On the Understanding of BDD Scenarios' Quality: Preliminary Practitioners' Opinions

Gabriel Oliveira and Sabrina Marczak

Computer Science School, PUCRS, Porto Alegre, Brazil
gabriel.pimentel@acad.pucrs.br,sabrina.marczak@pucrs.br

Abstract. [**Context & Motivation**] In agile development, acceptance tests are written to express the details from the conversations between customers and developers. One of the formats to express those details is BDD (Behavior-Driven Development) scenarios, which use a ubiquitous language, one that business and technical people can understand, to build an executable specification that represents a system behavior. [**Question/Problem**] Problems caused by bad documentation are known to cause project failure and we believe those problems apply to documentation in the format of acceptance tests as well. Thus, in the long-term, we seek to understand what would be the definition of a good BDD scenario and the criteria to define it. [**Principal idea/results**] To achieve that, we previously identified known requirements' quality attributes that would be suitable to evaluate BDD scenarios' quality. Based on that list of attributes, we now aim to validate that list with practitioners, identify their interpretation of the listed attributes, and uncover general recommendations to write BDD scenarios. [**Contribution**] Preliminary results from our initial set of interviews revealed practitioners' interpretations for consistent, testable, valuable, understandable, and unambiguous attributes and some recommendations to write good BDD scenarios, such as the use of declarative form of writing.

Keywords: documentation quality, documentation evaluation, behavior-driven development, empirical study

1 Introduction

Most agile methodologies represent requirements using user stories. Cohn [1] states that a user story card is an expression of the essential elements of a requirement - it has just enough information to remind everyone what the story is about. A verbal conversation takes place to refine that customer requirement. When the conversation gets down to the details, the customer and the developer specify what needs to be done in the form of acceptance tests. Bjarnason et al. [2] clarify this agile approach of integrating requirements engineering with testing stating that the conceptual difficulty of specifying tests before implementation led to the conception of Behavior-Driven Development (BDD) – an approach that incorporates aspects of requirements analysis, requirements documentation and communication, and automated acceptance testing.

BDD is an agile practice which uses a language that business and technical people can understand to describe and model a system [3]. The model is formed by a series of textual scenarios, expressed in a format known as Gherkin, designed to be easily understandable for all stakeholders and easy to automate using dedicated tools. The scenarios related to a particular feature are grouped into a feature file, that contains a short description of the feature and the scenarios that compose it. Each scenario is made up of a number of steps, where each step starts with one of a small number of keywords. The natural order of a scenario is *Given... When... Then...*, where *Given* steps describe the preconditions for the scenario and prepares the test environment, *When* steps describe the action under test and *Then* steps describe the expected outcomes.

It is well known that bad requirements are one of many potential causes of a project failure [4] and that bad scenarios documentation can lead to misleading information that will negatively impact the tests ability to reflect the system coverage and the team confidence on them [5]. Therefore, we judge it necessary to better understand how we can prevent BDD textual scenarios to suffer from problems caused by bad documentation by proposing guidelines on how to write good BDD scenarios [6]. To the best of our knowledge, BDD scenarios practitioners can only rely on a few guidelines and examples of good and bad scenarios provided by Smart's experience reported on his book [3].

Requirements are evaluated by a set of quality attributes. The Business Analyst Body of Knowledge (BABOK) [7] brings nine attributes a traditional requirement must have in order to be a quality one, as follows: atomic, complete, consistent, concise, feasible, unambiguous, testable, prioritized, and understandable. Also, the INVEST (Independent-Negotiable-Valuable-Estimable-Scalable-Testable) framework described by Cohn [1] is often used to evaluate for user stories. Lucassen et. al [8] argue that qualitative metrics are not sufficient to evaluate user story quality employ highly. Due to that fact, they define additional criteria to evaluate user stories on their QUS Framework. We preferred to clarify the BABOK and INVEST attributes interpretation on BDD scenarios before using other frameworks such as QUS.

Our on-going research has the goal to uncover what a good BDD scenario is. The first steps of our on-going research on this topic were to acquire known quality attributes in a literature review and to use them on a pilot study with graduate students to understand how novice evaluators use those attributes to judge the quality of BDD scenarios [6]. Based on that list of filtered known quality attributes, we are now seeking to understand what are practitioner's interpretation of those attributes when reading BDD scenarios and their recommendations on how to write good BDD scenarios. To achieve that, we acquire practitioners personal criteria, tied to textual scenario's details, and ask them how each of their criteria map to our list of known quality criteria.

