Requirement Engineering and the Role of Design Thinking

Anas Husaria[1,2] and Sergio Guerreiro[1,2]
[1]*Instituto Superior Técnico, University of Lisbon, Av. Rovisco Pais 1, 1049-001 Lisbon, Portugal*
[2]*INESC-ID, Rua Alves Redol 9, 1000-029 Lisbon, Portugal*

Keywords: Design Thinking, Software Development, Requirement Engineering, UX, Agile, HCI.

Abstract: Recently, interest in the use of Design Thinking (DT) has been on the rise as a field to study related to Human-Computer Interaction (HCI). Companies today are seeking innovation and user-centricity in the software development projects regardless of the field they are in. Companies like IBM and SAP among others have taken steps forward to innovate their software development processes using Design Thinking, by creating academies and innovation labs. DT is used as a method to improve the User Experience (UX) while interacting with a computer software. Requirement Engineering (RE) is a process of defining, documenting and maintaining requirements in the system design and software engineering process, while RE takes on the initial phase in software engineering. This position paper reviews the practices of RE as well as how DT could have a role to mitigate the challenges that RE could have.

1 INTRODUCTION

Human-Centred Design is heavily used today in the field of software engineering, as a method to promote a better User Experience (UX) and a better engagement of users with the developed software which ultimately would benefit both the business and the end users (Hehn et al. 2019). DT has the potential to add value to the current RE process by tackling the challenges it has, which would ultimately improve the perceived UX of the developed software. In order to ensure the quality of the provided UX in the field of Human-Computer Interaction (HCI) usability was always one important measure (Mekler and Hornbæk 2019). In addition to that, Design Thinking (DT) is being considered as one of the most influential method to solve complex problems and provide usable solutions (Hehn and Uebernickel 2018). Uber and Airbnb among other software-based companies have used DT to innovate their service model and to develop a usable software that is solving real life problems (Geogy and Dharani 2016; Liedtka 2018).

Although Requirement Engineering has been heavily researched the need to innovate in this field is relevant due to the evolving environment of software developments. By introducing the iterative work style (e.g. Scrum and Kanban) Agile Methods have increased the flexibility in software development (Inayat, Moraes, et al. 2015). Nevertheless, DT has the potential to support in solving complex problems and mitigating the risks that current processes are having. This paper has analysed articles and research publications which reviewed Requirement Engineering and studied the applications of Design Thinking (DT) on current software development problems.

The paper starts with reviewing the requirement engineering literature, particularly by comparing traditional and agile requirement engineering methods. Afterward, the paper describes the usage of DT in RE, particularly analyse the challenges of agile requirement engineering which DT has a potential to solve. Then, the paper explains how DT is applied in an intensive software development project. Finally, the paper analyses the benefits of using DT up front in a software project by taking a case study from the industry as an example where DT was applied up front at a large organization. In the conclusion section, the paper summarizes the findings and suggests future research questions in field.

2 REQUIREMENT ENGINEERING REVIEW

Requirement Engineering (RE) focuses on the comprehensive understanding of stakeholder (i.e., customers, users, etc.) requirements and the documentation of these requirements to make them

Husaria, A. and Guerreiro, S.
Requirement Engineering and the Role of Design Thinking.
DOI: 10.5220/0009489303530359
In *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020) - Volume 2*, pages 353-359
ISBN: 978-989-758-423-7

available for a structured software engineering process (Batool et al. 2013). Additionally, RE supports the process of identifying, modelling, documenting and communicating both the requirements and the context of a system (Heikkila et al. 2015). While RE has matured to become an essential activity in the field of software engineering, the software product quality is widely based on the quality of the development process used to build it (Takeuchi and Ikujiro 1986). Therefore, it is significant to keep improving the process with the aim to enhance the quality of the final software product.

2.1 Traditional Requirement Engineering Process in Comparison to Agile Requirement Engineering

In order to have a better understanding of RE, this paper provides a review for the two main approaches used in research for RE.

