Guidelines for the adoption of Behavior-Driven Development (BDD): An approach with Design Science Research

Shexmo Richarlison Ribeiro dos Santos[1], Fabio Gomes Rocha (Supervisor)

[1]Federal University of Sergipe (UFS)
Master's Program in Computer Science
Start: March 2023, End December 2024
Proposal approval (qualification): January 2024
49.100-000 – São Cristóvão – SE – Brasil

shexmor@gmail.com, gomesrocha@gmail.com

***Abstract. Context**: In software development, looking for ways to save time and systematize the process is necessary. **Problem**: Lack of clarity regarding the aspects necessary for adopting BDD. **Solution**: Present guidelines for implementing BDD. **Method:** Identify and present via a mind map such guidelines through a Design Science Research approach. **Expected contribution**: The potential assistance in improving processes and software products through the correct adoption of BDD.. The link to the video with additional information is: https://youtu.be/TQ2pOc8G1xg.*

1. Introduction

The software architect is the professional responsible for building software, considering internal and external factors such as the architectural standard and the clients, respectively [Kruchten(2008)]. For an architect to successfully develop software, one must have knowledge of the aspects inherent to its creation, such as the existing frameworks, to achieve the objectives aimed at maintaining software quality.

A framework is defined as a structure that serves as a basis for the construction of applications to be developed systematically. There is also a need for the frameworks used in software architecture to be able to meet such expectations focused on agile methods, for example, the framework Behavior-Driven Development (BDD), used in the life cycle of a system [Bruschi et al.(2019)].

BDD focuses on behaviour according to what the software is expected to perform, so there is better interaction between those involved in requirements elicitation to contribute to greater assertiveness in the delivery of releases. The BDD structure defines software behaviour using the "given-when-then" pattern, which can be expressed in natural language, specific to the domain and then executed through automated tests. In this way, there is better communication between those involved in the process to improve the delivery of the final product concerning the validation of the elicited requirements.

Since BDD is a framework used from requirements elicitation to software implementation and maintenance, communication between stakeholders and the development team is necessary to contribute to the quality of the software delivered. Thus, when using BDD in software development, the people involved are essential for BDD to achieve its objective proposed by Dan North [North(2006)]: a framework that assists in communication and delivers releases according to expected behaviour. This study involves people, process and technique to maintain the system's quality as a whole.

According to Brooks [Brooks and Bullet(1987)] no single software meets all requirements; software is built based on customer needs and needs to maintain the quality inherent in its execution. Furthermore, according to Boehm [Boehm(2006)], over time, software needs to meet new demands, so there is a need to seek to achieve what the customer expects regarding its final product.

To seek improvements regarding the development and evaluation of software [Shaw(2002), Shaw(2003)], the objective of this research is to "**Analyze** studies on BDD, **with the purpose of** characterization, **with respect to** adoption and application, **from the point of view of** researchers and practitioners, **in the context of** software development". To achieve the proposed objective, we defined the following research questions:

- What is the current state of the art of BDD?
- What is the current state of practice of BDD?
- Are the discovered guidelines replicable in teams that do not use BDD?

To answer the research questions, we used the Design Science Research methodology to subdivide our study into three knowledge bases: Systematic Literature Review, Survey and Case Study - ISO/IEC/IEEE 25010 Standard. Based on the results found in these studies, we will be able to carry out a case study with a team that does not use BDD to validate the guidelines identified by previous studies regarding the adoption of this framework.

2. Behavior-Driven Development (BDD)

In 2003, Dan North [North(2006)] created the Behavior-Driven Development (BDD) framework, used throughout the software lifecycle. BDD is a framework that focuses on system behaviour. Written in *gherkin* language, the framework has gained notoriety among both researchers and industry, bringing benefits such as better communication and reduced time in delivering releases, in addition to greater assertiveness in eliciting requirements [Pereira et al.(2018)].

