On the Empirical Evaluation of BDD Scenarios Quality: Findings of an Empirical Study

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

***Abstract*—BDD scenarios is a known format to represent acceptance tests in Agile methodologies. Those tests' goals are to communicate assumptions and expectations, by expressing the details that result from the conversations between customers and developers, and also validate that an user story has been developed with the functionality the team had in mind when they wrote it. We believe that this formalization of the behavior of a feature in the form of BDD scenarios need to be a good quality to avoid known requirement problems that arise from bad documentation. However, to the best of our knowledge, there are only informal guidelines to guide practitioners on their evaluation of the quality of their own scenarios. To address that lack of guidance, we used known quality criteria, already used to evaluate either traditional requirements or user stories, to support our inquiries on interviews with 18 practitioners about their personal evaluation criteria and their interpretations of those known quality attributes in their scenario's contexts. Those insights are used to construct new quality attributes, suited to evaluate BDD scenarios, and a reading technique in the form of a questionnaire, to guide quality evaluations.**

***Index Terms*—documentation quality, documentation evaluation, behavior-driven development, empirical study, reading technique**

I. Introduction

BDD... Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Requirements quality importance... Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Field Study summary, RQs... Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Results summary... Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Therefore, this paper proceeds as follows: Section 2 reviews the concepts of Requirements, Use cases and BDD and reflects upon the different set of criteria to validate requirements. Section 3 presents the study design we performed to acquire a deeper understanding of how quality attributes could be used to validate BDD scenarios and the preliminary findings during of this study. Section 4 and 5 concludes this paper by summarizing our perceptions about the study and outlining some directions for our on-going research, respectively.

II. Related Work

Requirements validation is a phase on traditional requirements engineering (RE) process that is known to support the three other activities (requirements elicitation, requirements analysis and requirements specification) by identifying and correcting errors in the requirements [1]. The Business Analyst Body of Knowledge (BABOK) [2] states that acceptable quality requirements exhibit many characteristics, as follows: atomic, complete, consistent, concise, feasible, unambiguous, testable, prioritized, and understandable. BABOK second edition [3] lists cohesion, completeness, consistency, correction, viability, adaptability, unambiguity, and testability as characteristics a requirement must have in order to be a quality one.

The BABOK 3rd edition [2] argues that one way to validate quality characteristics is through a review. Zhu [4] states that software review is primarily an individual effort and the kinds of reading techniques the individual uses are paramount to the outcome and effectiveness of software review. A software reading technique is a series of steps for the individual analysis of a textual software product to achieve the understanding needed for a particular task, a set of instructions that guide the reader how to read the artifacts, what areas to focus on and what problems to look for. The most widely used format of reading technique are checklists [4], a list of questions to provide reviewers with hints and recommendations for finding defects during the examination of software artifacts. One example of checklist is the one used by Cockburn [5] to evaluate the quality of Use Cases. Cockburn [5] says that a use case captures a contract between the stakeholders of a system about its behavior and describes the system's behavior under various conditions by interacting with one of the stakeholders (the *primary actor*, who wants to perform an action and achieve a certain goal). They are used to express solution requirements for software systems and can be put into service to stimulate discussion within a team about an upcoming system.

Conversations about requirements rather than formal documents and defined phases from traditional requirement engineering approaches is also seen in agile software development methods [1]. Most of those methods employ user stories, which describe functionality that will be valuable to either a user or purchaser of a system or software [6]. Each story must be written in the language of the business, not in technical jargon, so that the customer team can prioritize the stories for inclusion into iterations and releases. Lucassen et. al [7] summarize that user stories only capture the essential elements of a requirement: *who* it is for, *what* it expects from the system, and, optionally, *why* it is important. To complement user stories, acceptance tests are specified as soon as the iteration begins and, depending on the customer team's technical expertise, can be put into an automated testing tool. Those tests helps the customer team to communicate assumptions and expectations to the developers, by expressing the details that result from the conversations between customers and developers, and also validate that a story has been developed with the functionality the team had in mind when they wrote the story [6].

One way to format acceptance tests is using scenarios. Behavior-Driven Development (BDD) is a set of practices that uses scenarios as an ubiquitous language to describe and model a system [8]. They are expressed in a format known as Gherkin, designed to be both easily understandable for business stakeholders and easy to automate using dedicated tools such as JBehave and Cucumber. According to Smart [8], bringing business and technical parties together to talk about the same document helps to build the right software (the one that meets customer needs) and to build it right (without buggy code). The scenarios related to a particular feature are grouped into a single text file called a feature file. A feature file contains a short description of the feature, followed by a number of scenarios, or formalized examples of how a feature works. 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...* a context *When...* an action is performed *Then...* an outcome is obtained. BDD scenarios are similar to use cases scenarios as they both describe a system behavior under certain precondition (expressed on the *Given* clauses of a BDD scenario) to achieve a certain goal (expressed on the *Then* clauses of a BDD scenario).

However, to the best of our knowledge, BDD scenarios can only be evaluated based on characteristics taken from Smart's [8] experience, such as: scenarios steps expressiveness, focused on what goal the user want to accomplish and not on implementation details or on screen interactions (writing it in a declarative way and not on a imperative way); the use of preconditions on the past tense, to make it transparent that those are actions that have already occurred in order to begin that test; the reuse of information to avoid unnecessary repetition of words; and the scenarios independence. Even though Smart [8] specifies a few examples scenarios in order to demonstrate those characteristics he described, we feel it would be important to have a refined quality questionnaire built by the collective knowledge of other practitioners, similar to the one used on Cockburn's use cases [5].