2.1.1 Traditional Requirement Engineering Process

Requirement Engineering (RE) could be looked at from three dimensions: the specification dimension, the representation dimensions and the agreement dimension. Studying RE using this framework, would support in creating a better understanding of RE, by classifying the different approaches (Pohl 1993). The waterfall life cycle model, which emerged in 1970s, has sat the bases for the term "traditional requirements engineering" (Batool et al. 2013). The model suggested that the approach to develop a system can be done via a sequential order of progress (Batool et al. 2013). The sequential phases of RE in waterfall are requirements definition, system and software design, implementation and unit testing, integration and system testing, and finally operation and maintenance (Batool et al. 2013).

- Requirement Definition

During the requirements definition stage, several processes for collecting requirements which meet the needs of the users are involved (Takeuchi and Ikujiro 1986). The requirements definition stage starts with elicitation where the stakeholders set the margins and scope of both the requirements and the system using techniques such as, interviews, brainstorming, prototypes and use cases (Pohl 1993). The discovering of requirements from other sources also happen during the requirement elicitation phase (Heikkila et al. 2015). For diving deeper in some of the most important techniques for requirements elicitation the paper has summarized some of them.

Interviews: Interviewing is an approach for discovering facts and opinions held by potential users or stakeholder of the software to be developed as well as the chance to clear up misunderstandings (Kotonya and Sommerville 1998). There are two different types of interview styles, one is closed, in which the requirement engineer would have pre-defined set of questions. While the other one is an open interview, where the requirement engineer goes to discuss with stakeholders with an open-ended way what they need to have in the system (Kotonya and Sommerville 1998). The benefit of interviews on one hand, is to make developers aware with a lot of contextual information. On the other hand, interviews pitfall is that it is difficult to contemplate the large amount of qualitive data acquired, while meeting different stakeholder could lead to providing conflicting information (Kotonya and Sommerville 1998).

Use cases: the objective of using use case is to describe system-user interactions, in a way that would explain the user need out of the system (Kotonya and Sommerville 1998; Paetsch and Maurer 2003). A use case examines a flow of interaction between an external actor and a system. The functional requirement of the system could be represented in use cases at an early stage of the software development process (Kotonya and Sommerville 1998).

Observation and social analysis: in the observational methods, the requirement engineer would be involved in viewing users doing their work and take notes to build up knowledge about their reality. It is a useful method for learning about currently executed tasks and processes (Kotonya and Sommerville 1998).

Focus Groups: in this technique a group of four to nine users from several backgrounds and with multidisciplinary skills discuss in a free form the features of software prototype. Focus groups help to understand user needs and viewpoints as well as what is of high priority to them (Macaulay 1996).

Brainstorming: is a method to promote creative solutions for an examined problem. Brainstorming includes two stages, one is the ideas generation, where ideas should not be criticized, and the second is idea selection or evaluation where the team decides on what is the most feasible idea to implement (Heikkila et al. 2015).

Prototyping: prototypes of software systems are usually used to support in the elicitation and validation of the software. A prototype of system is often done in an early stage of the development process, to help collecting more information about the

expectations of users from the system (Heikkila et al. 2015). Usually the requirement engineer would select one or some of these methods to use in requirement definition stage depending on what fit the project.

- Requirements Analysis

Then next step in requirement definition stage, is analysis and negotiation. In this stage a deeper understanding and a logical breakdown of the business through clarifying sessions can be acquired while confirming that the elicited requirements are comprehensive, viable, prioritized and consistent (Batool et al. 2013; Pohl 1993).

- Requirements Documentation

The next step is the requirement documentation or specification, where the requirements are documented in a written form and used for stating functional and non- functional requirements (Batool et al. 2013; Pohl 1993). The purpose of requirements documentation is to communicate requirements between stakeholders and developers (Heikkila et al. 2015).

- Requirements Validation

The validation or verification activity contains checking if the requirements statements are coherent and if they match the user needs using test cases or prototypes (Pohl 1993). The main goal of requirements validation is to make sure that the documented requirements are providing an acceptable description of the system to be developed. Requirement documents, organizational standards, and organizational knowledge are used as inputs for the validation process (Heikkila et al. 2015).

