%	0%

Interesting to mark that, according to Chasanidou et al. (Chasanidou et al., 2015), a large number of design techniques and tools facilitate the DT innovation process. Selecting correct methods is essential mainly during the initial phases. Comprehending which and how are applied those DT methods and tools in Software Engineering provide the recognition that alternatives apply in the software development process. The results found show that there are still not a large number of software tools supporting assist DT methodology use along the software development process.

4.5 RQ.5

Regarding question **RQ.5** we identified the following aspects, all represented in Figure 3: Regarding software development:

- Had a Designer in the team during all DT phases increases the development team efficiency (95% of the developers) (Q21);
- DT methodology use increases the developed product quality (88% of the developers) (Q24);
- DT enables the creation of a friendly interface (82% of the practitioners) (Q16);
- DT improves the information flow definition (79% of the practitioners) (Q15);
- Interaction inside the development team during the project was satisfying (79% of the developers) (Q25);
- DT use supports the entire development process (75% of the practitioners) (Q20);

- DT allows a suitable architectural definition (58% of the developers) (Q17).

Regarding requirements:

- From prototypes it is possible to identify mistakes in requirement understanding (89% of the practitioners) (Q18);
- DT improves the system's requirements elicitation (86% of the respondents) (Q14).

Regarding Tools:

- Recognize as Positive the experience of using prototypes to support project functionalities development (96% of the developers) (Q22);
- Recognize as positive the visualization of the system's components, buttons, and screens through prototypes regarding the efficiency of their project's deliveries (91% of the developers) (Q23);
- DT facilitates implementation activities, once that prototypes are validated with the customer (88% of the practitioners) (Q19).

The survey also enabled the identification of the following results about the advantages of using the methodology Design Thinking in an agile software development project. The practitioners mentioned in the open question that Design Thinking enables: "Allow previous planning of what needs to be developed", "Improve customer reliability", "Improve end-user participation (active product participant)", "Reduce risk and investment in unnecessary product or service"; "Promote the validation and verification of new customer experiences, especially for the user experience (UX) and user interface (UI) area". These

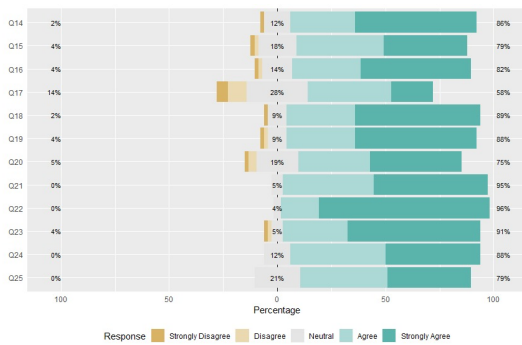


Figure 3: Developers' Perception about Applying Design Thinking in Software Development Projects.

findings confirm the research results performed by Pereira (Pereira, 2018) and Vetterli et al. (Vetterli et al., 2013b).

5 THREATS TO VALIDITY

Regarding the internal validity of the survey, we conducted a review of the questionnaire questions in order to mitigate the threats (Wainer, 2007). In addition, three other researchers validated the survey questions. Regarding external validity, the survey participants might not be representative of the general population of software developers, the generalizability of our survey results might be limited. To mitigate this risk, we advertised our survey through various channels to a large audience. Having gathered data from software developers from various organizations, and with different levels of software development experience, we believe that our sample is fairly representative of software developers of agile teams and that it provides an interesting perspective. Participants could freely decide whether to participate in the study or not (self-selection). They were informed about the survey's topics and an estimated duration for the participation. This could have biased the selection of participants as only participants who could spare enough time might have participated. We tried to mitigate this risk by advertising through various channels.

Regarding the validity of conclusion, according to Travassos and Amaral (Travassos and Amaral, 2002) is related to the ability to reach a correct conclusion about the relationships between treatment and the result of the research. To mitigate this risk the results presented were analyzed by three researchers. We recognize that there is no way to fully mitigate this threat by considering only the number of survey participants, but we consider that results that have been partially ratified by other research could be considered as indicative of practitioners' perception of DT.

In the future, we will conduct the survey with a larger population of participants to ratify these findings.

6 CONCLUSION

Design Thinking techniques and supporting tools used along the agile software project development have been widely adopted by professionals in the software industry. However, there are not many practical examples of the technique applied in the literature, by using real software projects, and also not many empirical studies that validate the results of using DT in real software systems.

This work investigates Design Thinking practice applied to real projects, and which are its benefits according to the perception of practitioners in some software development companies. 76.3% of the survey participants work or already worked in agile software development projects applying DT methodology or DT tools and the most used technique is prototyping followed by brainstorming.

In requirement elicitation, the most used technique is also prototyping, followed by users' interviews. Besides, professionals reported that prototyping allows mistakes identification in requirements understanding before implementation, which prevents possible problems among project stakeholders.

The benefits observed in DT methodology applied to software development use are: 1. Enhancing the requirement elicitation process; 2. Allowing errors identification in requirements understanding from prototyping; 3. Easing implementation once that prototypes are validated directly with clients. Although agile team developers do not sympathize with documentation, according to the survey participants, practitioners still use requirement specification documents to implement software functionalities, even when using DT techniques in requirement elicitation.

As future work, it is intended to investigate and monitor the main challenges faced by practitioners in software development teams when using Design Thinking, as well as which are their suggestions to adopt the methodology widely in real contexts.

ACKNOWLEDGMENTS

This research work has the support of the Research Support Foundation of the Federal District (FAPDF) research grant 05/2018.