For user stories, a format to represent agile requirements, the INVEST (Independent-Negotiable-Valuable-Estimable-Scalable-Testable) acronym presented by Cohn [6] seems to be one of the mostly used criteria in use by industry practitioners, as Heck and Zaidman [9] have stated on their practitioners interviews. Lucassen et. al [7] define additional criteria to evaluate user stories on their QUS Framework, as follows: atomic, minimal, well-formed, conflict-free, conceptually sound, problem-oriented, unambiguous, complete, explicit dependencies, full sentence, independent, scalable, uniform and unique. Following on his steps, we also find it important to discover attributes suited to evaluate BDD scenarios.

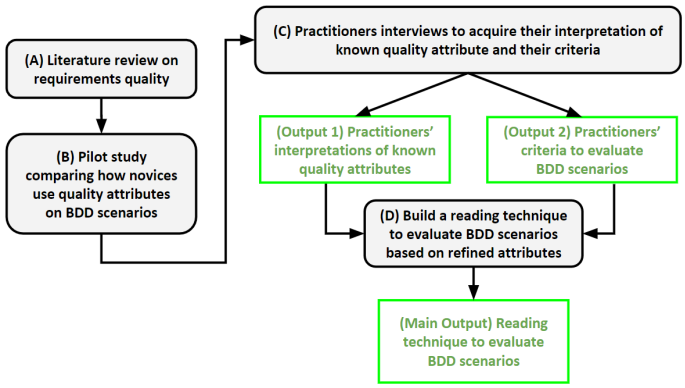

Fig. 1. Research Design

III. Research Context

The main goal of this research is to provide a set of criteria suited to use with BDD scenarios and the proper guidance to use those criteria effectively. To accomplish that, two research questions were posed to drive the research goal:

Research Question 1 (RQ1): What are the quality attributes suited to describe a "good" BDD scenario by the view of different members of software development team?

Research Question 2 (RQ2): How software development team member identifies those attributes on BDD scenarios?

A. Research Design

Since there is no standard on what is considered a "good" scenario due to the few views on this matter, this is a concept that still needs to be fully understood. A qualitative approach is suitable for situations like this, as mentioned by Creswell [10]. Also, we believe the opinions from industry practitioners working on multiple companies and contexts would be a natural input to construct a quality concept for BDD scenarios. Therefore, we judge that the use of an empirical approach would suit our objectives and research questions. Finally, we decided to run a field study for data collection using semi-structured interviews, as we value multiple different opinions taken from different contexts.

With those decisions made, we found ourselves wondering what kind of approach would best suit our needs: use open-questions interviews with practitioners and form an structured result from their knowledge and opinions; or create a certain set of criteria that we judged sufficient and let the interpretation of this sample model guide the interviews. We feared that the participants' plurality of terms and opinions would not provide enough data to formulate a reading technique without the guidance of some kind of theory, while the second could narrow the practitioners diverse experiences to our view of the subject being debated.

Therefore, we judged it necessary to have a mixed approach: our interviews would be guided by a sub-set of known quality attributes that we had previously seen evaluating BDD scenarios. First, we had to discover those known quality criteria. A literature review, represented in our research design 1 as Action A, has confirmed that traditional attributes [3][2] and the INVEST[6] acronym were used with agile requirements. To verify how effective those attributes could be when used with BDD scenarios, we performed Action B and organized a pilot study with 15 graduate students to understand how they evaluate BDD scenarios, already reported in our previous paper [11]. The output of this study was the list of attributes used in the practitioners' interviews, as follows: concise, estimable, feasible, negotiable, prioritized, small, testable, understandable, unambiguous, and valuable.

With the knowledge acquired on the pilot study, we were able to use a list of quality attributes that we knew could potentially work with BDD scenarios to guide the interviews and motivate participants to criticize them and suggest better alternatives. Additionally, we used known examples of BDD scenarios to aid practitioners realize their own quality criteria while reviewing those scenarios. 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.

This paper reports on Action C and forward – the practitioner's interviews and the reading technique in a form of a questionnaire. This action produced us a list of practitioner's interpretations of known quality attributes and their own personal criteria. With that data at hand, we are then able to refine our list of known quality attributes to make it suited to BDD scenarios and produce a questionnaire similar to the one Cockburn [5] uses for Use Cases (Action D).

B. Participants Selection

In order to acquire the first participants for our interviews, we organized direct searches on our own Linkedin social-network using the terms "BDD" and "Behavior-Drive-Development". Our belief was that a personal touch, showing our main researcher profile (that works with test automation and have experience with BDD) to potential participants, would improve our chances of finding good candidates. Within every practitioner profile, we double-checked the existence of BDD experience and proceeded to invite the person to connect and dedicate their time in an 1hr slot interview. By the time of the writing of this paper, 79 invitations were accepted and 34 invitations were not yet answered. From those who accepted to connect, we proceed to invite for an 1 hour interview.

In total, we interviewed 18 professionals. Each interview lasted in average 77 minutes, being the longest 1 hour and 51 minutes and the shortest 62 minutes long – a total of 1390 minutes were recorded. The interviews were conducted via Skype[2] tool and it was voice-recorded (with permission).