- Requirements Management

At the end of the development process, the requirement management stage is initiate. This activity focuses on keeping a track of changes in the requirements and ensuring that alterations are made to meet business needs and stakeholder's requirement (Heikkila et al. 2015; Northrop and Clements 2012).

2.1.2 Agile Requirement Engineering

Since the mid-80s, there was an on-going discussion on how to improve the current processes of Requirement Engineering. The sequential phases approach to product development is not fit because it does not provide enough flexibility (Schön, Thomaschewski, and Escalona 2017). Since that time, new models and experimentations to improve the current process were proposed and developed (Schwaber 2004). There have been already few agile methodologies developed, such as Extreme Programming (Beck 1999), Feature-Driven Development (Palmer and Felsing 2001), Kanban ((J. Anderson 2010) and Scrum (Schön, Thomaschewski, and Escalona 2017). In the meantime, the Agile Manifesto was created by the leaders of these methodologies after they have joined forces in 2001 (Hohl et al. 2018). The Agile Manifesto has provided principles to follow when developing a software in order to optimize the process and enable team's collaboration (Schön, Escalona, and Thomaschewski 2015). The main values of the Agile Manifesto are:

- Working software over comprehensive documentation
- Customer collaboration are over contract negotiation
- Responding to changeover sticking to the plan
- Individuates and interactions are over processes and tools

One of the Agile Manifesto principles states "changes in requirement are welcome, even late in the development", which proposes that RE would be a continuous process throughout the lifecycle of the system (Inayat, Moraes, et al. 2015). The most common agile software method is scrum (e.g. of the scrum framework depicted in Figure. 1 (Armitage, Cordova, and Siegel 2017) which focuses on having frequent deliverables in small iterations of work that keep the process dynamic.

The requirements are initially defined by the client, while they are listed in a backlog. Then every two weeks they are discussed with the team in order to have them fully comprehended and prioritized then moved into the next sprint backlog (Schwaber and Sutherland 2017).

Although the agile values would improve the flexibility of Requirement Engineering in the software development process nevertheless, recent researches were indicating project failure rates on the rise, even for the projects that are using agile processes (Inayat, Moraes, et al. 2015). The reason behind that is that, the agile new method of working has brought a challenge and an unclarity to the execution of Requirement Engineering processes (Inayat, Moraes, et al. 2015).

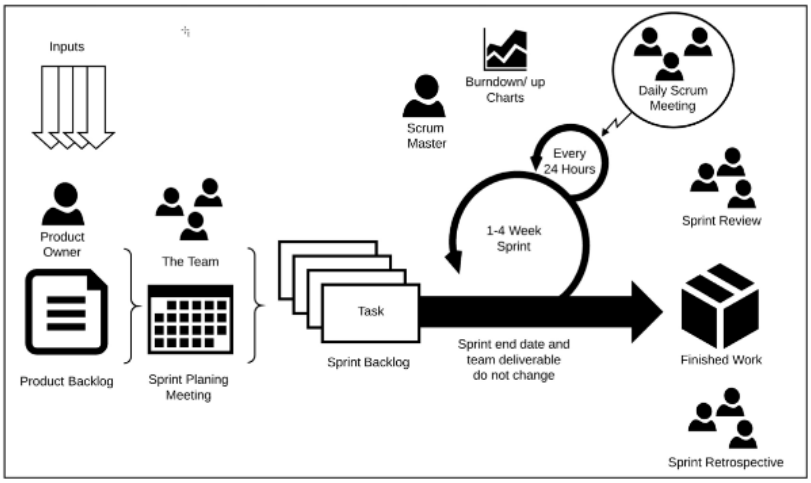

Figure 1: Scrum Framework.

