Combining Design Thinking and Scrum to enhance Test Suite during innovative software development*

Flávia Lobato de Morais[1][0000−0003−4981−0250], Awdren Fontão[2][0000−0002−2988−9646], and Sergio Cleger Tamayo[1][0000−0002−2025−1638]

[1] SIDIA - Institute of Science and Technology, Manaus, Brazil
{flavia.morais, sergio.tamayo}@sidia.com
http://www.sidia.com/
[2] FACOM - Federal University of Mato Grosso do Sul
awdren@facom.ufms.br

Abstract. Design Thinking (DT) is a human-centered methodology that has been used in generating and testing innovative ideas. It supports on how to understand user needs in order to derive solutions. Some of these solutions involve innovative functionalities. However, DT applies a test level (acceptance testing) only at the end of the cycle. When using an agile methodology such as Scrum, it is necessary to anticipate everything that might interfere with a Sprint Release. The innovative functionality must be delivered quickly due to competitiveness and time-to-market with the fewest possible defects in the hands of users. From the Software Testing perspective as a way of optimizing the testing process, it is interesting to anticipate all inputs that will allow the improvement of the Test Suite. Our goal is to explore how DT with Scrum can anticipate a set of inputs for Test Suite enhancement in innovative functionalities development. In this study, the DT method was applied within a Scrum setting in a case study that was conducted within the Brazilian Institute of Science and Technology. It involves a partner retail company that requested a user system with face authentication and recognition functionalities.

Keywords: Software Testing · Test Suite · Design Thinking · Scrum.

1 Introduction

The industry around software projects has moved to an agile paradigm that needs to prioritize high quality. This quality is an embracing theme in Software Engineering, and, can be quantified through software functionality and user acceptance. Design Thinking (DT) can be characterized as an attempt and error-based learning iterative process through prototyping and user testing [4]. This iterative feature is the essence of agile methodologies, such as Scrum. Scrum has been used in projects where you need to add value in the software for the

* Supported by organization Sidia.

customer in a short time, which includes innovative projects. Like DT, agile product development is characterized by iteration and experimentation with a clear focus on user requirements [2]. When it comes to requirements, software testing is an essential activity for Software Engineering, which is observing software execution to validate whether it behaves as expected [16].

A problem within Scrum is the gap of communication and constant cooperation between the Scrum team and its customers [14], but there is a bridge that exists between the two, that is the PO (*Product Owner*). This problem can be supported by using DT, since there are team interactions with the user (through testing) to get usage perceptions [15]. In any case, DT may not favor team negotiations and engagement, this is a point that Scrum can support by increasing change capacity and collaborating to continually exchange experience and knowledge [14]. The DT only considers the acceptance test level, ie the final user determines if the software is working according to what the user expects [17], and these many interactions can result in interface and interaction paths modifications. This can propagate changes to alternate flows, and also, new functional test cases [20]. Variations in interaction flow sets, however related to user perceptions, can lead to changes in functional requirements and, consequently, an increase in the suite of test cases.

Due to the presented scenario, and as indicated by Micheli et al. [15], it is important to explore the synergies between DT and other processes during software development. Bertolino in [3] also mentions that there is a need to cohesively test the interconnection with other disciplines (in the case of this study: DT and Scrum) to explore the functional and non-functional properties of the software; even within emerging development paradigms, such as those involving innovative functionality, thus facilitating rapid feedback throughout the development cycle. And this development paradigm that requires fast/frequent changes integration, has driven the need for improvement of the test suite [11].

Thus, the purpose of this paper is to investigate the use of DT to improve the test case suite in a Scrum environment for software development with innovative features. For this, an exploratory case study is described that helps to identify existing benefits and challenges in the project. This project involves the development of software with innovative features such as authentication and facial recognition on retail mobile devices. Findings from our study indicated that, for example: (1) Analyzing the testing requirements at the same time as the functional and non-functional software requirements proved to be the most efficient approach; (2) Identifying the high importance interaction flows around the innovative feature; (3) Mapping a set of models and architectures as a way to refine the parameter that can be used on input set to test cases.

