Creating User Scenarios through User Interaction Diagrams by Non-Technical Customers

Douglas Hiura Longo and Patrícia Vilain
Informatics and Statistics Department,
Federal University of Santa Catarina,
Florianopolis, Brazil
douglashiura@inf.ufsc.br, patricia.vilain@ufsc.br

Abstract—This paper investigates the applicability of User Interaction Diagrams (UIDs) as user scenarios for specifying requirements of software built by non-technical customers. User scenarios represent an alternative to representation of Acceptance Test-Driven Development (ATDD). Two methods for building user scenarios using UIDs were proposed: the progressive and the regressive methods. The progressive method for construction of scenarios provides a description from any starting point until the expected result is reached. The regressive method is based on the Assert-First technique, introduced in Test-Driven Development (TDD), where the user scenario is constructed the other way round, that is, from the expected result to the starting point. These two methods were applied in an experiment where the results demonstrated that the regressive method requires significantly less effort as compared to the progressive method. The quality criteria of the two methods were different, where the regressive method yielded better results.

Keywords: Requirements Engineering; ATDD; Assert First; TDD; User Scenarios.

I. INTRODUCTION

It is known that understanding user requirements is critical to the success of a project [1]. The basic idea of Automated Acceptance Testing - AAT is to document requirements and desired outcomes in a format that can be automatically and repeatedly tested [2]. AAT represents customer expectations [1] and was adopted in agile software development holding great promise of improving communication and collaboration among those involved [2, 3, 4].

According to Hoffmann et al., Acceptance Test-Driven Development - ATDD facilitates the requirements specification, raising awareness, to those involved, of the importance of testing as an auxiliary mechanism for quality assurance. In theory, the customer expresses requirements as input to the software paired with some expected result [5]. However, in practice, customers prefer to express requirements at interaction meetings, while acceptance tests are written by developers [2].

Alvestad investigated whether a non-technical customer could express requirements based on domain specific languages, but was not successful in confirming his assumptions [6]. Domain specific languages are used by tools for automated tests. The tools for AAT support these requirements representations: formal, semi-formal and informal (NL, Stories, Tables). However, such tools do not include the end user in the requirements specification process [7]. According to Haugset and Hanssen, the application of AATs is barely reflected in practice and it is somewhat inappropriate for customers to express requirements in the form of automated acceptance tests [2].

The tools or languages in general induce the sequential creation of a requirement specification, which is where the requirement is specified, starting from any point towards the desired outcome. This is called the progressive method.

However, Test-Driven Development - TDD is a set of development practices, where the code is developed from tests. The Assert-First technique has a powerful simplifying effect during test development [8]. This technique consists of writing the test assertions first by following a regressive process in order to complete the test, writing the minimum of code lines. In this way, the regressive method consists of creating the requirement specification from the result.

User Interaction Diagrams (UIDs) represent the interaction between the user and the system, and can support the users' representation of scenarios [9, 10]. The present study investigates the possibility of a non-technical customer to express the system requirements by using UIDs as user scenarios. However, it also investigates the creation of requirements specification using the progressive and regressive methods.

To evaluate the proposal, we considered an experiment with 21 non-technical participants, and the requirements specification for a game. The objective of the experiment is to demonstrate the use of UIDs as an agile method for requirements specification, and check different results between the two methods.

This paper is organized as follows: Section II presents details of the proposal. Section III describes the evaluation methodology for verifying the proposal's efficacy. The results of the study are presented in Section IV. Section V presents threats to validity. Finally, the conclusions are presented.

II. RESEARCH PROPOSAL

This study uses UIDs for representing software requirements. This proposal replaces the information types

DOI reference number: 10.18293/SEKE2015-179

represented in the UIDs by values of the user scenarios. Fig. 2 shows an example of user interaction with a calculator and the representation of a user scenario through a UID.