3 DESIGN THINKING IN REQUIREMENT ENGINEERING

In the field of software development, Requirement Engineering (RE) has been always used to support the understanding of complex business problems and for the elicitation of user needs in volatile world where user needs are unclear, and several parties are involved in a project. Although agile frameworks have contributed a lot to the field of software development making the work organized (Heikkila et al. 2015). However, Design Thinking is providing a more structured method to solve these complex problems. Design Thinking (DT) is defined as an innovative human-centred method to integrate the human requirement (desirability), the technological capabilities (feasibility) and the business requirement to be profitable (viability) (Brown 2008). DT is using interdisciplinary teams, focusing on human needs, fast prototyping and an ongoing cycle of learning iteration to gain empathy at an early stage of the software development process (Armitage, Cordova, and Siegel 2017; Kolko 2015)

3.1 Challenges of Agile Requirement Engineering that DT has the Potential to Solve

While the complicated part in any system design is what to build, the goal is to find the “sweet spot” or which is a point that gives the right amount of requirement information which keeps the right amount of ambiguity for the development team to discover along the process (Wahono 2003). DT Role in RE can be investigated by case study (Hehn and Uebernickel 2018). DT has the potential for a positive impact on some of the current RE practices (Hehn and Uebernickel 2018). During their research several challenges related RE in agile set-ups have been identified. Some of which are discussed below in addition to the role of DT to positively impacting each challenge. Some of the found challenges are discussed below.

Issues with customers or users, the difficult access to and contacting with directly with customers slows the process of clarifying requirements down (Heikkila et al. 2015; Inayat, Salim, et al. 2015). DT provides a solution regarding customer or user availability that normal RE methods face, since DT is a process-oriented method, the user interviews can be planned upfront, which will support overcoming this challenge (Hehn and Uebernickel 2018).

Non-functional requirements are neglected, user stories are usually focused on software features, but non-functional requirements such as security and usability are not well covered (Inayat, Salim, et al. 2015). Although DT neglects most of non-functional requirement as well, however, it enforces heavily the importance of eliciting usability requirements (Hehn and Uebernickel 2018).

Additionally, majority of the knowledge often stays tacit as agile practices rely on highly skilled people 2+20. However, DT supports the distribution of knowledge throughout the team (Hehn and Uebernickel 2018). Additionally, the involvement of interdisciplinary teams assists in enriching different

viewpoint which would enhance the whole team understanding of the user needs (Hehn and Uebernickel 2018).

Inaccurate effort estimates of time and cost happens due to the agile project characterizes (Heikkila et al. 2015; Inayat, Salim, et al. 2015). DT approach by focusing upfront on the user needs and the problem space, would provide a clear scope and product vision by elaborating well defined problem statement at an early stage (Hehn and Uebernickel 2018). This would serve as a mitigation to problems such as imprecise effort estimation and usage of the inappropriate technology (Hehn and Uebernickel 2018).

3.1.1 Design Thinking in Intensive Software Development Project

To have deep comprehension of DT for RE as socio-technological activity (Hehn and Uebernickel 2018) adopted a quantitative approach and applied it to a case study. Current RE practices in software development projects can be enhanced by applying DT and vice versa (Hehn and Uebernickel 2018). DT is proven to be a supportive approach in turning complex problems into well-defined ones that then would be easily approached by the known RE activities (Mekler and Hornbæk 2019). However, an effective integration between both DT and RE would leverage a synergetic relationship (Hehn et al. 2019; Hehn and Uebernickel 2018). Software Development can be a great ground for DT applications. Therefore, DT would enable innovation and creativity by iterative re-framing of the problem and solution space, making sure the best User Experience (UX) is provided (Hehn and Uebernickel 2018).

4 BENEFITS OF USING DESIGN THINKING UP FRONT IN A SOFTWARE DEVELOPMENT PROJECT

In a large organization, using design thinking method early on in software projects could have a positive influence on the success of the project. To verify that, we have investigated a medium size project in a large organization, the project initial phase involved the following:

- The goal of the project is to create a new financial analysis tool that.
- Around 20 people in the project were involved.