This paper aims to highlight our preliminary findings from the already conducted interviews, that partially fulfill our goal. Therefore, the following sections present our empirical study and results.

2 Research Method

Our long-term research has the goal to uncover the concept of quality in BDD textual scenarios, as summarized in Figure 1, based on the opinion of practitioners involved in industry projects that have or had been using BDD scenarios. Their quality criteria, the rationale behind their opinions and their interpretation of known quality attributes will be used to consolidate our understanding of BDD scenarios quality. In order to aid us overcome the challenge of interpreting their different opinions, often phrased in different ways, we believe it would be best to guide the conversation with a set of known quality attributes and with real BDD scenario examples.

Therefore, our first step to achieve our goal, shown in Figure 1 Step (A), was to discover the list of quality attributes used on agile requirements, which brought us the following attributes from the BABOK [7] and INVEST [1]: atomic, complete, consistent, concise, estimable, feasible, independent, negotiable, prioritized, small, testable, understandable, unambiguous, and valuable.

Our second step, shown in Figure 1 Step (B), was to actively use those known quality attributes with BDD scenarios and better understand what would be the challenges on this process. To that end, we organized a pilot study with novice practitioners [6], that indicated that some attributes in the list may not be suited for BDD scenarios individually (like *complete* or *consistent*) or may be seen as a confusion source to the evaluator (like *atomic* or *independent*). Therefore, the refined list of attributes to use on BDD scenarios are: concise, estimable, feasible, negotiable, prioritized, small, testable, understandable, unambiguous, and valuable.

We are now using those attributes on the interviews with practitioners as shown in Figure 1 Step (C). During the 8 already conducted interviews, notes were taken to summarize practitioners personal criteria, their recommendations on how to write good BDD scenarios, what they think about those known quality attributes, and how they interpret each one – mapping their personal criteria into those known quality attributes.

Our subsequent analysis was guided by grounded theory procedures [9]. The interviewer conducted open-coding on the recording of each interview, just after they were finished, with the objective of identifying participants' rationale and summarize their personal quality criteria, recommendations on how to write good BDD scenarios, opinions about those known quality attributes, the interpretation

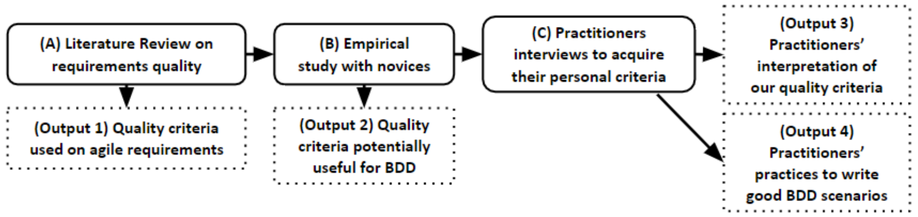
(A) Literature Review on requirements quality
(B) Empirical study with novices
(C) Practitioners interviews to acquire their personal criteria
(Output 1) Quality criteria used on agile requirements
(Output 2) Quality criteria potentially useful for BDD
(Output 3) Practitioners' interpretation of our quality criteria
(Output 4) Practitioners' practices to write good BDD scenarios

Fig. 1. Research Design

of each one, and the mapping of their personal criteria into those known quality attributes. The interviewer codes were discussed with the second author and refined. That summary has been the input used to generate our preliminary results in Section 3.

Additionally, the pilot study have shown us the need to guide conversations with known examples to aid practitioners realize their own quality criteria. To avoid our own bias towards what would be a good or bad BDD scenario, we decided to not create the examples ourselves. Instead, we handed them real BDD scenarios, taken from an open source project that employ BDD scenarios to detail their applications' behavior. Our project of choice was Diaspora[1], a decentralized social network with a list of feature files mapping the behavior of the different application screens. To the best of our knowledge, Diaspora is Github's open source project with the most feature files available.

From the 8 interviewed practitioners (ranging from P1 to P8), some of them (P4, P8) are responsible for the scenarios' refinement, some (P1, P2, P5, P6, P7) are also responsible for the coding of automated checks, and one (P3) is responsible for the creation of scenarios and application code.