This paper is organized as follows. The next section presents the concepts related to DT, Scrum and Software Testing. Following, Section 3 presents related work. The DT + Scrum + Test approach is presented in 3.1 subsection. The case study, including planning, execution, and results analysis are described in 4 section. And, lastly, the conclusion and future works are presented.

2 Background

2.1 Design Thinking (DT)

According to [5] design thinking (DT) is a collaborative and user-centered approach for problem resolution, that drives innovation through iteration, idea generation, and creative practices. DT permeates human-focused innovation activities through a detailed understanding process (through direct observation) to extract the wants and needs of people not described in traditional research, such as an interview [6]. DT consists of six stages [6]:

- Empathise: consists in imagining and observing scenarios from different users' perspectives in order to capture the context, understand the problems that will be the focus of the project and develop solutions;
- Define: analysis and synthesis, where user information is analyzed and insights obtained to propose new solutions;
- Ideate: innovative solutions are created from the insights gained in the previous step. That's when the big brainstorming occurs. Stakeholders are encouraged to think and propose solutions that are in accordance with the observed context;
- Prototype: can occur in parallel with the definition and ideation stages and allows the construction of solutions;
- Test: tests are applied to users in order to validate ideas generated through prototypes. UX assessment techniques (*User eXperience*) can be applied as a way to evaluate proposed solutions.;
- Implementation: after testing the prototype with users, where feedback and improvements are obtained, the time has come for the idea to be implemented and marketed.

2.2 Scrum

Schwaber [18] defines Scrum as "an Agile process or framework for Agile project management. It's a project management process, certainly not a methodology, because that would be too heavy". The methodology focus is to find a way for team members to produce software flexibly and in a changing environment. The Scrum framework is a framework where you can address and solve complex and adaptive problems while developing products that are productive and creative and have the highest possible value-added [18].

The Scrum framework is constituted in the following way: the product backlog is done through the so-called user stories (list of requirements necessary for the development of the product). The product backlog is transformed into a sprint backlog, these tasks will be developed during each sprint. It can be from 2 to 4 weeks, and each sprint backlog item corresponds to one working day, then, for each sprint the product increments.

2.3 Software Testing

As said by [3]: ”Software testing is a broad term encompassing a wide spectrum of different activities, from the testing of a small piece of code by the developer (unit testing), to the customer validation of a large information system (acceptance testing), to the monitoring at run-time of a network-centric service-oriented application.”

3 Related Work

Häger et al. [10] describe an approach that integrates DT with Scrum (DT @Scrum) with the propose of inserting DT during software development. The authors highlighted the potential for developing innovative solutions that are allowed when integrating DT with Scrum. However, they do not address the perspective of Software Testing and, more specifically, the improvement of the test case suite.

Hehn and Uebernickel [12] investigate the use of DT to support requirements engineering for projects that involve intensive and innovative software systems. The authors further indicate that investigating the integration between DT and other Software Engineering disciplines is important to understand the benefits and challenges. The authors, while addressing requirements engineering, did not analyze integration from the perspective of test case improvement.

Dobrigkeit et al. [7] propose a methodology that integrates DT, Scrum, and Lean Startup. The study does not evaluate the proposed methodology nor analyze it in the software testing scenario. In this same research line, Sohaib et al. [19] share an experience report about a framework that unifies DT with XP (eXtreme Programming). The framework has not been applied to real software projects. The authors indicate the opportunity to research DT together with Scrum.

Among existing mapping or systematic reviews of DTs there is motivation to: (1) study how DT improves software development strategies [8]; (2) articulate DT with other disciplines [13]; (3) experimentally investigate approaches involving DT in innovation processes [15]. From a Software Testing perspective, the research agenda leads to: (1) exploring the advantages and disadvantages of black-box testing aligned with other processes [1]; (2) proposing appropriate testing approaches to the required objectives (in this case, innovative features) [13]; (3) even if multiple techniques have been applied together to enhance software testing, it is necessary to explore the practicality of these [9] method combinations.

As a way to explore the practicality of combining DT, Scrum and Software Testing to improve the test suite, we planned and executed a case study.