To implement that, a design thinking facilitators team have joined the project for one-week time to run a design thinking workshop in order to come up with a clear problem statement. The design thinking workshop went through the whole process of design thinking, starting with empathy, where the team have run several interviews and done shadowing for current users. Next step, the joint team have defined a well-stated problem statement. After that, an ideation workshop took place where it was possible to obtain several ideas to solve the defined problem statement. The team have used the how-might we method to inspire creativity during the solution generation phase. As a next step, the team have decided on the best solution to prototype and provided a digital mock-up for the users to test. The user testing stage took place afterward and brought several learnings for the team to reiterate the prototype until the users have accepted the prototype.

The prototype and testing results have been submitted to the project management to decide on investment and implementation. Based on the positive results of the testing stage and the impact perceived if the new solution was implemented, the project management have decided to invest in implementing this new solution.

The software project is currently running, and we are still monitoring the results of our investigation on using design thinking up front in a new software project. However, monitoring this case study have led us to identify few preliminary results. We have categorized these results in three parts which are the main aspects of design thinking. The aspects are Human Desirability, Technological Feasibility and Business Viability.

The results are described according their respective category in Table 1.

Table 1: Identified Benefits of Using Design Thinking Up Front in Software Project.

Human Desirability	User adoption rates of the first release of the software have been seen be above average in comparisons to other projects.
Technological Feasibility	The choice of the appropriate technology to support the selected solution came after, which saved the company the effort of investing in several technologies before deciding solution.
Business Viability	As the project implementation and user adoption are in parallel going up and forward. The business has noticed lower running costs in this project in comparison to similar ones at the same department that did not use design thinking up front.

The above results in Table 1 are still preliminary, and we are still monitoring this project by running interviews with the project team to understand the effect of using design thinking up front on the full life cycle of the software development project. However, the current results show a positive potential for design thinking role as requirement engineering method.

5 CONCLUSIONS

Requirement Engineering usually faces the difficulty of discovering and meeting the unarticulated and changing needs of various stakeholder (Hehn et al. 2019). Current RE practices in software development projects can be enhanced by applying DT and vice versa (Hehn and Uebernickel 2018). As DT has the potential of solving some challenges of Agile RE. As a result, a better development process would lead to better built product and ultimately improved UX. Thus, integrating Design Thinking in current Requirement Engineering studies a potential to improve the outcome of development process. Additionally, the case study we are running shows a positive impact for the role of DT in software development project. Future research is recommended on special case studies where design thinking is applied for mature software product in large companies.

ACKNOWLEDGEMENT

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UIDB/50021/2020 and by the European Commission program H2020 under the grant agreement 822404 (project QualiChain).

REFERENCES

Armitage, A., Andrew, C., Rebecca, S., 2017. "Design-Thinking: The Answer to the Impasse between Innovation and Regulation." *UC Hastings Research Paper* 250: 26–36. https://ssrn.com/abstract=3024176.

Batool, A., Motla, Y., Hamid, B., Asghar, S., Riaz, M., Mukhtar, M., Ahmed, M., 2013. "Comparative Study of Traditional Requirement Engineering and Agile Requirement Engineering." In *2013 15th International Conference on Advanced Communications Technology (ICACT)*, , 1006–14.

Beck, K,. 1999. *Extreme Programming Explained: Embrace Change*. USA: Addison-Wesley Longman Publishing Co., Inc.

Brown, T., 2008. "Design Thinking." *Harvard Business Review* 86(6): 84–92. www.hbr.org.

Geogy, M., and Andhe, D., 2016. "A Scrutiny of the Software Requirement Engineering Process." *Procedia Technology* 25: 405–10. https://www.sciencedirect.com/science/article/pii/S2212017316304728 (January 6, 2020).

Hehn, J., Mendez, D., Uebernickel, F., Brenner, W., Broy, M., 2019. "On Integrating Design Thinking for a Human-Centered Requirements Engineering." *https://arxiv.org/abs/1908.07223*: 1–5.