[1]https://diasporafoundation.org/
[2]https://www.skype.com

The interviews were performed either in English or Portuguese during a period of 3 months (from September to November in 2017) and then manually transcribed to the participant's native language. We realized we had reached saturation when no new practitioner quality criteria was found. That is when we decided to stop the interview process and start the compilation of the findings. Those participants profiles are summarized in Table I.

C. Interview Design

Taking inspiration on the questions used by Heck and Zaidman [9] on their interviews with practitioners to understand how User Stories quality can be understood, we list below the set of questions to capture practitioners interpretations of quality attributes and their own evaluation criteria. We assume the practitioner being interviewed works (or have worked in the near past) with BDD scenarios. The authors' original interview script was organized into two parts - one done with minimal introduction from their side and without showing the participants their framework and another in which they asked the participants to use their framework, transformed into a checklist, on some examples taken from open source projects. In the same way, we keep our list of quality attributes to ourselves until question 7 – we first ask the practitioner to list their own quality criteria and use them reviewing a Diaspora's feature file of their choice. Once this phase is done, our list of quality attributes is shown. With that list, we ask the practictioner's interpretation of them, to put them to use into another Diaspora's feature file (again chosen by the participant), to consider how suited they are for BDD scenarios, and to map them to their own criteria – making sure no criteria or attribute is forgotten. Our final interview script follows below:

1) What's your role on the project?
2) What's your main task on the project?
3) For how long do you use BDD?
4) How do your project use BDD scenarios?
5) What do you pay attention to when reviewing/writing BDD scenarios?
6) On Diaspora, evaluate a feature file according to your criteria.
7) Do you think any of the quality attributes could help when reading/writing BDD scenarios?
8) What is the meaning of each attribute on BDD scenarios?
9) On Diaspora, evaluate a feature file according to the attributes.
10) Do you miss any other attribute?
11) What quality attributes did you find difficult to use?
12) To what extent has the attributes helped you assess a scenario's quality?
13) How your criteria maps to those attributes?

The four starting questions were used to understand participant's profile and their answers were summarized in Table I. Questions 5 and 6 were useful to acquire participant's criteria without any bias of our list of known quality attributes and their answers were explored on section IV-B. If only a few answers emerged for those questions, we provoked their thoughts using a list of criteria taken from Smart's experiences [8] and from the answers of our pilot interviewee, as follows: steps too long/too short; too few/many steps; business language usage; title description; keywords (Given/When/Then) usage and order; repetition of steps. Questions 7 to 11 were useful to acquire participant's interpretations of our known quality attributes and were analyzed on section IV-A. Question 12 was useful to assess how useful such a list was and tease the interviewee to suggest better ways to evaluate BDD scenarios. Question 13 was a final check in case the conversation haven't linked participant's criteria and interpretations to attributes.

The interview script was validated by a doctoral student with 4 years of experience working with BDD, among the industry and the academy. Additionally, a pilot interview was conducted with a software tester that practices BDD for around 6 years – an interview that brought us a base list of criteria that could be used to provoke additional answers from later interviewees.

TABLE I
PARTICIPANTS' PROFILES

Participant	Role	BDD experience	Location
P1	Test	3 years	England
P2	Test	3 years	Netherlands
P3	Developer	3 years	Netherlands
P4	Coach	3 months	Denmark
P5	Test	9 months	Netherlands
P6	Test	2 months	Netherlands
P7	Test	3 years	Netherlands
P8	Dev/Coach	10 years	Hungary
P9	Coach	3 years	England
P10	Developer	6 years	Sweden
P11	Test	4 years	Australia
P12	Test	3 years	Brazil
P13	Developer	3 years	Brazil
P14	Test	1 year	Brazil
P15	Coach	5 years	Australia
P16	Developer	1 year	Brazil
P17	Test	4 years	United States
P18	Test	4 years	England

D. Data Analysis

Cruzes and Dyba state [12] that a number of different methods have been proposed for the synthesis of qualitative and mixed-methods findings such as the ones we typically find in software engineering (SE), which lead them to conceptualize thematic synthesis in SE. Thematic synthesis draws on the principles of thematic analysis to identify the recurring themes or issues in the primary sources of data, analyzes these themes, and draws conclusions from them.

Our interviews were conducted either in English or Portuguese and all the text was coded in English – a process made possible by the fact that both the main researcher, who interviewed the participants and coded the data, and the second researcher, who reviewed the codes, reads and speaks both languages.

We understand that both our list of quality attributes and our base criteria may change, so we decided to be open to that possibility and choose an integrated approach of analysis. Also, as we specifically decided to guide our interviews with quality attributes and with certain criteria inquiries that potentially work with BDD scenarios, with the goal to help interviewees put into words their feelings and thoughts about quality, we could not deny the effect it had into the interviewees – an effect that should appear on the analysis as well.

E. Threats to Validity

The fact that only the main researcher was involved into the coding process may have impacted the themes creation in an unforeseen way, even with the careful review of the second researcher. Additionally, we have not taken into consideration the gender, role, location nor the type of industry a participant works into to reflect upon the data taken from the interviews. We understand that different life experiences may have brought different quality criteria and opinions, so we tried to mitigate this effect interviewing people from many different areas and companies. Finally, we could have phrased the items in our questionnaire in many different ways, so we have to acknowledge that our own language bias may have driven us to write them in that way we presented. A specialist review o that questionnaire is listed as a future work to refine that questionnaire and avoid this threat.