3.1 Approach based on a mix of Scrum, Design Thinking and Software Testing

The purpose of this case study is exploratory, that is, an investigation of a particular case in various scenarios. The case study analyzed in this paper refers to the combination of DT, Scrum, and Software Testing to improve the test case suite. All this in the context of a software project with innovative features. The idea of this paper is to anticipate all inputs that will allow improving the test case suite, as a way of optimizing the testing process. For this purpose, we explore the combination of DT stages and Scrum in the testing for a biometric authentication system to retail.

4 Case Study in a biometric authentication system

This project developed a face recognition system for a large retail company. This company needed to improve its service to its customers/users by facilitating the identification in auto self totems spread across its 20 stores in four Brazilian states. This project was designed, implemented and tested in six sprints in three months.

The case analysis perspective is from the viewpoint of software testers, who want to use the combination of DT + Scrum + Software Testing to improve the test case set (test suite). Table 1 below describes each of the proposed activities, while Fig. 1 shows which activities were applied to the project.

Table 1. Activities Project

ID	Activity Name
ID01	Conducting client interviews
ID02	Investigating the features used by users in the self-service system
ID03	Researching in blog sites
	Researching papers on face recognition studies
ID04	Investigating existing face recognition architectures
ID05	Elaborating interface flow, navigation, and iteration between system screens
ID06	Creating paper usability testing and prototyping
ID07	Creating interactive prototyping
ID08	Performing paper usability testing and prototyping
ID09	Performing functional registration and facial recognition tests
ID10	Reporting defects and defect reporting system improvements
	Supporting developers and designers with possible questions about the registration process done by testers
ID11	Validating the new screen flows of the system through interactive prototyping
ID12	Modifying interface and interaction flow between system screens
ID13	Performing exploratory registration and face recognition tests
ID14	Performing guerrilla tests using the observation technique
ID15	Validating the new system screen flow by applying exploratory and guerrilla tests

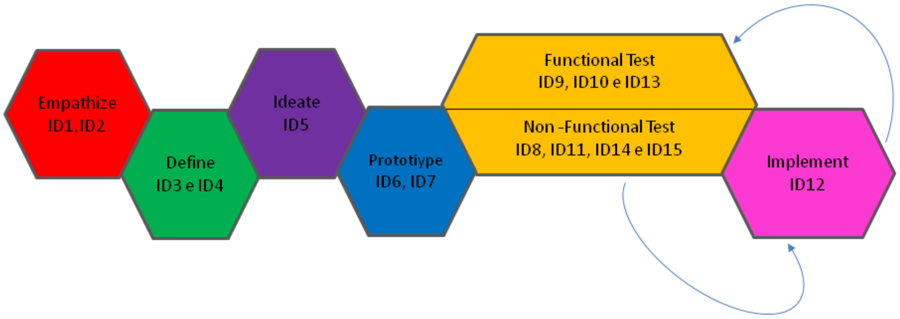

Fig. 1. Design Thinking applied to face recognition

- Empathize: this phase groups the activities ID01 and ID02. Where it is proposed to interview with the client to understand their needs and expectations. And more specifically, understanding what is expected of innovative functionality. In this phase, it is also proposed to investigate which features are most and least used in similar characteristics systems. This phase is conducted by the design team;

- Define: activities ID03 and ID04 are associated with this phase. In activity ID03, Design, Development and Validation teams propose research on websites, blogs, and journals on existing studies around innovative functionality. The activity ID04 is performed by the Development team and proposes the research of architectures related to innovative functionality;

- Ideate: phase involving activity ID05. This is an activity performed by the Design team that elaborates interface flow, navigation, and iteration between system screens;

- Prototype: phase applied by the Design team and involving activities ID06 and ID07 that are related, respectively: Creation and conducting usability testing in paper prototyping; and Prototyping and creating interactive prototyping;

- Test: this phase is characterized by grouping various activities and interaction with the implementation phase. It is performed by the design and validation teams. For a better understanding of the activities involved, it was divided into two sub-phases: Functional and Non-functional (originally present in the DT phases).
  - Functional sub-phase, being represented by ID09, ID10 and ID13 activities. In these activities, functional and exploratory tests are performed around innovative functionality. The activity ID09 produces the defect report and improvements, to the defect reporting system. Activity ID10

  supports developers and design teams for any doubts or questions about applied test records. Activity ID13 produces performing exploratory registration and face recognition tests;
  - Non-Functional sub-phase, being represented by ID08, ID11, ID14 and ID15 activities. The activity ID08 produces performing paper usability testing and prototyping. Activity ID11 validates the new system screen flow through interactive prototyping. Activity ID14 covers the implementation of the guerrilla test using the observation technique. In Activity ID15 the validation of the new flow system interfaces takes place, applying exploratory and guerrilla tests.