Hehn, J., Uebernickel, F., 2018. "The Use of Design Thinking for Requirements Engineering - An Ongoing Case Study in the Field of Innovative Software-Intensive Systems." *26th IEEE International Requirements Engineering Conference (RE'18)* August. https://www.alexandria.unisg.ch/publications/254658.

Heikkila, V., Damian, D., Lassenius, C., Paasivaara, M., 2015. "A Mapping Study on Requirements Engineering in Agile Software Development." *Proceedings - 41st Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2015*: 199–207.

Hohl, P., Klünder, J., van Bennekum, A., Lockard, R., Gifford, J., Münch, J., Stupperich, M., Schneider, K., 2018. "Back to the Future: Origins and Directions of the 'Agile Manifesto' – Views of the Originators." *Journal of Software Engineering Research and Development* 6(15).

Inayat, I., Salim, S., Marczak, S., Daneva, M., Shamshirband, S., 2015. "A Systematic Literature Review on Agile Requirements Engineering Practices and Challenges." *Computers in Human Behavior* 51: 915–29. https://www.sciencedirect.com/science/article/pii/S074756321400569X?via%3Dihub (January 29, 2020).

Inayat, I., Moraes, L., Daneva, M., Salim, S., 2015. "A Reflection on Agile Requirements Engineering: Solutions Brought and Challenges Posed." In *1st International Workshop on Requirements Engineering in Agile Development (READ)*, Association for Computing Machinery (ACM), 61–67.

Anderson, J, D., 2010. *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press.

Kolko, J., 2015. "Design Thinking Comes of Age." *Harvard Business Review* (September): 66–71.

Kotonya, G., Sommerville, I., 1998. *Requirements Engineering: Processes and Techniques*.

Liedtka, J., 2018. "Why Design Thinking Works." *Harvard Business Review* 2018(September-October).

Macaulay, L, A., 1996. *Requirements Engineering*. Springer-Verlag London.

Mekler, E, D., and Hornbæk, K., 2019. "A Framework for the Experience of Meaning in Human-Computer Interaction." *Conference on Human Factors in*

Computing Systems - Proceedings (CHI 2019): 1–15. https://doi.org/10.1145/3290605.3300455.

Northrop Linda M., and Clements Paul C., 2012. "A Framework for Software Product Line Practice, Version 5.0." *Software Engineering Institute*: 258. https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=495357.

Paetsch, F., Maurer, F., 2003. "Requirements Engineering and Agile Software Development." : 1–6.

Palmer, S, R., Felsing, M., 2001. *A Practical Guide to Feature-Driven Development*. 1st ed. Pearson Education.

Pohl, K., 1993. "The Three Dimensions of Requirements Engineering BT - Advanced Information Systems Engineering." In eds. Colette Rolland, François Bodart, and Corine Cauvet. Berlin, Heidelberg: Springer Berlin Heidelberg, 275–92.

Schön, E., Escalona, M., Thomaschewski, J., 2015. "Agile Values and Their Implementation in Practice." *International Journal of Interactive Multimedia and Artificial Intelligence* 3(5): 61–66. http://www.ijimai.org/journal/sites/default/files/files/2015/11/ijimai20153_5_8_pdf_59735.pdf.

Schön, E., Thomaschewski, J., Escalona, M., 2017. "Agile Requirements Engineering: A Systematic Literature Review." *Computer Standards & Interfaces* 49: 79–91. https://www.sciencedirect.com/science/article/pii/S0920548916300708 (January 6, 2020).

Schwaber, K., 2004. *Agile Project Management With Scrum*. USA: Microsoft Press.

Schwaber, K., Sutherland, J., 2017. Scrum.Org and ScrumInc *The Scrum Guide: The Definitive The Rules of the Game*. http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf.

Takeuchi, H., Ikujiro, N., 1986. "The New New Product Development Game." *Harvard Business Review* 64: 137–146.

Wahono, R., 2003. "Analyzing Requirements Engineering Problems." In *IECI Japan Workshop*, , 55–58.