Initial Investigations on the Influence of Requirement Smells on Test-Case Design

Armin Beer
Beer Test Consulting
Baden, Austria
armin.beer@bva.at

Maximilian Junker, Henning Femmer
Technical University of Munich
Munich, Germany
{junker,femmer}@in.tum.de

Michael Felderer
University of Innsbruck
Innsbruck, Austria
michael.felderer@uibk.ac.at

***Abstract*—**Requirements-based testing has become a critical quality assurance technique designed to ensure a sufficiently high degree of product quality. However, the quality of the test cases depends on the quality of the requirements specification. In a preliminary experiment, we analyze potential links between the quality of the requirements and of the test cases.

Keywords— System testing; requirements-based testing; requirement smells; requirements quality; traceability; software quality; empirical study.

I. Introduction

Requirements-based testing has been recognized as the key to aligning business value and risks in industry. For instance, in the banking and insurance domain, according to the experience of the industry by the first author, up to 70% of the development effort is invested in testing.

System and integration tests are performed by people who are typically skilled in the design and execution of functional tests and have good domain knowledge [1]. However, sometimes too little attention is paid to a detailed review of the requirements specification with respect to its quality and domain-specific content. As a consequence, the quality of the requirements and smells in the requirements specification may impact the quality of the derived test cases.

Therefore, the objective of this paper is to investigate whether (system) testers design functional tests effectively in response to low-quality requirements. For this purpose, students of a software testing course had to solve two tasks: the creation of functional test cases on the basis of a Software Requirement Specification (SRS) with requirement smells [3] during a lecture, and a correct SRS as homework. The setting was similar to the student experiment published in [1].

Our work is related to empirical studies in the field of testing within the framework of the development of web services and designed to introduce systematic test design techniques into a system test team [2].

However, the influence of the quality of the SRS needed to create efficient test cases has rarely been subject to scientific analysis. In this paper, we analyze this problem in the context of a software testing course at the University of Innsbruck.

The structure of the remainder of this paper will now be described. Section II provides the background to requirements-based testing and requirement smells. Section III describes the research method. Section IV presents the results of the work of the students. Finally, Section V concludes this paper and presents future work.

II. Background

In this section, we briefly present the topics of requirements-based testing and requirements checks with the automatic ConQAT tool.

A. Requirements-based testing

Requirements-based testing is an approach in which test cases are designed on the basis of test objectives and test conditions derived from the requirements, e.g. tests that perform specific functions or probe non-functional attributes such as reliability or usability. The standard process has four phases: test planning, design, execution and evaluation. Test design starts by defining abstract test cases on the basis of the use-case description and test-design techniques. The tester then creates physical test cases with preconditions, test steps and test data.

Anti-patterns: We provide selected anti-patterns (various test-case design errors such as missing verification steps, unspecified test data etc.) in the first place to connect the erroneous use-case description to test-case design. Anti-patterns also indicate the misuse of test-case design techniques.

B. Requirements smells and ConQAT

A requirement smell is an indicator of a quality problem in a requirements document. A smell in a requirements document is always connected to a specific location in the document and a detection mechanism exists for each type of smell [3].

Below are examples of requirement smells (the occurrence of the smell in the sentence is highlighted in *italics)*.

- Passive voice without naming the agent (e.g. “When the contract *is closed*, the notification *is sent*.”)
- Imprecise phrases (e.g. “*Optionally*, the system *should* send an e-mail.”)
- Vague pronouns (e.g. “When, after sending the message, the system receives a warning signal, it should retry *it*.”)

Requirement smells may impair the readability of a requirements document and can lead to misunderstandings between the authors and readers of a document.

ConQAT is an open-source toolkit for performing quality analyses of software artifacts, such as code, as well as requirements documents. For this study, we used existing requirement smell analyses based on ConQAT [3]. This tool support enables us to check the requirements specifications automatically for the occurrence of requirement smells.

III. Research Method

In order to investigate the role of the SRS quality in the test-case design process, students of a course in software testing were assigned the task of designing test cases during a lecture and at home. These tasks were carried out at the University of Innsbruck (Austria) in November and December 2016.