REFERENCES

- Bitner, M., Ostrom, A., and Morgan, F. (2008). Service blueprinting: A practical technique for service innovation. *California Management Review*, 50:66–94.
- Bittner, E. and Shoury, O. (2019). Designing automated facilitation for design thinking: A chatbot for supporting teams in the empathy map method. In *52nd Hawaii International Conference on System Sciences, HICSS 2019, Grand Wailea, Maui, Hawaii, USA, January 8-11, 2019*, pages 1–10.
- Brown, T. and Katz, B. (2009). *Change by design: how design thinking can transform organizations and inspire innovation, 1st Edition*. Harper Collins.
- Canedo, E. D. and da Costa, R. P. (2018). The use of design thinking in agile software requirements survey: A case study. In *HCI (18)*, volume 10918 of *Lecture Notes in Computer Science*, pages 642–657. Springer.
- Chasanidou, D., Gasparini, A. A., and Lee, E. (2015). Design thinking methods and tools for innovation. In *HCI (18)*, volume 9186 of *Lecture Notes in Computer Science*, pages 12–23. Springer.
- Correa, L., Maria, D., Bellio, J. C., Marczak, S., and Conte, T. (2018). Software: Um estudo de caso no contexto de academias de musculacao. In *Cadernos do IME: Série Informática: Vol. 41*.
- de Almeida, E. M., Damasceno, E. F., and L’Erário, A. (2018). Teaching multidisciplinary teams requirements for undergraduate students: an approach to augmented reality software in design thinking context. In *FIE*, pages 1–7. IEEE.
- de Carvalho Souza, C. L. and Silva, C. T. L. L. (2015). An experimental study of the use of design thinking as a requirements elicitation approach for mobile learning environments. *CLEI Electron. J.*, 18(1).
- Femmer, H. and Vogelsang, A. (2019). Requirements quality is quality in use. *IEEE Software*, 36(3):83–91.
- Hehn, J., Fernández, D. M., Uebernickel, F., Brenner, W., and Broy, M. (2019). On integrating design thinking for a human-centered requirements engineering. *CoRR*, abs/1908.07223.
- Hehn, J. and Uebernickel, F. (2018). Towards an understanding of the role of design thinking for requirements elicitation - findings from a multiple-case study. In *AMCIS*. Association for Information Systems.
- 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 *RE*, pages 504–505. IEEE Computer Society.
- Higuchi, M. M. and Nakano, D. N. (2017). Agile design: A combined model based on design thinking and agile methodologies for digital games projects. *Revista de Gestão e Projetos*, 8(2):109.
- Knauss, E. (2019). The missing requirements perspective in large-scale agile system development. *IEEE Software*, 36(3):9–13.
- Kusters, R. J., van de Leur, Y., Rutten, W. G. M. M., and Trienekens, J. J. M. (2017). When agile meets waterfall - investigating risks and problems on the interface between agile and traditional software development in a hybrid development organization. In *ICEIS (2)*, pages 271–278. SciTePress.
- Liedtka, J. and Ogilvie, T. (2011). *Designing for growth: A design thinking tool kit for managers*. Columbia University Press.
- Lira de Carvalho Souza, C. and Silva, C. (2015). An experimental study of the use of design thinking as a requirements elicitation approach for mobile learning environments. *CLEI Electronic Journal*, 18:6–6.
- Long, F. (2009). Real or imaginary: The effectiveness of using personas in product design - frontend. *Proceedings of the Irish Ergonomics Society Annual Conference*, pages 1–10.
- Masood, Z., Hoda, R., and Blincoe, K. (2017). Motivation for self-assignment: Factors agile software developers consider. In *CHASE@ICSE*, pages 92–93. IEEE Computer Society.
- Moggridge, B. (2007). *Designing interactions*. MIT Press.
- Morelli, N. and Tollestrup, C. (2006). New representation techniques for designing in a systemic perspective. *Engineering and Product Design Education Conference*.
- Pereira, J. C. (2018). Aplicação do design thinking integrado com métodos ágeis na gestão de projetos de software. *U9J*.
- Pereira, J. C. and de FSM Russo, R. (2018). Design thinking integrated in agile software development: A systematic literature review. *Procedia computer science*, 138:775–782.
- Plattner, H., Meinel, C., and Weinberg, U. (2009). *Design-thinking*. Springer.
- Schwaber, K. (2005). Agile project management. In *Extreme Programming and Agile Processes in Software Engineering, 6th International Conference, XP 2005, Sheffield, UK, June 18-23, 2005, Proceedings*, page 277.
- Souza, A., Ferreira, B., and Conte, T. (2017). Aplicando design thinking em engenharia de software: um mapeamento sistemático. pages 1–10.
- Travassos, G. H., G. D. and Amaral, E. A. G. (2002). Introdução à engenharia de software experimental. *COPPE/URFJ Relatório técnico*, RT-ES590/02.
- Urbig, D. and Verlage, S. (2003). The value chain’s values: Interpretations and implications for firm and industry analysis. *Proceedings of Perspectives in Business Informatics Research (BIR)*, pages 1–15.
- Vetterli, C., Brenner, W., Uebernickel, F., and Petrie, C. (2013a). From palaces to yurts - why requirements engineering needs design thinking. *IEEE Internet Computing*, 17.
- Vetterli, C., Brenner, W., Uebernickel, F., and Petrie, C. J. (2013b). From palaces to yurts: Why requirements engineering needs design thinking. *IEEE Internet Computing*, 17(2):91–94.
- Wainer, J. (2007). *Metodos de pesquisa quantitativa e qualitativa para a ciencia computacao*. PUC RIO.