IV. Results

This chapter explore the interviews participants responses (from P1 to P18) in order to clarify how their opinions answer our research questions. The multiple interpretations to traditional attributes used by the interviewees had force us to question if words such as "concise" and "understandable" would be suited to represent the characteristics of "good" BDD scenarios. Therefore, we decided to dealt with each interpretation of each attribute in separation. In the last subsection, we group those interpretations of and interviewees' personal criteria into 8 new attributes, more suited to evaluate BDD scenario's quality than known quality attributes, that serve as the answer for our RQ1 and provide us the insights needed to answer the RQ2 by building a proper questionnaire to evaluate BDD scenarios.

A. Quality attributes interpretation

Interviewees' interpretations of each known quality attribute that could potentially be used with BDD scenarioshelp us understand what words would help to evaluate BDD scenarios. To that end, the answers of the questions 8 and 9 were analyzed. Additionally, the answers of the question 10 helped us to evaluate if something was missing in our list, while the answers of the question 7 helped us to eliminate some attributes, judged as not useful for the participants.

Concise attribute was deemed important for all 18 interviewees. A concise BDD scenario has no unnecessary details (P3, P5, P8, P10, P11, P16, P17), is focused (P4, P7, P10, P12, P13, P14, P18), clear (P1, P3), and has only a few (P2, P7) brief and comprehensive (P3, P6, P9, P11, P18) steps. P5 reports that, a concise scenario *should be specific without giving way too much detail* and should have *no steps longer than they should*. P2 said that he *does not like overly long scenarios as the longer they are the most likely they contain implementation details*. For P14, a concise BDD scenario should be *straight to the point, without unnecessary details, one that does not stall, like those who are very long and want to explain many things, it does not need to be very long, trying to explain point-by-point, it can be more direct*. P1 preferred to use the word "clear" rather than concise, meaning that *one need to write clear steps and need to be sure that everybody has the same understanding so it's really clear and there's hard to read it differently*. P6 always question himself: *if I remove this line, will it actually change the implementation and the outcome? what can I remove without changing the meaning that would make it understandable?*

A small scenario, often considered the same as a concise scenario (P2, P7, P8, P9, P15), is one with just a few steps (P1, P3, P4, P10, P11, P13, P14), which test only one thing (P5, P6) and for which the code to turn it into an executable step is small (P16). P10 expands on his opinions by saying that *if [the scenario is] longer than 4 lines it's possible not small, if it's wider than the example I looked, some were very very wide, some I would break in some point. If I need a lot of words to express this specific example, it's certainly not small. And if one need lots of lines to do the same thing, it's probably not small*. Two participants, P5 and P6, reduces the word small to atomic – the fact that a test describes one thing. P6 describes that a certain scenario is not atomic because *it's testing two things, it's describing two things. These BDD scenarios are obviously written not to describe functionality or be testable – these are written as test scripts*. P16 declares that *a scenario cannot be neither small or big, It has to have enough steps, and well described and it's good already*", as he sees *small as in the implementation part of that step*. Additionally, some participants do not believe that scenarios have to be small (P12, P17, P18).

A testable scenario is one who allow the reader to follow its steps (P1, P12, P14, P18), has clear outcomes (P2, P4, P9, P15) and pre-conditions (P6), is independent (P16, P17), focused (P5), and cover all feature (P12), if we analyze all the scenarios together. For P1, be a testable scenarios means *it should be clear, something easy to understand so you can follow the steps*. For P2, testable means that *scenario's intended behavior, or what you are trying to verify, should be clearly expressed in the scenario*. P6 states that testable attribute is reflected on the Given steps, as *if your Given conditions are overly complex or very vague then stuff becomes hard to test, that's mostly because the setup for a test is overly complex. And if the setup is overly complex than it means your software architecture is overly complex*. P16 says that a testable scenario should be *as auto-sufficient as it can* in order to achieve independence of scenarios. For P5, *the Given/When/Then [structure] helps, to give you focus, to test one thing*. For P12, testable represents how complete the scenarios' coverage of the feature behavior

is, saying that "if it's not testable I can't even measure the test's coverage". Participants P3, P7, P8, P10, P11 and P13 says that testable is not related with scenario's, but with the product or system being tested.

For some participants, an understandable scenario uses an ubiquitous language (P2, P3, P5, P6, P8, P9, P10, P11, P15), is self-contained (P18), and is written in "good english" (P17). For others (P12, P16), understandable means only that technical people understand what to do. P11 defines understandable as *quite related to a term we use which is called ubiquitous. What that means is that you basically come up with a terminology which is ubiquitous. What that mean is that all your project team members they are aware of that terminology, and you do not use that terminology where is too technical or business focused alone. It's a terminology that both your technical and business people understand and it is specific to your domain, So, if you do that, all of the scenarios they will be understandable, they will be easy to understand, for everyone, irrespective of their background*. For P18, an understandable scenario *mean who someone, that has never seen it before, reads the scenario and knows 'ok, this is what a user is able to do', so it is self contained, includes all the information needed*. P17 interprets understandable as in the use of "good english", the use of *proper grammar, proper tenses, proper point of view*". When referring to tenses, P18's views are: *the Given steps should either be on past tense or present perfect tense, as it states things you already have; When steps should be active tense because it's something you are doing; Then steps should be present or future tense because that's an outcome that's expected*. For P12 and P16, understandable scenarios are the ones where technical people understand what to do. For P12, "I read and have no questions on what am I suppose to do. It should have a specific focus, it should take into consideration environment details and business details". The remaining participants (P1, P4, P7, P13, P14) associated understandable as part of other attributes.