A. Research Goal and Questions

The goal was to investigate how testers who are not experts in requirements engineering can bridge the gap between the interpretation of an SRS and test-case design. We derive the following research questions that we address in the empirical investigations performed in cooperation with students:

(RQ1) Does the quality of the SRS influence the quality of test cases?

(RQ2) Which requirement smells correlate with anti-patterns in the test cases?

Depending on the software testing techniques used, RQ1 addresses the quality of the test cases and RQ2 the quality of the alignment between the requirement smells and these cases.

B. Study Design

The object of the study is a course-participation management system. The application consists of 37 requirements and is web-based: it allows courses to be edited, supports searches, and retrieves functionalities and printing masks. As the students are familiar with the procedures covered by the application, the participants can be considered domain experts. Just as bank employees are familiar with banking software which they use regularly, a participation management system in a university course is regularly used by all students and is therefore within their domain of expertise.

The subjects of the study were 26 German-speaking BSc. students in computer science at the University of Innsbruck taking a software engineering course. All students have basic knowledge of software engineering and were trained in software testing. In addition, we consider them to be domain experts as they are familiar with the domain of the system under test, i.e. the course participation management system of the University of Innsbruck. The participants were asked to develop functional test cases for the two use cases “search” and “edit courses”. The requirements-based testing process was performed with ConQAT

The procedure of the study is shown in Figure 1.

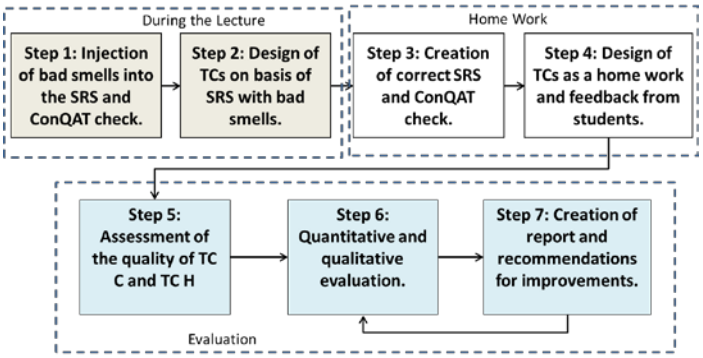

Figure 1: Design of the study

The first phase of the study (steps 1 and 2) is performed during the lecture. Steps 3 and 4 are part of the homework. The results are evaluated in steps 5, 6 and 7.

Step 1: We injected requirement smells into the use-case description and business rules of the SRS. For example, we removed the agent from some use-case steps, added imprecise phrases (i.e. optional, maximum), combined several use-case steps into one, etc. Table I presents the findings of the ConQAT tool [3] for the use-case, business rule 6 and the requirements.

TABLE I: Result of ConQAT check

Result of ConQAT-check		
Smell ID	**Requirement smells**	**Number**
S1	Passive voice without agent	13
S2	Loophole	4
S3	Negative words	6
S4	UI details smells	3
S5	Imprecise phrase smells	8
S6	Vague pronouns smells	5

Step 2: Prior to the first exercise during the lecture, we introduced the course participation management system to the students. For this purpose, we explained typical use cases to them. After discussing the specific requirements, we asked the students to apply test design techniques in order to devise test cases for the two pre-selected use cases. All students carried out this task in the class and had 60 minutes to create 5-10 test cases. We informed the students about the existence of anomalies in the SRS but not about their specific location or type. The students used a template to define the test cases, including the name of the test case, the test goal, the pre-conditions, a number of test steps with corresponding input data and the expected results.

After the design of test cases, we asked each student to answer a questionnaire to check the perceived difficulty of the task.

Step 3: The lectors created an SRS without requirements smells and checked it with ConQAT. The content of the SRS with the requirement smells was not altered.

Step 4: Time constraints obliged the second part of the study to be performed as homework. The students had to fulfill the same task, but now using the correct SRS with no

"bad smells" at all. They were asked to describe the anomalies in the SRS detected during the design of test cases in the lecture. The results were submitted to the lector.

Step 5: The quality of one test case covered during the lecture and one from the homework of each student was assessed and documented by a lector (see also Figure 2).