Some interviewees (P5, P6) had less than a year of experience using BDD scenarios, while others had up to 3 years (P1, P2, P4, P7) or up to 10 years (P8). Some of them (P1, P4) write scenarios after meetings with developers and product owners, but the majority (P2, P3, P5, P6, P7, P8) write scenarios themselves and validate later with the team. Regarding gender, we had a majority of male interviewees (P2, P3, P4, P6, P7, P8). Finally, almost half of the interviewees (P2, P6, P8) were self-employed consultants, while some (P4, P5) worked in big companies (more than 3000 employees). Others (P3, P7) worked in small companies (couple hundreds employees) and one (P1) worked at a startup (less than 50 employees).

One threat to the validity of our interview-based study is that the main analysis was done by a single researcher, the one doing the interviews. That researcher bias may have directed the observations in unexpected ways. To address this limitation, the generated codes were reviewed by the second author. Also, another threat to be acknowledged is the fact that the most of our 8 interviewees belonged to the same role. We plan to expand both the number of interviews and our role coverage in the following months.

3 Results

For the interviewed practitioners, each BDD scenario should: have a single goal – to validate a single business rule; achieve that goal using a few steps; and express those steps in a domain specific language, natural to business people. Our summary from their collective quality attributes interpretations is represented in Table 1, along with their recommendations – separated into good writing patterns to be followed and bad writing patterns to avoid.

[1] `https://diasporafoundation.org/`

Table 1. Summary of quality attributes for BDD scenarios

Attribute	**Interpretation**	**Bad Patterns**	**Good Patterns**
Concise	"To the point", few and small steps	Unnecessary details Mixing steps order Data tables	Declarative writing Short statements Only essential details
Testable	Single and clear goal and clear outputs	Keyword Repetition Mixing steps order	Declarative writing Title matching Then 1 or 2 Given Steps Only 1 Then step
Understandable	Consistent use of business terms	Technical jargon Mixing steps order Data tables	Declarative writing Data tables Fictional characters
Unambiguous	Single action, scenario cover one behavior	Mixing steps order Keyword Repetition Weak words	Only 1 When step Fictional characters
Valuable	Why this scenario exist	Mixing steps order	Expressive feature and scenario description

Estimable, *feasible*, *negotiable* and *prioritized* were judged not fit to evaluate scenarios by all practitioners, although P2 and P3 declared *prioritized* suited to conceptual features (often referred as an epic, a group of user stories). Also, all those attributes demand a domain knowledge of the product that cannot be found in a scenario's textual description.

For the majority of interviewees (P2, P4, P5, P6, P8) *valuable* attribute was on that category as well, as the value of a scenario depends on who one asks and the context it is being used. Others (P7, P3, P1) argued that scenarios titles and the feature description should indicate how valuable each scenario by stating "why" each should be there.

Ambiguity in scenarios was sometimes (P1, P7) referred to the understanding that two scenarios test the same thing - thus, making sure all scenarios test different aspects of the feature and that they together cover the entire feature would make scenarios *unambiguous*. For some others (P2, P3, P4, P8), using weak words such as "something good" mark a scenario as ambiguous. For P6, scenario's coverage of the feature should also be mapped into unambiguous attribute. Finally, P5 judged a scenario *unambiguous* if it was *understandable*.

Understandable attribute was often (P1, P2, P3, P5, P6, P7, P8) interpreted as the act of writing scenarios in a business language, using business terms in a consistent way between scenarios, leaving implementation details out of the textual descriptions. As scenarios are meant to be used by technical and business people, their description should not contain technical details such as HTTP response codes. For P4, an *understandable* scenario is a *small* one.

Testable attribute represent that the scenario intention should be clearly stated (P2, P4), only one behavior should be tested (P6) and the *Then* clause should be reflected on the title (P5). However, others (P3, P7, P8) do not see this attribute as applicable to BDD scenarios, as it's a product characteristic. P1 have stated that *testable* is the same as *understandable*

Differences between the *concise* and *small* attributes were not clear, thus we joined them in Table 1. Some interviewees (P2, P3, P7, P8) said they were supposed to be equal, others declared that *small* scenarios are those with few number of steps – between 4 or 5 steps (P1) – while *concise* is more related with "to the point" (P4, P5, P6) statements – short phrases, fitting into the reader screen, that does not carry unnecessary details.