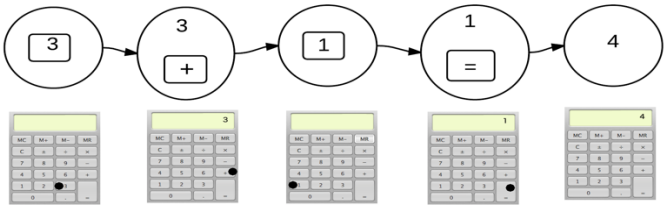
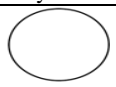


Figure 1. User scenario of a calculator for the sum function, with sample pictures of the user's interaction (points) with the calculator.

Fig. 1 shows the user's interaction with the calculator, following the user scenario. In this example, the user enters the values for the sum (3 + 1 =) and the system displays the result (4).

Table 1 below presents the symbols for the language to represent user scenarios through UIDs.

TABLE I. SYMBOLS FOR THE LANGUAGE OF USER SCENARIOS REPRESENTATION THROUGH UIDS.

Symbol	Use
	**Ellipse** - represents a state of interaction
	Arrowed line - represents the transition between interaction states and flow direction.
	**Rectangle** – represents the user input, its value is represented by a set of characters contained within the ellipse.
Characters sequence	**Value** - represents the system output, where a set of characters is contained within the ellipse.

Every interaction state (ellipse) contains the values of the user input and system output. The flow of the interaction states is represented by the direction of the transition. The initial state is the first interaction state following the direction of the arrows. The final state, in turn, is the last interaction state of the flow. For the construction of the requirements specification as user scenarios, the following methods are presented:

- Progressive: it indicates the expected result only at the end of the construction flow. Initially, the interaction states are built in order to achieve the expected result. Fig. 1 shows the progressive flow of the construction of a requirement specification, that is, every specification of the sum function is constructed, and its result is shown only at the end of the flow.
- Regressive: similarly to the Assert-First technique [8], the regressive method starts the construction of the user scenario from the result, and adds other interaction states specifically to reach the outcome (initial state). Fig. 2 shows the scenario where the requirement is constructed with the regressive method.

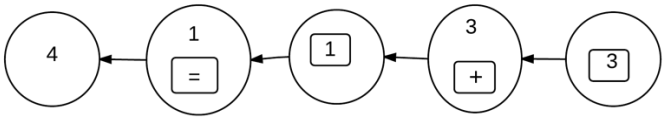


Figure 2. Regressive method to create the sum function scenario of a calculator.

III. ASSESSMENT OF THE PROPOSAL

The purpose of user scenarios is the specification of software requirements. Customers and developers to promote communication and collaboration can use these scenarios during development. However, it is important that such requirements be complete, consistent and realistic [11, 12].

To assess applicability and usefulness of the proposal in relation to completeness and consistency of requirements represented by non-technical[1] participants, we conducted an experiment. The experiment aimed at the construction of user scenarios of the 8-Puzzle game. We proposed to investigate the following questions:

RQ1: What quality factors (completeness and correctness) of the requirements are represented in user scenarios?

RQ2: Are such quality factors (completeness and correctness) associated with progressive or regressive methods?

RQ3: Which of the proposed methods facilitates the construction of user scenarios?

A. The 8-Puzzle Game

8-Puzzle is a game consisting of a board with 3 rows and 3 columns. The board has a number sequence from 1 to 8 and an empty space. The goal is, starting from a random state, to order the sequence of numbers. Fig. 3 shows the final state.

1	2	3
4	5	6
7	8	

Figure 3. Final state of the 8-Puzzle game.

The operations for the configuration of the board include moving the empty space up, left, right or down. In digital implementations, the empty space is moved by using the keyboard arrow keys or by clicking on a number next to the empty space.

B. Methodology for assessment

For the assessment, we considered the construction of user scenarios for the requirements specification of the 8-Puzzle game with non-technical participants. The materials needed were: a pencil or pen, an eraser, and blank paper. Fig. 4 shows the diagram with the activities carried out during the three evaluation stages: preparation, experiment and result analysis.

[1] Non-Technical participants are not knowledgeable about UIDs, FIT, and Automated Acceptance Testing.