Step 6: The metrics described in the next chapter had been created during the preparation phase. They are used to recognize and compare the quality of the TCs of the lecture and the homework.

Step 7: We documented the lessons learned and presented them to the students.

IV. Results

In this section, the measurement results for each research question are presented and discussed.

A. Measurement Results

For RQ1, the dependent variables measure the *quality of test cases*, i.e. whether an erroneous or imprecise SRS has an impact on the design of these cases. We evaluated three quality attributes in terms of the five-point Likert scale (from 1 (very good) to 5 (very bad), taking a reference test case into account.

- *Definition of the test goal*, i.e. the defined test objectives are understandable, unique and reasonable.
- *Correctness of the test case*, i.e. the test cases were created correctly, taking the SRS and test techniques into account.
- *Completeness of the test case*, i.e. the correct number of test steps according to a sample solution is defined.

The following two criteria shown in Table II are related with respect to RQ2:

- *Categories of requirement smells*, i.e. the requirement smells detected by ConQAT.
- *Anti-pattern categories*, i.e. the types of errors in the test cases detected after a review.

The anti-patterns we observed in the test case steps are assigned to the test steps. The anti-patterns which were introduced because of requirement smells are linked to the categories detected by ConQAT (see Figure 2).

In Table II, requirement smells are assigned to anti-patterns taking potential errors of test case design into account. For instance, if the smell *S1-passive voice* occurs, a login step with an actor could not be implemented.

In Table III, an example of the evaluation of one student's submission is presented. A comparison of the results in the example indicates the negative influence of the requirement smells on the quality of the test case C. The lector linked the requirement smells S1, S3 and S5 to anti-patterns P1, P2, P3 and P7 with the aim of finding the reason for the defect in the TC.

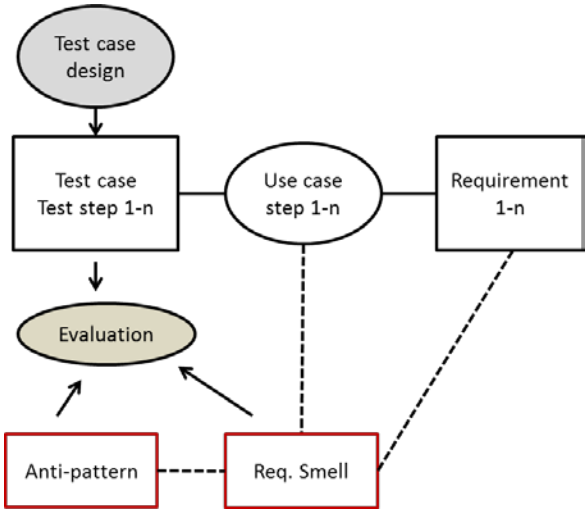

Figure 2: Design of test cases and evaluation of results

Table II: Relation of Bad Smells to Anti-Patterns

ConQAT requirement smells	Test-case design -Anti-patterns
S1 - Passive voice without agent	P1 - Anti-pattern 1: Missing test case steps: login depending on the role of actor and expected result are not implemented.
	P3 - Anti-pattern 3: Verification step is missing: Successful creation of entry is not tested.
S2 - Loophole smell	P5 - Anti-pattern 5: test data are not specified.
S3 - Negative words	P7 - Anti-Patten 7: Selection of list item cannot be implemented.
S4 - UI details smells	P4 - Anti-pattern 4: Inprecise test step - Links to requirements and GUI-details are not recognized correctly.
S5 - Inprecise phrase smells	P2 - Anti-pattern 2: Test case design method boundary value analysis is not applied.
S6 - Vague pronouns smells	P5 - Anti-pattern 5: Test data are not specified.
X - Inprecise phrase smells - not detected by ConQAT	P6 - Anti-pattern 6: Negative test cases are not implemented correctly.

Table III: Example of the Evaluation of Quality of Test Case C – Class and Test Case H – (Homework)

Criteria	TC C	TC H
Student 1		
Number of test steps	8	8
Definition of test goal	2	1
Correctness of test case	3	1
Completeness of test case	2	1
Req. smells	S1,S3,S5	
Pattern	P 1,2,3, 4, 7	Correct

Table IV and summarizes the results of the test cases (TC) submitted by 26 students.