BDD is proposed through the "3 amigos" technique, where a meeting is held between the product owner, tester and developer to specify objectively and concisely the behaviour expected by the system when eliciting the requirements, in this case, the user stories and their acceptance scenarios, to outline the behaviour expected by the software. Since such stories must be written directly and clearly to achieve the proposed objective, BDD needs to make its scenarios testable [Silva and Fitzgerald(2021)]. Thus, it is clear how BDD can collaborate to elicit non-functional requirements [Santos et al.(2024)].

According to Bruschi *et al.* [Bruschi et al.(2019)] improved communication through BDD results in better collaboration between those involved in the process since the requirements are standardized by the "3 amigos" to also help in the documentation and automation of tests and factors necessary for validation. Furthermore, with the time saved by reducing rework, the team can focus on other activities inherent to the software. Therefore, after outlining the requirement to be implemented, the next step is to write it using BDD following the *gherkin* language.

From this point on, the functionalities begin to be written, implemented and validated, seeking to maintain the objective expected by the requirements elicited in each release to deliver what is expected by the system. In addition to assisting in the communication and collaboration of the parties involved [Couto et al.(2022)], because it is written in simple language, BDD also helps in the dynamic documentation of the system [Pereira et al.(2018), Silva and Fitzgerald(2021), Nascimento et al.(2020)].

Living documentation refers to the agility required for updates in process documentation. With such aspects, it is expected that the use of BDD will bring savings in time and money, as it reduces the need for rework due to poor quality releases, has better quality validation [Binamungu et al.(2018), Moe(2019)], in addition to a better understanding of the code [Guerra-Garcia et al.(2023)].

3. Design Science Research

Design Science Research is a methodology that has been used over the years in the area of Information Systems [Horita EA et al.(2020)]. According to Hevner *et al.* [Hevner et al.(2010)], research in Information Systems is composed of two strands: Design Science and Behavioral Science. Furthermore, the authors reported that both strands are complementary; thus, they correlate, as shown in Figure 1.

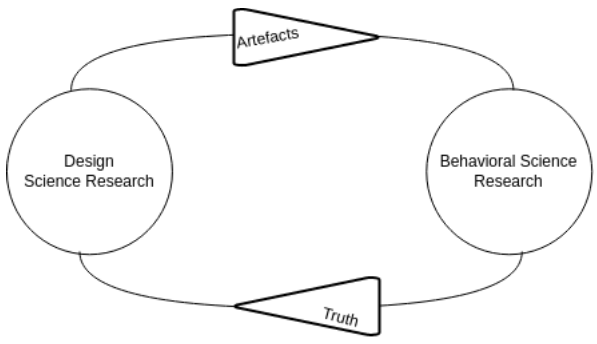

Figura 1. Relationship between Design Science and Behavioral Science

Design Science Research seeks to create innovative artefacts through methods inherent to this type of research to improve the artefacts' effectiveness and usefulness in the real context [Horita EA et al.(2020)]. While Design Science seeks to deliver artefacts, Behavioral Science, through the delivered artefacts, seeks to present the truth about the adoption of the artefacts. Thus, the correlation between the two approaches is perceived as being at different stages of the same cycle. Therefore, Figure 2 presents the methodology approach inherent to this study.

Through a systematic literature review, we identified how BDD has been researched and used to explain the perspective of researchers and professionals. With the results found, we conducted a survey that presented the point of view of professionals who use BDD in their work routines to explain their perception regarding this framework. Furthermore, a case study identified how BDD elicited non-functional (quality) requirements through the performance efficiency characteristic expressed in the ISO/IEC/IEEE 25010 Standard [Santos et al.(2024)].