Some bad patterns were identified by participants on Diaspora's feature files. The disregard for the natural order of steps (Given/When/Then) hurt *concise* (P2, P3, P6, P8), *testable* (P2, P7), *understandable* (P2, P4, P8), *unambiguous* (P3), and *valuable* (P3) attributes. The multiple use of a step affect each attribute differently: multiple *Given* steps show that the scenario may not be *testable* due to the many dependencies that need to be set up (P6, P7); multiple *When* steps makes a scenario purpose more *ambiguous* (P7); multiple *Then* steps gave the impression that more than one business rule was being tested at once, hurting the scenario' single goal represented by the *testable* (P3) attribute or even making it more *ambiguous* (P4). Also, representing input and output data into tables harm the scenario's *concise* (P1) and *understandable*(P7) attributes.

An identified good pattern was that writing scenarios into a declarative way would make it more *testable* (P1, P2, P5), *concise* (P8), and *understandable* (P7), preventing changes on the user interface to force a change on scenario's descriptions. Another good pattern is representing common user characteristics using fictional character names to represent a type of user or a role in the system. Referring to those fictional characters using third person speech would make the scenario more *understandable* (P3, P7) and *unambiguous* (P6).

Some interviewees had identified missing characteristics, such as complete (for P1, P4, and P6, scenarios on a feature file should cover all aspects under test), independent (for P6, scenarios should be read and executed in any order), atomic (for P6, scenarios should describe one thing), unique (for P3, it should group together *testable*, *understandable*, *unambiguous*, and *valuable*), modular (for P3, it should group together *small* and *testable*), and ubiquitous (per P8, having a consistent way of writing, is only partially covered by *understandable*). However, those characteristics occurrences were almost individual ones.

Finally, most of the interviewees (all, except P5) judged that having a list of attributes would be helpful, as those words make it easier to talk about scenarios quality, help express their informal criteria in a better way and work as a validation checklist to help a reviewer. P6 warned us that a long criteria is harmful - 5 or 6 should suffice. Another interviewee (P5) had not agreed that a list would be helpful - BDD technique require one to talk to people and make them agree upon some scenario descriptions, thus narrowing one's view to a limited set of quality attributes would cause more harm than good. Additionally, for P8, scenarios writing is a team exercise and represents a team consensus and there would be no need to have a perfect scenario – a good enough scenario, written by many hands, would reach the BDD technique goal of building a single system model shared by business and technical people.

4 Conclusion

This paper presents our preliminary findings to support our on-going research's goal, revealing practitioners' interpretations for and some recommendations to write good BDD scenarios taken from our initial set of interviews. We believe that our on-going effort will yield the necessary information to effective use those attributes on BDD scenarios. In addition to running more interviews to consolidate the presented results, another next step is to represent their interpretations in a series of questions, mixed with their recommendations, as an improved guide to identify how good a scenario is.

Acknowledgments. The results presented in this paper were achieved in cooperation with Hewlett Packard Brasil LTDA, using incentives of the Brazilian Informatics Law (Law n^o 8.2.48 of 1991).

References

[1] Cohn, M.: User Stories Applied: For Agile Software Development. Addison Wesley Longman Publishing Co., Inc. (2004)

[2] Bjarnason, E., Unterkalmsteiner, M., Borg, M., Engström, E.: A multi-case study of agile requirements engineering and the use of test cases as requirements. Information and Software Technology **77** (2016) 61–79

[3] Smart, J.: BDD in Action: Behavior-Driven Development for the Whole Software Lifecycle. Manning Publications, Shelter Island, USA (2014)

[4] Kamata, M.I., Tamai, T.: How does requirements quality relate to project success or failure? In: International Requirements Engineering Conference, New Delhi, India, IEEE (2007) 69–78

[5] Neely, S., Stolt, S.: Continuous delivery? easy! just change everything (well, maybe it is not that easy). In: Agile Conference, Nashville, USA (2013) 121–128

[6] Oliveira, G., Marczak, S.: On the empirical evaluation of bdd scenarios quality: Preliminary findings of an empirical study. In: Workshop on Empirical Requirements Engineering in conjunction with the International Requirements Engineering Conference, Lisbon, Portugal, IEEE (2017)

[7] IIBA: A Guide to the Business Analysis Body of Knowledge (BABOK Guide) 3rd Edition. International Institute of Business Analysis (2015)

[8] Lucassen, G., Dalpiaz, F., VanDerWerf, J., Brinkkemper, S.: Forging high-quality user stories: Towards a discipline for agile requirements. In: Int'l Requirements Engineering Conference, Ottawa, Canada (2015) 126–135

[9] Corbin, J., Strauss, A.: Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. SAGE Publications, Inc (2004)