- Implement: Activity ID12 is found in this phase, being the responsibility of the Design team. Its main activities are modifications, interfaces and interaction flows between system screens.

4.1 Execution

For the execution of this study, we considered a self-service project, consisting of a mobile application for the Android platform and the face recognition customer authentication service. For this, the project was developed in five sprints (Sprint 0 to Sprint 4). Next, principal activities during the project are shown by specific sprints, the participating teams, and the inputs generated to improve the test suite. For each sprint, there is a table that allows you to understand the inputs generated from the integration of DT, Scrum and Software Tests.

- **Sprint 0 held from 10/23/2018 to 11/07/2018.** In this sprint are present the Empathize and Define phases. The Empathize phase consisted, as in its definition, of the participation of the Design team. In the first activity (ID01) retail store employees who use the system or know the existing self-service system were interviewed. The goal was to learn about the need to offer its users the face recognition system and what they expected to improve with this new service. As an artifact, this activity receives information from the customer and has helped to generate knowledge about the customer's needs and expectations regarding the innovative functionality of the system, in this case, face recognition. Information on customers that the system should eliminate/mitigate for possible purchase fraud was acquired for the improvement of the test suite, as well as to speed up the use of the self-service system. Then, it was possible to identify what should be included and prioritized in the test cases focused on system security.

  In the activity ID02, which seeks to investigate which features are most and least used, we use the observation and interview technique. These were carried out with retail store staff and with customers. It was possible to identify which features of the self-service system were most used and which were the least used. In addition, we identify difficulties, ease of use and existing problems. As an artifact, this activity receives which features are most used and least used by the various users in the self-service system. And it generates

Table 2. Sprint 0 - From 10/23/2018 to 11/07/2018.

Sprint 0	**Empathize** – Customer information was acquired in order to meet their expectations of providing their users with a face recognition system for self-attendant. **Team(s)** – Design	**Testing Description** – We obtained knowledge from the customers due to their interest in the self-service system and in reducing fraud. – New usability features and improvements to existing features. **Inputs generated** – Knowledge body about features to offer with the new product. It aims to support an early test requirement identification. – A set of existing features in the current self-attendant system and their related issues.
	Define – Research was conducted on facial recognition studies, self-attendant services and technical studies on face recognition. **Team(s)** – Design – Development – Validation	**Testing Description** – Planning and specification of test cases based on the most common face recognition characteristics and identification parameters to be used for face recognition. **Inputs generated** – Existing models on face recognition. – Self-attendant services. – Registration and face recognition processes.

knowledge about the features existing in the system, and its ease of use.

The purchase and payment features pf the self-service systems were identified for the improvement of the test suite. Thus, test cases should faithfully cover the main and alternative purchase and payment flows.

This Sprint also comprises the ”Define” phase, specifically in the third and fourth activity with the participation of the Design, Development and Validation teams.

For activity ID03 we highlight the research in information sources, focusing on face recognition functionality, indicated in this activity. Technical studies on face recognition were performed and, with this, it was possible to analyze contents about facial recognition and self-services. This provided knowledge of facial recognition, as well as self-service.

As an input for the improvement of the test suite were generated existing types and models of training for easy registration and recognition, as well as self-service services. This allowed us to start specifying test cases based on the most common features of facial recognition. In activity ID04 the ex-

Table 3. Sprint 1 - From 11/07/2018 to 11/14/2018.