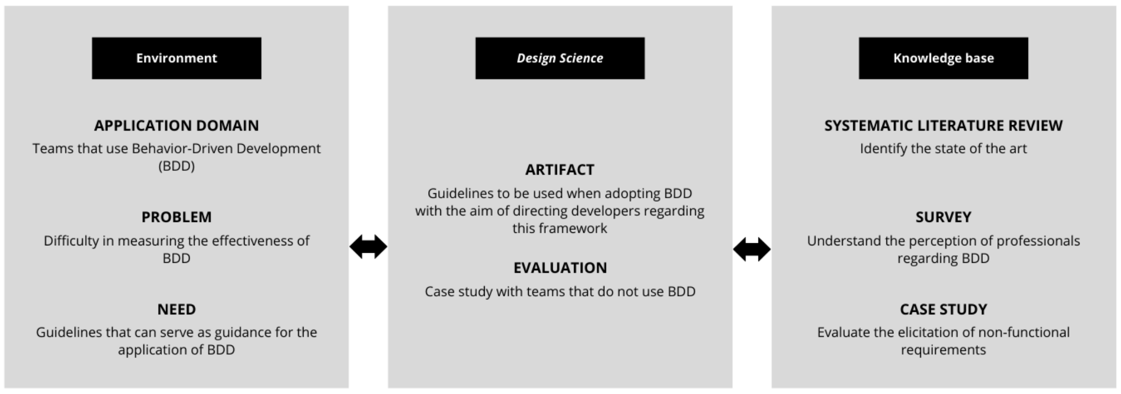
Environment
APPLICATION DOMAIN
Teams that use Behavior-Driven Development (BDD)
PROBLEM
Difficulty in measuring the effectiveness of BDD
NEED
Guidelines that can serve as guidance for the application of BDD
Design Science
ARTIFACT
Guidelines to be used when adopting BDD with the aim of directing developers regarding this framework
EVALUATION
Case study with teams that do not use BDD
Knowledge base
SYSTEMATIC LITERATURE REVIEW
Identify the state of the art
SURVEY
Understand the perception of professionals regarding BDD
CASE STUDY
Evaluate the elicitation of non-functional requirements

Figura 2. Design Science Research

Through these studies, we could outline guidelines for adopting BDD, which will finally be validated through a case study with a team that does not use this framework.

4. Partial results

Below are summaries of the partial results found in the studies inherent to this research.

4.1. Systematic Literature Review

We identified the main points related to the adoption of BDD, from the study methodology to where it is most applied and with which tool[1]. According to Shaw [Shaw(2002)], it is used in Software Engineering as a research strategy to analyze aspects of software development by creating an analytical model, generally, and then validating it in a formal analysis or user experience. Thus, we could understand the main characteristics related to BDD (analytical model) when we perform a Systematic Literature Review. Through the Survey, we could validate our results with professionals who use this framework.

4.2. Survey

Through the data obtained in this study, it was possible to observe the point of view of 43 professionals who use BDD in their work activities to characterize this framework, making it possible to advance the understanding regarding BDD and also strengths and weaknesses inherent to its adoption. We inferred that it is necessary to have experience using BDD to achieve the potential expected by the framework, so its adoption has benefits such as quality in delivery and readability. The lack of experience in using BDD directly impacts the performance of work activities, so the more experience is gained in using the framework, the fewer situations will be considered harmful regarding its adoption.

4.3. Case Study - ISO/IEC/IEEE 25010 Standard

BDD presented positive results from eliciting non-functional requirements, carrying out objectively among team members, and automating tests to meet the proposed acceptance

[1]This study has already been approved at the Simposio Argentino de Ingeniería de Software presented at Jornadas Argentinas de Informática (JAIIO 53). We are waiting for the proceedings to be published.

criteria [Santos et al.(2024)]. Regarding the elicitation stage of non-functional requirements, since they are considered more complex, it is expected to have difficulty carrying out this stage. Through the *gherkin* language used by BDD, there was a better understanding of what to expect from the behaviour of the software, so the user stories were described with a clear purpose. Through this study, the necessary steps for eliciting non-functional requirements were perceived to present the need to validate the requirement through user stories and their respective acceptance criteria and to identify whether what was expected by the software was achieved. Thus, this research contributed to understanding the elicitation of non-functional requirements with BDD, which was used in practice in a real situation.