Ambiguity on scenarios was interpreted as the use of vague statements, weak words and contradictions (P2, P3, P4, P5, P12, P14), and the lack of a single scenario's intention (P7, P11, P15, P17). Additionally, some concerns were mapped to this attribute, as the fact that scenarios with different descriptions test the same thing (P1, P3), the lack of enough test coverage (P6) and the steps' high granularity (P9, P16). One of the sources of ambiguity is bad use of language by using vague, not clear, statements such as "Then the outcome is ok" or "Then the result is good" (both suggested by P2) or "When we see the change in the GUI" (suggested by P4), opening too much room for interpretation (P5, P14). Also, P12 raised the concern of having one step contradicting the other. Another source of ambiguity is when the scenario intention is not clearly stated - quite often by the use of multiple When steps (P7, P11). P15 was also concerned with imperative steps, where the action was clear enough but the intention was not. Despite the scenario's description, P1 and P3 have reported that different scenarios may be doing the same thing, even if written differently (P3). For P6, scenario's coverage of the feature should also be mapped into unambiguous attribute, as well as the use of third person rather then first person statements – referring to "the admin user" rather than "I login as admin user". P13 declared that unambiguous maps to the ubiquitous term, already mentioned for some other participants as an interpretation for understandable. The participant P16 declared the use of steps written with a declarative language as another source of ambiguity due to the fact that it can hide a lot of implementation details.

A valuable scenario is unique (P6, P10, P11, P15, P18) by validating some different and interesting behavior. Some participants could not identify how valuable a scenario is by reading their scenario description alone – that characteristic would have been discussed with other team members during formal or informal conversations (P5, P8, P9, P17). Some participants preferred to say a scenario should have value for the business (P1, P12, P13, P14), to the technical team (P16), as a documentation source (P3), or as a communication tool (P2). For P6, *valuable makes sense because you have to think of a scenario - is this scenario going to add value to whatever we are doing?*. P11 agrees that uniqueness should belong to valuable, but that attribute should also cover the importance of a scenario for business people. For P8, valuable comes from discussions with the team, as they *only makes scenarios for things that are valuable*. Some other meanings of valuable appeared: for P2, those scenarios are a communication tool, so the communication and acting up on the misunderstandings is what's valuable on scenarios; for P3, *what value it brings to me is the documentation*. Those last two opinions were not tying how valuable a scenario is to their scenario description, and therefore could not be used to evaluate a scenario's quality.

Prioritized, Negotiable, Feasible and Estimable were largely regarded as not useful to evaluate BDD scenarios textual descriptions, as either they are useful only for conversations around scenarios, suited for feature file (sets of scenarios) only, dependent on steps' technical implementation knowledge or project's domain knowledge. P11, who declared those attributes as useful for conversations only, explains his thoughts about those attributes as follows: *So I think the other would already have been considered by the time you reach that stage, because, so if you are remember I was talking about example mapping, which is a step which comes before feature files. So, the way you write your examples, so when you write your examples you need to make sure they are estimable for example. And obviously your scenarios would be derived from examples and therefore will automatically be estimable. So it's more relevant at the time of example mapping rather than feature file*. Similar declarations were heard from all the participants who either have conversations before writing scenarios or validate them after they are written

Some characteristics outside our list were quoted as important as well, like completeness (P1, P4, P11, P18), coherence (P3, P8), cohesion (P12, P16), integrity (P17), and declarative rather than imperative language (P10, P11, P18).

B. Participant's criteria

In order to better understand how each attribute presence or absence is reflected on a BDD scenarios, participant's criteria, more tied to the actual textual scenario than generic attributes, were taken from the answers of the questions 5 and 6. Additionally, those criteria were tied back to the attributes with the question 13. To aid on the reporting of them, those criteria have been tied together into four groups: language criteria, steps criteria, title criteria and other criteria.

Language groups together characteristics that depends on writing decisions, such as the consistent use of business terms instead of technical ones, the declaration of actions rather than the description of steps and the subject point of view. Participants have mapped the consistent use of business terms, like *a glossary that appears consistently on all scenarios* (P9), as a good practice impacting understandable (P1, P2, P3, P5, P6, P7, P8, P9, P10, P11, P15, P18), unambiguous (P3, P8, P10, P13), and concise (P4, P11) attributes and as a pain-point for the estimable (P3) attribute – for P3, *if it's on technical language it's very easy to estimate*. In opposition, the use of technical language is considered as harmful for understandable (P1, P2, P5, P6, P8, P9, P11, P14, P17, P18), concise (P2, P10), unambiguous (P2), small (P10), valuable (P7), and testable (P10) attributes. Other examples of technical language would be the use of HTTP response codes (P5), or automation steps using elements on the user interface like "click a button".

Despite the consistent use of business language terms, some participants also pointed out the need to write steps using third person point of view. P18 highlights the use of third person as a good thing as *it could get your whole team to think like the user and that would be very useful*. The use of all forms of third person point of view is considered as a good practice and enhances the understandable (P2, P7, P17), concise (P9), unambiguous (P6), and valuable (P14, P18) attributes. Additionally, some participants (P16, P17), while inclined to use third person point of view on their steps, were already happy only with consistency - using always first person or always third person.