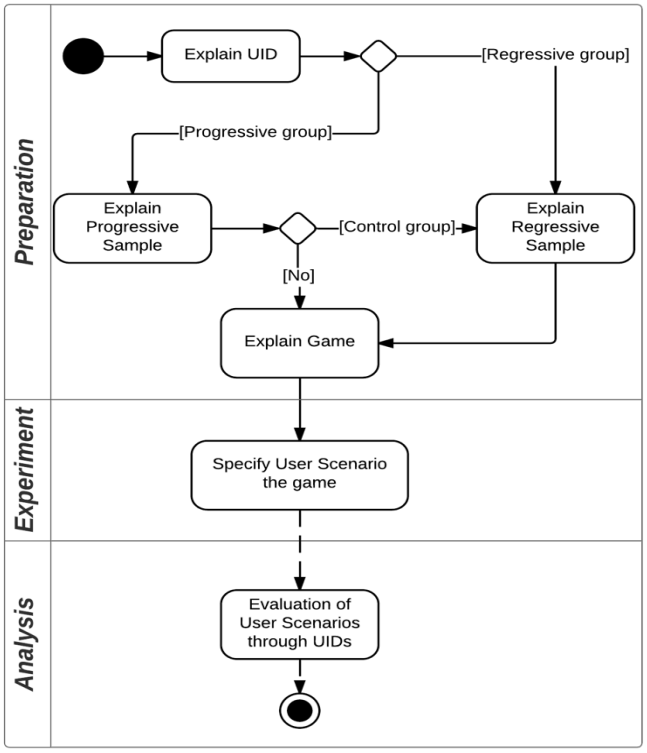


Figure 4. Diagram of activity for assessment of user scenarios through UID.

1) Preparation: In order to answer the questions and analyze the differences between the methods, the participants were divided into three groups: progressive, regressive and control (progressive/regressive). In the preparation stage, the participants were trained for about 15 minutes, according to each group. So, during the preparation stage of the progressive group, explanation activities were carried out about: UIDs, the progressive method, and the 8-Puzzle game. The regressive group carried out explanation activities about: UIDs, the regressive method, and the 8-Puzzle game. For the control group, explanation activities were performed about: UIDs, the progressive and regressive methods, and the 8-Puzzle game. This way, the participants of the control group made their own choice of method during the experiment. During the explanations, the participants were allowed to use a pencil and some paper in case they wanted some practice.

2) Experiment: In the experiment stage, each participant had to specify the final state of the 8-Puzzle game, or victory, still considering some user interaction for the board configuration. The user scenarios had to be developed using paper and a pencil or pen, at the participant's choice. Each participant had the maximum time limit of 15 minutes for the experiment performance. The time spent by each participant to complete the task was collected during the experiment.

C. Analysis of question RQ1

Fig. 6 shows an expected user scenario of the 8-Puzzle game.

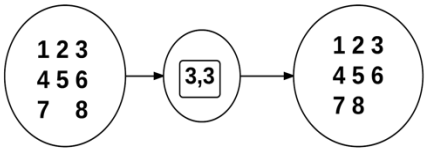


Figure 5. Expected user scenario of the 8-Puzzle game.

User scenario variations such as user input to move the pieces (numbers) on the board, quantity of interactions, and direction of transition flows were considered adequate even being different from the diagram in Fig. 5. Question RQ1 can be answered by evaluating the user scenarios. The analysis considered the evaluation of the quality factors: completeness and correctness.

1) Completeness: It means that all user-required services must be defined [11, 12]. Completeness can be evaluated by assigning complete/incomplete values. It is considered to be complete the user scenario that presents:

- The end result or state of victory; and
- At least one user input to the board configuration.

In [13], incompleteness is defined as ambiguity type. It occurs when a statement fails to provide enough information to have a single clear interpretation. It is considered to be incomplete the user scenario that:

- Does not present state of victory;
- Does not present interaction of playing; or
- Is incorrect.