Sprint 1	**Define** – Continuity of research on facial recognition studies initiated in sprint 0 but focusing on the difficulties detected in registering and recognizing users. – We researched on existing types of face recognition architectures **(Activity developed until Sprint 3)**.	**Testing Description** – Specification of new test cases from the difficulty scenarios detected while registering images for the face recognition system. – We have obtained knowledge of existing architectures for face recognition.
	Team(s) – Design – Development – Validation	**Inputs generated** – Knowledge and analysis of the difficulties detected during the face recognition stages. – Knowledge of existing architectures for face recognition. – Parameters configuration for testing planning.
	Ideate – Elaboration of interface flows, navigation and iteration between system flows from interviews and meeting with retail store employees and self-service users.	**Testing Description** – We received the UX and UI documents. – We analyzed the exception flows.
	Team(s) – Design	**Inputs generated** – New interface and navigation flow to be developed and specified on features.

isting architecture survey of facial recognition was performed. This activity was characterized by planning and specifying the tests after identifying the parameters to be used for face recognition. As an artifact, this activity receives several existing and applied architectures and, therefore, it is possible to generate the implementation of registration and facial recognition.

- **Sprint 1 held from 11/07/2018 to 11/14/2018.** In this sprint are present the Define and Ideate phases. Specifically, in the Define phase, the focus is on activities ID03 and ID04. For these activities, we continue to analyze and improve the knowledge database on innovative functionality. These activities further extended this sprint as a way of establishing a common vocabulary shared with the teams involved. As far as testers are concerned,

Table 4. Sprint 3- From 12/05/2018 to 12/19/2018.

Sprint 3	**Prototype** – Paper prototype usability tests were created to simulate the use of the self-service system with real users, in order to gauge the understanding of the system flow.	**Testing Description** – We got knowledge of the prototype that would be applied. – We received updates the UI and UX documents.
	Team(s) – Design	**Inputs generated** – Review test case and new specification.

Table 5. Sprint 4- From 12/20/2018 to 01/08/2019.

Sprint 4	**Test** – We performed usability tests with real users **(Activity developed until Sprint 5)**.	**Testing Description** – We obtained knowledge of the feedbacks presented by users from the usability tests applied. – Execution of functional test cases with real users to validate registration and face recognition functionalities. – We performed defect reporting and enhancements to the defect reporting system. In addition to supporting developers and designers in the possible existing doubts of the registers performed, functional test, exploratory and guerrilla tests. – We performed exploratory tests of registration and face recognition.
	Team(s) – Design – Validation	**Inputs generated** – New test case specifications and changes to existing cases were generated. – Knowledge of the defects and improvements that were evidenced for later registration. – Estimated effort to run the test suite.

this vocabulary served as input for terms to be adopted in the test cases. Since vocabulary was common, this further helped developers and designers to find it difficult to replicate test cases.

In the Ideate phase, activity ID05, from the results of previous activities, consisted of the elaboration of the interface flow, navigation, and interaction between system screens for face recognition. From interviews with retail store employees and self-service users, it was possible to identify the need to

Table 6. Sprint 5 - From 01/09/2019 to 01/22/2019.

Sprint 5	**Prototype** – Creation of interactive prototyping in order to validate interface and flow modifications suggested by real users.	**Testing Description** – Interactive prototyping was created to validate the feedback given by users during functional and exploratory tests.
	Team(s) – Design	**Inputs generated** – -
	Test – Validation of new system screen flows was performed through interactive prototyping. – The guerrilla test was applied using real users. We used the observation technique. – Exploratory tests were applied along with guerrilla tests using real users.	**Testing Description** – We obtained user feedback from the new flow showed. – We obtained feedback on the usability seeking to improve user interaction with existing face recognition functionalities. – We obtained feedback on existing face recognition bugs, enhancements and improvements to existing flows.
	Team(s) – Design. – Validation.	**Inputs generated** – Test cases update. – Participation and interaction of validation and designer teams for the improvement of flow and screen interface. – Feedback from valid users who have led change flows and updates in test cases. – New defect and improvement records were generated to describe the continuous improvement of software quality.
	Implement – We performed modification of interface and interaction flow between the system screens, according to the feedback obtained in the tests.	**Testing Description** – Review of test cases based on modifications implemented for FR.
	Team(s) – Design.	**Inputs generated** – Knowledge of changes made to new screen flows and revision of test case specifications.

create new screen flows and navigation between them. In this activity it was evident the need to balance the expectations of the customer (the retailer) and potential users of the system. The documents generated by the designers that contained the flows of user interaction and navigation between screens was used as input for the improvement of the test suite. This facilitated the description of expected flows in the test cases and the gestures with which it would be possible to interact with the software. These inputs were also used as criteria for defining the set of inputs from the perspective of interaction gestures for the test cases.