Despite the language terms and the point of view, there's also the concern about how granular a scenario step description should be. P2 says scenarios should *focus on the problem, not the solution* and P8 calls it as *declarative*, with *imperative* being his opposite. For P17, *Gherkin is meant to be declarative because it tries to describe behavior that add business value, it's not necessarily to give the implementation on how that behavior works*. For those participants who mix scenario's writing with conversations, either before or after the writing of them, the use of declarative language is seen as a good practice that enhances understandable (P7, P8, P9, P14, P15), testable (P1, P2, P5), concise (P8, P17), and unambiguous (P14, P15) attributes. Imperative language usage was considered as harmful for testable (P1, P18), small (P5), understandable (P7), concise (P17). However, P12 and P16, who uses scenarios with a technical approach in mind, considered declarative form of writing harmful for understandable (P12, P16), estimable (P16), feasible (P16), testable (P16), unambiguous (P16), and valuable (P16) attributes. Imperative writing was considered a good practice that enhances estimable (P12, P16), understandable (P12, P16) feasible (P16), testable (P13, P16), unambiguous (P16), and valuable (P16) attributes.

Steps characteristics, look at how many steps a scenario have, how long are they, and what step keyword to use. Having long scenarios with many steps is considered harmful mainly for concise (P2, P3, P6, P7, P8, P9, P15, P17, P18) and small (P1, P3, P4, P5, P10, P11, P13, P14) attributes, as already discussed. There were also reports about it affecting unambiguous (P3), understandable (P14), and testable (P18) attributes. In a similar way, lengthy statements are considered harmful for concise (P3, P5, P7, P9, P17), small (P1, P3, P4, P10), and understandable (P17) attributes. P17 says that, *the more imperative you write your scenarios the more lines it will have, so if you have too many lines, you can kinda guess, you can probably state some of those things a little bit better*, which could indicate that having fewer steps is not a proper goal, but having scenarios written in a declarative format rather than imperative would.

Additionally, there was a concern with the natural step order of Given/When/Then being violated. P2 said that alternating the use of When and Then steps *means that you need to split up in two smaller scenarios* and that *a good test is exactly one verification and alternating when and then means you are testing more than one thing in the same scenario*. Therefore, this violation of the natural step order is considered a bad practice that affects understandable (P2, P4, P8, P10, P14, P17, P18), concise (P2, P3, P8, P9, P14, P18), testable (P2, P7, P18), unambiguous (P3, P14, P17), valuable (P3, P18), and small (P6) attributes. In a similar way, the multiple repetition of the same step, demonstrated by the excessive use of Given/When/Then steps in sequence or the "And" keyword, is also judged a bad practice. P4 summarizes it saying that *If there are many Ands than it makes it probably harder to understand*. This repetition of the same step bad-practice affects unambiguous (P4, P7, P11, P12, P14), concise (P4, P7, P8, P14), testable (P4, P6, P9), understandable (P4, P7, P14), and small (P5) attributes.

Despite analyzing the language and the steps characteristics, participants were also asked to analyze the scenario's title and feature file's description. Feature titles and descriptions were often pointed out as *intended business outcome or the indented business value* (P2) in one way or another. That good practice would enhance the understandable (P4, P14) and valuable (P7) attributes. Regarding scenarios' titles, participants had declared it should either express the intended action (P3, P5, P9, P10, P13, P16, P17) or the intended outcome (P2, P8, P12, P14) or both (P18, P11). Surprisingly, only few have declared it affects an attribute – from those who did, they mapped it with understandable (P9, P18), testable (P2, P15), and concise (P12) attributes.

Despite the scenario's language, titles and steps descriptions, Gherkin language allow other types of constructions

such as the use of a background section, tags, scenario outlines examples, data tables or double quotes aid to pass parameters. P15 reflects on the need of a background section, when asking himself: *does adding a Background make it harder for people to read? If it makes it harder for people to read, then 'no', it's a bad idea. If you've got a quite large number of scenarios, the Background should have a different title as well*. The use of background section can enhance the concise (P2, P4, P17, P18) and understandable (P2, P4, P17) attributes.

Tags were reported as being used for scenario's categorization (P1, P2, P3, P11, P12, P13, P14, P16) and as a communication tool (P3, P7, P10, P15, P17). The use of tags affects valuable (P1, P3, P4, P14, P17), understandable (P3, P11), prioritized (P3, P14), and unambiguous (P3) attributes. However, P2 says it can harm understandable due to their technical nature.

Data tables had caused different reactions from participants. As summarized by P8, *tables has pros and cons, such as, well, the pros are for repetitive tests that needs to cover a different range in the input and they could be considered one scenarios. The cons are that you might accidentally bunch many scenarios together that should be kept separate - so it should be less and less easy to read, less clear*. The use of data tables affects concise (P1, P3, P11, P17, P18), understandable (P3, P7, P17, P18), unambiguous (P14), and testable (P14) attributes.

In a similar way as tables, scenario outlines have received mixed reactions from participants. P5 uses it in a pragmatic way, as stated: *so we used these scenarios outlines with examples to check the different type of cases. It's a really easy way to get all of this, to test all of these cases, to check if all the different options are covered and are correct. But I also understand how, you know, the fact that they are so easy to use and it's so easy to add more test cases is also a risk because then you end up with huge test suites and you have to be critical of does each example actually add value*. The use of scenario outlines affects concise (P17, P18), understandable (P17, P18), unambiguous (P1), and testable (P17) attributes.