5. Future steps

Finally, we will conduct a case study with teams that do not use BDD to validate the guidelines we identified through the Design Science Research approach. We aim to help professionals who use BDD in their work activities and researchers who seek to understand and research this framework.

Referências

Leonard Peter Binamungu, Suzanne M Embury, and Nikolaos Konstantinou. 2018. Maintaining Behaviour Driven-Development specifications: Challenges and opportunities. In 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER). IEEE, 175–184.

Barry Boehm. 2006. A view of 20th and 21st century software engineering. In Proceedings of the 28th international conference on Software engineering. 12–29.

Frederik P Brooks and No Silver Bullet. 1987. Essence and accidents of software engineering. IEEE computer 20, 4 (1987), 10–19.

Stefania Bruschi, L Xiao, M Kavatkar, et al. 2019. Behavior Driven-Development (BDD): a case study in healthtech. In Pacific NW Software Quality Conference.

Thiciane Couto, Sabrina dos Santos Marczak, Daniel Antonio Callegari, Michael Móra, and Fábio Rocha. 2022. On the Characterization of Behavior-Driven Development Adoption Benefits: A Multiple Case Study of Novice Software Teams. Anais do XXI Simpósio Brasileiro de Qualidade de Software, 2022, Brasil. (2022).

Cesar Guerra-Garcia, Anastasija Nikiforova, Samantha Jiménez, Hector G. Perez-Gonzalez, M. T. Ramírez-Torres, and Luis Ontañon-García. 2023. ISO/IEC 25012-based methodology for managing data quality requirements in the development of information systems: Towards Data Quality by Design. Data and Knowledge Engineering 145 (2023), 102152–102152. https://doi.org/10.1016/j.datak.2023.102152

Alan Hevner, Samir Chatterjee, Alan Hevner, and Samir Chatterjee. 2010. Design science research in information systems. Design research in information systems: theory and practice (2010), 9–22.

Flávio Horita EA, Fabio Gomes Rocha, Layse Santos Souza, and Gustavo R Gonzales. 2020. Design science in digital innovation: A literature review. In XVI Brazilian Symposium on Information Systems. 1–7.

Philippe Kruchten. 2008. What do software architects really do? Journal of Systems and Software 81, 12 (2008), 2413–2416.

Myint Myint Moe. 2019. Comparative Study of Test-Driven Development TDD, Behavior-Driven Development BDD and Acceptance Test–Driven Development ATDD. International Journal of Trend in Scientific Research and Development 3 (2019), 231–234.

Nicolas Nascimento, Alan R Santos, Afonso Sales, and Rafael Chanin. 2020. Behavior-Driven Development: an expert panel to evaluate benefits and challenges. In Proceedings of the XXXIV Brazilian Symposium on Software Engineering. 41–46.

Dan North. 2006. Introducing BDD.

Lauriane Pereira, Helen Sharp, Cleidson de Souza, Gabriel Oliveira, Sabrina Marczak, and Ricardo Bastos. 2018. Behavior-Driven Development benefits and challenges: reports from an industrial study. In Proceedings of the 19th International Conference on Agile Software Development: Companion. 1–4.

Shexmo Santos, Tacyanne Pimentel, Fabio Gomes Rocha, and Michel S. Soares. 2024. Using Behavior-Driven Development (BDD) for Non-Functional Requirements. Software 3, 3 (2024), 271–283. `https://doi.org/10.3390/software3030014`

Mary Shaw. 2002. What makes good research in software engineering? International Journal on Software Tools for Technology Transfer 4 (2002), 1–7.

Mary Shaw. 2003. Writing Good Software Engineering Research Papers. In 25th International Conference on Software Engineering, 2003. Proceedings. 726–736.

Thiago Rocha Silva and Brian Fitzgerald. 2021. Empirical findings on BDD story parsing to support consistency assurance between requirements and artifacts. In Evaluation and Assessment in Software Engineering. 266–271.