2) Correctness: It is the quality factor indicating whether the participant understands and correctly applies the UID language. Correct/incorrect values are assigned for this quality factor. The correct value is assigned to the user scenarios where the language symbols are properly applied in accordance with Table I. The incorrect value is assigned to the user scenario that:

- Contains cyclic transitions;
- Contains transitions to more than one interaction state;
- Contains transitions to random values; or
- Does not consider the flow of transitions.

D. Analysis of Question RQ2

Question RQ2 can be answered by the formula below with the following hypothesis:

$$H_0 : F_{Progressive} = F_{Regressive}$$

$$\text{And} \qquad (1)$$

$$H_1 : F_{Progressive} \neq F_{Regressive}$$

- $F_{Progressive}$ is completeness and correctness of the user scenarios progressively specified; and

- $F_{Regressive}$ is completeness and correctness of user scenarios regressively specified.

The decision to accept H_0 or H_1 is made from the data collected from the experiment. By accepting the H_0 hypothesis, it can be stated that the quality factors of the expressed requirements are indifferent, that is, these quality factors are not associated with the method. However, by accepting the H_1, hypothesis, we affirm that the quality factors are different between the two methods, that is, these quality factors are associated with the method. The statistical test significance level should be at least 5% ($\alpha = 0.05$).

E. Analysis of Question QR3

The easiness or effort to construct the user scenarios are analyzed through time data. Therefore, it is necessary to carry out a distribution analysis of the time spent by the participants according to the method used for comparison.

IV. RESULTS

A. Preparation Stage

The experiment was conducted with 21 participants. The participants were prepared according to the progressive and/or regressive methods, resulting in three groups. The progressive group consisted of 8 participants. The regressive group was composed of 6 participants. The control group consisted of 7 participants. The graph in Fig. 6 shows the education level of the participants.

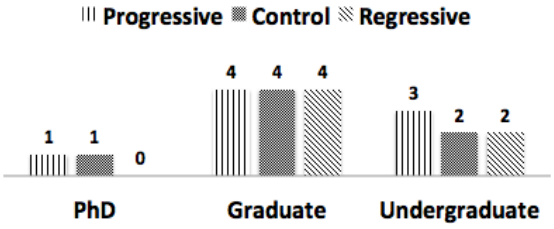


Figure 6. Education level of the participants.

The participants' ages range from 22 to 45 years. Fig. 7 shows box plots with the participants' age variation in each group.

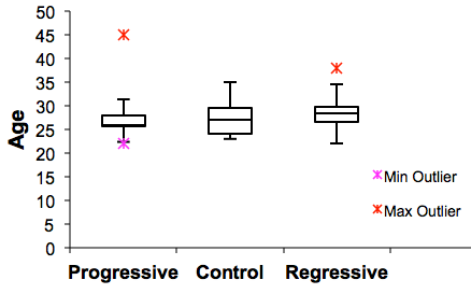


Figure 7. Participants' age variation in each group.

B. Experiment results

The user scenarios delivered by the participants were initially classified according to the progressive or regressive methods. Ten participants used the progressive method and eleven participants used the regressive method. The choices of the control group were 29% progressive method and 71% regressive method. Table II presents the correlation matrix between methods used during preparation and methods used by the participants during the experiment.

TABLE II. CORRELATION MATRIX BETWEEN PREPARATION AND EXPERIMENT.

Correlation Matrix		**Experiment**	
		Progressive	Regressive
Preparation	Progressive	8	0
	Regressive	0	6
	Control	2	5

The quality factors (completeness and correctness) of user scenarios are evaluated according to Section IV (Analysis of question RQ1).

1) Complete and Correct: The assessment considered fourteen of the user scenarios as complete and correct. As an example, Fig. 8 shows a user scenario that applied the progressive method, which was assessed as correct and complete.