Sprint 2 (11/15/2018 - 12/4/2018) involves activities developed and described in Sprints 1 and 3.

- **Sprint 3 held from 12/05/2018 to 12/19/2018.** In this Sprint the Prototype phase is applied through activities ID06 and ID07. Activity ID06 consists of creating paper prototypes that allow an initial usability test. The prototypes focused on the screens and flow of innovative functionality: customer recognition and facial identification. In activity ID07, interactive prototypes to be used in the Test phase (Sprint 4) were obtained from feedback gathered using the paper prototype. This Sprint was focused on prototyping so that screen and interaction flows could be validated by users. Thus, this sprint was not focused on generating any input for the improvement of test cases. However, with the usability tests applied it was possible to verify the proposed flows in the previous Sprints. This allowed the refinement of the flows described in the test cases.

- **Sprint 4 held from 12/20/2018 to 01/08/2019.** In this sprint, the focus is on the Test non-functional phase, which is the nature of the DT through the ID08, ID11, ID14 and ID15 activities. These cycles don't test the other non-functional related activities, as these involve other flows.

  Considering the non-functional part of the Test phase, during these activities, both the paper prototype and the interactive prototype, generated in the previous sprint, were presented to internal employees as a way of evaluating whether the defined flow would meet the actual interaction flow with the face recognition system. With this, it was possible to update the knowledge database around the perceptions about the system use. The prototypes also made it easier to exercise the previously defined test suites. In this case, we generate a sets of defect reports related to the system represented in the prototypes.

  As new defects and flow improvements were identified, it was necessary to rerun the activity ID07 in order to update the interactive prototype and apply new interactive usability tests until the registration and facial recognition was showing a correct and clear flow to the user.

  Once the prototype was defined, the functionalities and flows defined were implemented, a release for the validation team was generated and cycles of functional, exploratory and guerrilla tests were carried out.

Thus, there is a set of inputs generated in this sprint: (1) new test case specifications and refinements in existing ones; (2) reporting defects and improvements, and; (3) estimated effort to run the test suite.

- **Sprint 5 held from 01/09/2019 to 01/22/2019.** This sprint focuses on the Test and Implement phases. In the Test phase we exercised both the non-functional form through the activities (ID08, ID11, ID14 and ID15) performed in the previous sprint, as well as the functional form with the activities (ID09, ID10 and ID13) performed in the current sprint. This sprint also covers the implement phase that is exercised by activity ID12. It is important to highlight that activities related to functional testing can only be performed after activity ID12, as it involves the implementation of functionalities in the software. For each Test Phase activity we then performed activity ID12 for the Implement Phase. This activity generated input for the review of test cases based on the modifications focused on registration and facial recognition. All other functional test activities (ID09, ID10 and ID13) were performed after cycle releases.

5 Conclusion and Future Work

Findings from our study indicated that combining DT, Scrum and Software Testing supported in: (1) Analyzing testing requirements at the same time as functional and non-functional software requirements including user feedback as a validation strategy; (2) Identifying the high importance interaction flows around the innovative feature, specifically, features related to the innovative aspect; (3) Mapping a set of models and architectures as a way to refine the parameter that can be used on input set and expected results in a Test Suite; (4) Establishing a common vocabulary among the team members. This helped in understanding and replication of the Test Suite; (5) Refining the input set (gestures) on Test Suite description; (6) Validating and enhancing the test case flows; (7) Early prototyping in interactive mode, more than paper prototype, helped the testers to perform functional tests; (8) Introducing innovation features into software development helped to discover how cultural, physical and demographic characteristics can be important to enhance the test suite.