V. Discussion

In order to analyze our results, we first have to understand what we have: in one hand, we have multiple interpretations to traditional attributes; at the other, we have interviewees' personal criteria, attached to BDD scenarios details.

Each of those interpretations of each attribute can be seen as a potential attribute, as we cannot decide for ourselves what interpretation of each attribute is best suited than others without enforcing our own bias. In order to couple those interpretations to BDD scenarios, we mix them with interviewees' personal criteria, that are naturally attached to BDD scenarios, and the known quality attributes are a natural bridge to allow that mix. From now on those interpretations and criteria as characteristics, and map them to new set of quality attributes as stated in Table II. As we now deeply understand those attributes and how they were created, we can phrase them as questions and build a questionnaire to evaluate BDD scenarios properlly. Those new attributes answer our RQ1, while the questionnaire answers our RQ2.

Some of the participants opinions had to be left aside. Due to our focus on textual scenario's descriptions only, the interpretation of small as in a "small code for that step" did not hold a position in our groups. In the same way, the missing characteristic of independent scenarios, that would allow the evaluation of how a scenario execution could affect the other, was left out of our list. Neither are other automation characteristics described on prior sections.

Additionally, the interpretation that an understandable scenario is one where "technical people understand what to do" conflicts with our notion about how BDD scenarios should bring together business and technical roles and serve as a "standard documentation source" (P18) – therefore, our groups do not cover that as well.

TABLE II
Characteristics mapping to new attributes

Characteristic	**Attribute**
Concise – Not much details	Essential
Concise – Not many steps	Essential
Concise – Focused	Focused
Concise – Clear	Clear
Concise – Not Unnecessary Lines	Essential
Small – Not Many Steps	Essential
Small – Test One Thing	Singular
Small – Code Step Small	N/A
Testable – Clear Outcomes and Verifications	Singular
Testable – Follow the Steps	Complete
Testable – Clear and Simple Givens	Singular
Testable – Focused	Focused
Testable – Independent	N/A
Testable – Completeness	Complete
Understandable – Ubiquitous	Ubiquitous
Understandable – Self Contained	Complete
Understandable – Good English	Integrous
Understandable – Technical People Understand What to Do	N/A
Unambiguous – Vague Statements	Clear
Unambiguous – Single Clear Intention	Singular
Unambiguous – Scenarios Testing the Same	Singular
Unambiguous – Completeness	Complete
Unambiguous – Ubiquitous	Ubiquitous
Unambiguous – High Granularity	Clear
Valuable – Unique	Unique
Valuable – Business Value	Unique
Additional Attributes – Complete	Complete
Additional Attributes – Writing Patterns	Integrous
Additional Attributes – Cohesion	Unique
Additional Attributes – Integrity	Integrous
Additional Attributes – Declarative vs Imperative	Focused
Language – Ubiquitous	Ubiquitous
Language – Technical Jargon	Clear
Language – Actor Consistency	Ubiquitous
Language – Declarative rather than Imperative	Focused
Steps – Few and Short Steps	Essential
Steps – Steps Order	Integrous
Steps – Steps Repetition	Essential
Titles – Feature Description	Unique
Titles – Scenario Titles	Unique
Additional Constructions – Background	Essential
Additional Constructions – Tags	Unique
Additional Constructions – Data Tables	Essential
Additional Constructions – Scenario Outline	Essential

The essential attribute came from some of the interpretations of concise, such as the avoidance of unnecessary details and lines. The use of unnecessary lines would make more steps to be written, and therefore the "not many steps" interpretation, that was assigned to concise and small attributes, also falls into this new attribute. Therefore, only essential information should be written into textual BDD scenarios. The steps' criteria "unnecessary lines" and "many steps" reinforces that belief – both have affected either concise or small attributes badly. One practice to avoid pre-conditions repetition is the use of a background section – a criteria that aid on concise attribute and therefore would also aid to the new essential attribute. Another good practices are to either summarize steps into a data table, short and with meaningful titles, or to group similar scenarios together using a scenario outline. Both practices have benefits and potential problems, as already mentioned on the corresponding subsections, but they seem to positively affect the essential attribute idea as all of them had positively affected concise attribute according to some participants.

Declaring "what" a scenario should do, rather than describing "how" that action will be performed with a sequence of commands, was one of the interpretations of concise and testable attributes. Therefore, focused seems to be a word worthy of a new attribute. It can also be understood by looking from the "Declarative rather than Imperative" language criteria perspective, that some participants (P1, P5, P8, P16, P17) declared to affect either concise or testable. Strangely enough, that criteria had also affected understandable attribute in a similar degree, but no participant had said that understandable BDD scenarios should be more focused. Additionally, "Declarative rather than Imperative" also appeared as a missing quality attribute in the list used in the interviews, reinforcing the idea of having this characteristic as an exclusive attribute.

The fact that BDD scenarios should have a "single intention" was an interpretation mentioned for unambiguous attribute that is similar to the "test one thing" interpretation of the small attribute. Another interpretation of the unambiguous attribute was also that, sometimes, scenarios written differently were "testing the same thing". Therefore, we believe that the singular intention should be an attribute evaluated in separation of the others. Clear outcomes (assigned to verifications on Then steps) and pre-conditions (assigned to Given steps), both mentioned on testable attribute, would also be interpretations tied to this attribute.