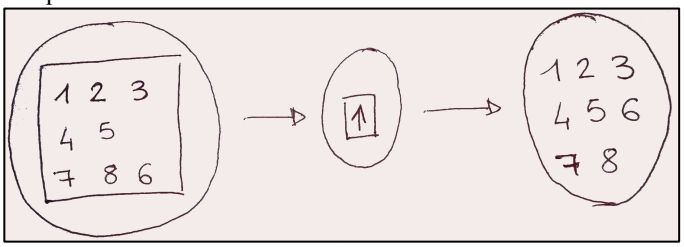


Figure 8. Complete and correct user scenario using the progressive method.

Fig. 9 and 10 show two scenarios that applied the regressive method and were assessed as correct and complete.

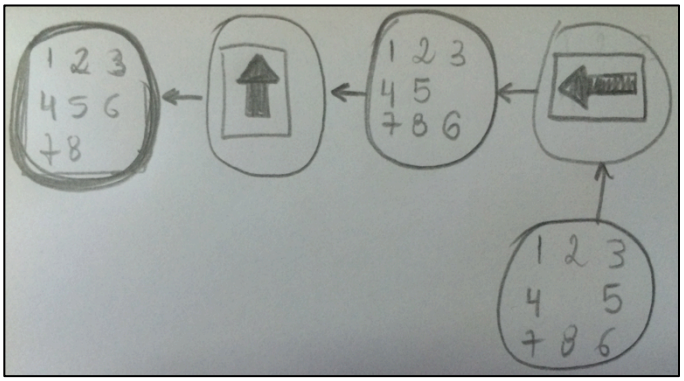


Figure 9. Complete and correct user scenario using the regressive method, applying movement to the number adjacent to the empty space.

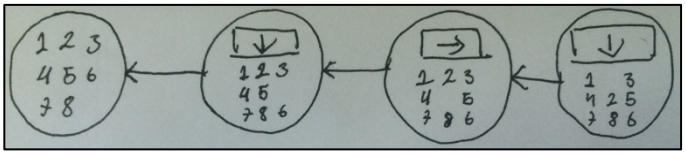


Figure 10. Complete and correct user scenario using the regressive method, applying movement to the empty space.

2) Incomplete and Correct: The assessment considered five of the user scenarios as incomplete and correct.

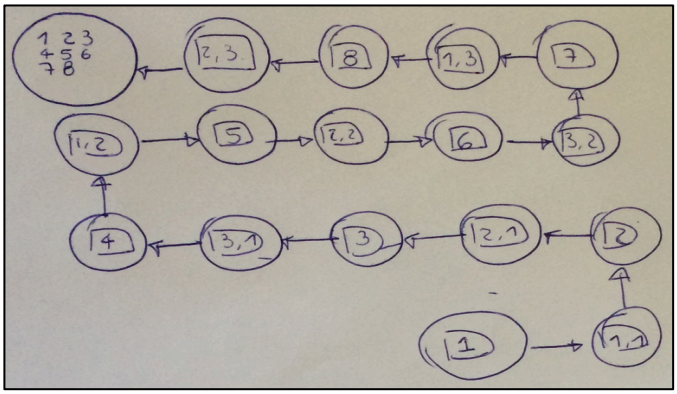


Figure 11. Incomplete and correct user scenario using the regressive method, not specifying interaction of playing.

3) Incorrect and Incomplete: In only two cases, the participants used UID incorrectly. In such cases, it is assumed that the requirement was incomplete due to absence of syntax.

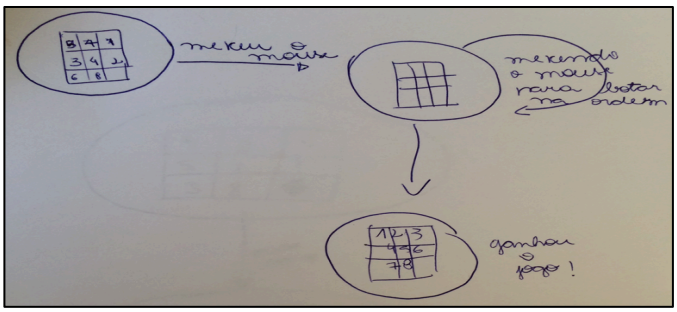


Figure 12. Incomplete and incorrect user scenario using the progressive method, an unintelligible user scenario.