TABLE IV: COMPARISON OF TEST CASES AT C AND H

Criteria	TC C Median	TC H Median
Number of test steps	7	8
Definition of test goal	2	1
Correctness of test case	3	1
Completeness of test case	2	2

TABLE V presents the number of anti-patterns detected in the evaluated test cases (TC) C and test case H.

TABLE V: ANTI-PATTERNS DETECTED IN TEST CASES (TC)

Anti-Pattern	Number TC-C	Number TC-H
P 1	27	10
P 2	17	7
P 3	22	12
P 4	17	1
P 5	2	0
P 6	0	0
P 7	15	0
Correct	0	11
Sum	**100**	**41**

Table VI shows some of the comments of the students regarding the interpretation of the SRS and the creation of test cases.

TABLE VI: PROBLEMS EXPERIENCED BY THE STUDENTS AS RECORDED IN A QUESTIONNAIRE

Interpretation of SRS	**Creation of test cases**
Some use cases did not have the right level of detail.	To derive the appropriate number of test-case steps from use-case steps.
Inconsistencies of assignment of use-case steps to masks.	To link the use-case steps with user interaction (GUI: Icons, Button etc.)
The interpretation of use-case steps was difficult.	To understand the domain-specific aspects of the system ("professionalism")
Without a graphical representation, it was difficult to understand the flow of information.	Transformation of use-case steps into test-case steps.

B. Discussion

The study yields initial indications that investments in the analysis and review of the requirements lead to better results in test-case design. Testers usually have knowledge of the application domain and in applying systematic test-design techniques. Given the pressure of time in a typical test project, it is a worthwhile question as to whether testers should also have a good knowledge of requirements engineering aspects in order to promote the improvement of the quality of an SRS. Incomplete requirements lead to conflicting interpretations. Testers often fill the resulting gaps in the SRS with their own ideas, while developers might have a completely different view. In our study, the quality of the test cases improved when we used an SRS without requirement smells, although this varied depending on the participants' skills in interpreting RE-artifacts and in test-case design. The number of anti-patterns dropped from 100 detected in the test cases created on the basis of the SRS with smells to 41 without requirement smells. As shown in Table V, the median of the quality attributes "definition of the test goal" and "correctness" was significantly better when using the SRS without "bad smells".

The validity of this conclusion concerns data collection and the reliability of the measurements, any of which might affect the ability to draw the right conclusion. The results of the study, i.e. the spreadsheets with the test cases as well as the questionnaire, were directly sent to two authors. The data was also reviewed and analyzed by two authors. Another concern is the learning effect of the students. However, by mapping the requirement smells to anti-patterns taking potential errors of test case design into account, still a conclusion about the relationship between requirements smells and the quality of the test cases can be drawn.

To sum up, our study indicates that applying a ConQAT requirements-smell analysis and correcting the requirements before or during test-case design results in:

- a better alignment between the requirements and functional test cases, and
- a higher number of correct test cases.

In general, extending the scope of the testers to identify anomalies in the SRS thus appears to be a viable procedure to narrow the gap between software engineering teams and testers and to motivate improvement measures in software development.

V. CONCLUSION AND FUTURE WORK

To close the gap between requirements engineers and test engineers, it might be worthwhile to extend the scope of the testers into the requirements engineering field. In this respect, the quality of the test cases obtained is a critical factor. The study shows that the early detection of "requirement smells" influences the quality of the test cases and promotes the comprehensibility of the requirements.

REFERENCES

[1] A. Beer, R. Ramler, "The role of experience in software testing practice", Proceedings of the 34th Euromicro Conf. Softw. Eng. and Advanced Applications: 258-265, IEEE 2008.

[2] M. Felderer, A. Beer, B.Peischl, "On the role of defect taxonomy types for testing requirements: Results of a Controlled Experiment", Proceedings of the 40th Euromicro Conf. Softw. Eng. and Advanced Applications, pp. 377-384, IEEE 2014.

[3] H. Femmer et al., "Rapid requirements checks with requirement smells", Journal of Systems and Software, Elsevier 2017.