Clear attribute appeared due to the fact that vague statements, an interpretation from unambiguous attribute, can harm as much as an excess of details. Additionally, high granularity steps, another interpretation of unambiguous attribute, could be easily identified as vague steps for technical people. However, the use of technical jargon, a criteria that harms understandable attribute, could represent less clear steps for business people. Therefore, there should be a certain balance that allows a scenario to be correctly understood by all parties involved. The word clear appeared as an interpretation of the concise attribute that would also reinforce the before mentioned unambiguous interpretations.

Completeness, an interpretation from testable and unambiguous attribute, can be seen from multiple perspectives. On the scenario level, all the information needed to understand and follow those steps should be present – represented by the "follow the steps" testable attribute interpretation and the "self contained" understandable attribute interpretation. On the feature level, the set of scenario's should provide enough coverage for that feature – a missing attribute on the interviews list mentioned in the early subsections.

A scenario's business value should be evident from its description and from the fact it is interesting – represented by the question "is it testing something fundamentally different to the other scenarios" from P15. That interpretation came directly from the valuable attribute. Even tough we could keep using valuable word to describe this characteristic, uniqueness seems to be more suited to represent it. To aid on that assessment, each scenario title should inform the reader about its behavior, preferably expressing its action and its outcome while correlating those with the actual steps. Also, feature file descriptions, often in the user story format, can aid on recognizing a set of scenarios importance. Tags, serving as the scenario's meta-data, can provide important information to express the scenario context.

The ubiquitous attribute, born from interpretations of understandable and unambiguous attribute, represent the use of business terms, either taken from a glossary or a team's common knowledge, and helps to reinforce the need to bring scenarios closer to technical and business people alike. Additionally, the criteria of using business defined roles and/or personas in a consistent way enhances the ubiquity of a scenario description, as some participants (P2, P6, P7, P17) argue that it positively affect either understandable or unambiguous attributes.

A scenario should remain integrous, and for that it should respect the rules of Gherkin language, as the natural sequence of Given/When/Then steps. Also, it should use proper steps tenses (Given steps in the past, When steps in the present, Then steps in the future), and it should respect each keyword type (Given describing pre-conditions, When describing actions and Then describing verifications) as described on both the understandable and the missing integrity attribute. Finally, as pointed out on another missing attribute on the interviews list, it's interesting to keep a certain writing pattern, to avoid having different styles of descriptions appearing on different scenarios written by different people.

Taking those attributes into consideration, the following questionnaire was assembled to guide practictioners on their evaluation of BDD scenarios. Each question is tied to one or more attributes, that can be mapped back to practictioner's interpretations and criteria using Table II.

- **[Unique]** Can you identify the business value or the business outcome by reading the Feature description?
- **[Complete]** Do you fell any scenario is missing without a reason?
- **[Singular]** Is the scenario describing only one action?
- **[Clear and Complete]** Do you fell any step is missing without a reason? This scenario carries all the information

needed to understand it?

- **[Essential]** Could any steps be removed without affecting the scenario understanding?
- **[Unique]** How different this scenario is from the others?
- **[Unique and Singular]** Can you identify what single action the scenario is trying to perform by reading its title? Does that match what the scenario is actually doing?
- **[Unique and Singular]** Can you identify what the scenario is trying to verify by reading its title? Does that match what the scenario is actually verifying?
- **[Integrous]** Keywords meaning (Givens are preconditions, Whens are actions and Thens are verifications) and its natural order (Given/When/Then) are respected?
- **[Ubiquitous]** Does that step uses any business terms defined in a glossary? What actor is performing this scenario?
- **[Essential]** Could any details be removed from that step description without affecting the scenario understanding? Could those necessary details be summarized in a data table?
- **[Focused]** Can you identify "What" the step is doing ? Does that step is written using a declarative way?
- **[Clear]** Can that step be interpreted in different ways ? Is is vague or misleading?

VI. Conclusion and Future Work

Questionnaire validation... Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

References

[1] V. T. Heikkila, D. Damian, C. Lassenius, and M. Paasivaara, "A mapping study on requirements engineering in agile software development," in *Euromicro Conference on Software Engineering and Advanced Applications*, 2015.

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

[3] IIBA, *A Guide to the Business Analysis Body of Knowledge (BABOK Guide) 2nd Edition*. International Institute of Business Analysis, 2009.

[4] Y.-M. Zhu, *Software Reading Techniques: Twenty Techniques for More Effective Software Review and Inspection*. Apress, 2016.

[5] A. Cockburn, *Writing Effective Use Cases*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1st ed., 2000.

[6] M. Cohn, *User Stories Applied: For Agile Software Development*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004.

[7] G. Lucassen, F. Dalpiaz, J. VanDerWerf, and S. Brinkkemper, "Forging high-quality user stories: Towards a discipline for agile requirements," pp. 126–135, 2015.

[8] J. Smart, *BDD in Action: Behavior-Driven Development for the Whole Software Lifecycle*. Shelter Island, NY: Manning Publications, 2014.

[9] P. Heck and A. Zaidman, "A quality framework for agile requirements: A practitioner's perspective," *The Computing Research Repository*, 2014.

[10] J. W. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. Sage Publications Ltd., 3 ed., 2008.

[11] G. Oliveira and S. Marczak, "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.

[12] D. S. Cruzes and T. Dyba, "Recommended steps for thematic synthesis in software engineering," in *International Symposium on Empirical Software Engineering and Measurement*, ESEM '11, (Washington, DC, USA), pp. 275–284, IEEE Computer Society, 2011.