In an overall assessment result, the participants delivered 67% complete user scenarios where 90% of them used UIDs correctly.

The graph in Fig. 13 displays the completeness and correctness distribution divided according to the scenario method of construction.

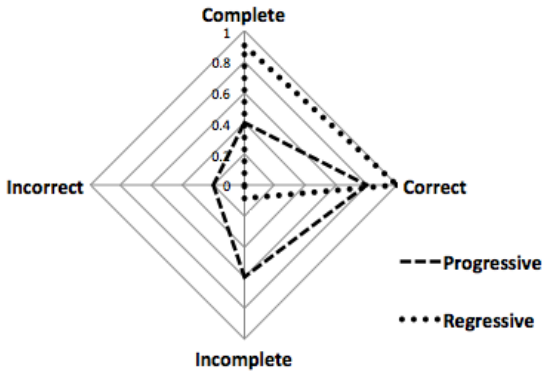


Figure 13. Graph showing the probability of correctness and completeness, divided according to construction method.

Axis "x" on the graph in Fig. 13 represents the distribution of correctness and axis "y" represents completeness. The positive area of axes "x" and "y" represents the best quality factor. It can be seen that the user scenarios specified by the progressive method have a 40% probability of being complete, and 80% of being correct. And the regressive method has 91% probability of being complete, and 100% of being correct (RQ1).

To demonstrate the difference of quality factors between the methods, we used Fisher's statistical test [14].

```
> n
             Evalution
Method        Complete and Correct Incomplete and Correct Incomplete and Incorrect
  Progressive                    4                      4                        2
  Regressive                    10                      1                        0
> fisher.test(n)

    Fisher's Exact Test for Count Data

data:  n
p-value = 0.04928
alternative hypothesis: two.sided
```

Figure 14. Fisher's statistical test applied with the statistical tool R.

According to Fisher's statistical test, p-value is 0.04928. However, the confidence level of the test must be ($\alpha = 0.05$). Thus, $p - value < \alpha$ then the alternative hypothesis (H_1) is considered, allowing to conclude that the quality factors are different between the methods (RQ2).

The box plot in Fig. 15 shows the distribution of the time spent divided according to the construction method.

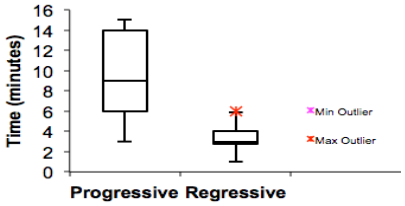


Figure 15. Distribution of time spent, according to construction method.

The time spent by the progressive group is between 3 and 15 minutes, and the regressive group is between 1 and 6 minutes. The regressive method has an outlier. The outlier (6 minutes), out of the distribution of time spent by the group, indicates that a participant may have found it more difficult to specify and develop the user scenario. The average time used by the progressive and regressive groups is 9 minutes and 3 minutes, respectively. Thus, according to the time distribution analysis, the regressive method involves less effort (RQ3).

V. Threats to Validity

While the 8-puzzle is simple and effective for experimental purposes, abstracting from this game scenario to the use of UID in real-world software project requirements is not easy. The case study research is incomplete without a discussion of concerns that may threaten results validity. Internal validity refers to the causal inferences made based on experimental data [15]. Our goal is simply to determine whether and how participants create user scenarios through UIDs.

The case study has two important considerations: knowledge of the case study; creation process of user scenarios.

The knowledge of the case study by participants refers to the domain on the 8-puzzle, considering the logic and rules. This knowledge is common to the participants. The user scenarios creation process is different from the process of reproduction (copy) of user scenarios. An example of reproduction is to copy the calculator user scenario (Fig. 1) and simply change the user setting values. Our case study avoids the process of reproduction of the participants, considering only the creation of user scenarios. We also observed that the problem domain allowed the evaluation of the effort to create user scenarios without interruption for fatigue.