As future work we are applying the approach in other projects with innovative features. It can help to refine the approach and list other gaps and benefits from combining DT+Scrum+Software Testing. We also plan to apply a set of interviews with team members in order to obtain a qualitative perspective from the approach used.

References

1. Ahmad, J., ul Hassan, A., Naqv, T., Mubeen, T.: A review on software testing and its methodology. i-Manager's Journal on Software Engineering **13**(3), 32 (2019)

2. Beck, K., Beedle, M., Bennekum, A.v., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al.: Manifesto para desenvolvimento ágil de software. Retirado em **20** (2001)
3. Bertolino, A.: Software testing research: Achievements, challenges, dreams. In: 2007 Future of Software Engineering. pp. 85–103. IEEE Computer Society (2007)
4. Beverland, M.B., Wilner, S.J., Micheli, P.: Reconciling the tension between consistency and relevance: design thinking as a mechanism for brand ambidexterity. Journal of the Academy of Marketing Science **43**(5), 589–609 (2015)
5. Bonini, L.A., Sbragia, R.: O modelo de design thinking como indutor da inovação nas empresas: um estudo empírico. Revista de Gestão e Projetos-GeP **2**(1), 03–25 (2011)
6. Brown, T., et al.: Design thinking. Harvard business review **86**(6), 84 (2008)
7. Dobrigkeit, F., de Paula, D., Uflacker, M.: Innodev: A software development methodology integrating design thinking, scrum and lean startup. In: Design Thinking Research, pp. 199–227. Springer (2019)
8. Elsbach, K.D., Stigliani, I.: Design thinking and organizational culture: A review and framework for future research. Journal of Management **44**(6), 2274–2306 (2018)
9. Garousi, V., Mäntylä, M.V.: A systematic literature review of literature reviews in software testing. Information and Software Technology **80**, 195–216 (2016)
10. Häger, F., Kowark, T., Krüger, J., Vetterli, C., Übernickel, F., Uflacker, M.: Dt@ scrum: integrating design thinking with software development processes. In: Design thinking research, pp. 263–289. Springer (2015)
11. Haghighatkhah, A., Mäntylä, M., Oivo, M., Kuvaja, P.: Test prioritization in continuous integration environments. Journal of Systems and Software **146**, 80–98 (2018)
12. Hehn, J., Uebernickel, F.: The use of design thinking for requirements engineering: An ongoing case study in the field of innovative software-intensive systems. In: 2018 IEEE 26th International Requirements Engineering Conference (RE). pp. 400–405. IEEE (2018)
13. Hrabovská, K., Rossi, B., Pitner, T.: Software testing process models benefits & drawbacks: a systematic literature review. arXiv preprint arXiv:1901.01450 (2019)
14. López-Martínez, J., Juárez-Ramírez, R., Huertas, C., Jiménez, S., Guerra-García, C.: Problems in the adoption of agile-scrum methodologies: A systematic literature review. In: 2016 4th International Conference in Software Engineering Research and Innovation (CONISOFT). pp. 141–148. IEEE (2016)
15. Micheli, P., Wilner, S.J., Bhatti, S.H., Mura, M., Beverland, M.B.: Doing design thinking: Conceptual review, synthesis, and research agenda. Journal of Product Innovation Management **36**(2), 124–148 (2019)
16. Myers, G.J., Sandler, C., Badgett, T.: The art of software testing. John Wiley & Sons (2011)
17. Pugh, K.: Lean-Agile Acceptance Test-Driven-Development: Better Software Through Collaboration. Pearson Education (2010)
18. Schwaber, K., Sutherland, J.: Guia do scrum-um guia definitivo para o scrum: As regras do jogo. Obtido em **16** (2013)
19. Sohaib, O., Solanki, H., Dhaliwa, N., Hussain, W., Asif, M.: Integrating design thinking into extreme programming. Journal of Ambient Intelligence and Humanized Computing **10**(6), 2485–2492 (2019)
20. Tahera, K., Wynn, D.C., Earl, C., Eckert, C.M.: Testing in the incremental design and development of complex products. Research in Engineering Design **30**(2), 291–316 (2019)