Construct validity refers to the appropriate use of evaluation metrics and measures [15]. We specifically avoid the use of absolute measures of completeness and correctness conforms the term as expressed in accepted IEEE standards [16]. To calculate other statistical measures, we used accepted statistics for agreement (Fisher's Exact Test) and scrupulously followed recommended practices in applying them.

External validity refers to the ability to generalize the findings to other domains [15]. The external validity of research contains two threats: the problem domain studied and the population of participants. The problem domain (8-puzzle) contains user interactions with the system. These user interactions are similar, such as: a calculator; authentication systems; or in areas of other studies that consider the applicability of UIDs in software engineering, as in [9] [10] [17].

Unfortunately, the study used a small population of participants, rather than a large population of participants. However, the population of participants was composed of different educational levels, and related areas of business, engineering and science.

Reliability refers to the ability of other researchers to replicate methodology [13]. We detail the proposal and the evaluation technique and the result, and we consider important other researchers to reproduce our study.

VI. CONCLUSIONS

The challenge of eliciting requirements from customers is worthy of investigation and so is any effort to simplify or assist in the process. The paper does both and some interesting results are shared. The applicability of user scenarios in the present study is related to agile software development, where requirements are customer expectations and should as well be used as tests for the application code. UIDs were used to allow non-technical customers to represent user scenarios. In this context, UIDs were quite suitable for creating user scenarios to specify software requirements.

As for the assessment of this proposal, an experiment was conducted with 21 non-technical participants to specify the requirements of a game. With statistical analysis of the experiment results, it was observed that the progressive and regressive specification methods are different. The regressive method resulted in 91% of complete requirements while the progressive method resulted in 40% of complete requirements. The participants delivered 67% complete user scenarios where 90% of them used UIDs correctly. However, in our study case, the novelty of the proposed regressive method based on the TDD assert-first technique is the reduction of effort, and improvement in the assessed quality factors of the requirements.

In spite of the fact that this study has considered only UIDs for representing user scenarios, a tool for automated acceptance tests is being built, with support for direct execution of user scenarios.

REFERENCES

[1] R. Miller and C. T. Collins, “Acceptance testing,” Proc. XPUniverse. 2001.

[2] B. Haugset and G. K. Hanssen, “Automated acceptance testing: A literature review and an industrial case study,” Agile, 2008. AGILE'08. Conference, IEEE. 2008, pp. 27-38.

[3] T. Dybå and T. Dingsøyr, “Empirical studies of agile software development: A systematic review,” Information and software technology. Elsevier, vol. 50, 2008, pp. 833-859.

[4] G. K. Hanssen and B. Haugset, “Automated acceptance testing using fit,” System Sciences, HICSS'09, 42nd Hawaii International Conference on. IEEE, 2009, pp. 1-8.

[5] L. F. S. Hoffmann, L. E. G. D. Vasconcelos, E. Lamas, A. M. D. Cunha and L. A. V, Dias, “Applying Acceptance Test Driven Development to a Problem Based Learning Academic Real-Time System,” Information Technology: New Generations (ITNG), IEEE, 11th International Conference on. 2014, pp. 3-8.

[6] K. Alvestad, “Domain Specific Languages for Executables Specifications,” Institutt for datateknikk og informasjonsvitenskap. p. 63, 2007.

[7] M. Kamalrudin, S. Sidek, M. N. Aiza, and M. Robinson, “Automated Acceptance Testing Tools Evaluation in Agile Software Development,” Sci. Int. 2013, pp. 1053-1058.

[8] K. Beck, Test Driven Development: By Example. Addison-Wesley Professional, 2003.

[9] N. Güell, D. Schwabe and P. Vilain, “Modeling interactions and navigation in web applications,” Conceptual Modeling for E-Business and the Web. Springer, 2000, pp. 115-127.

[10] P.Valderas, V. Pelechano, “A survey of requirements specification in model-driven development of web applications,” ACM Transactions on the Web (TWEB). ACM, Vol. 5, p.10, 2011.

[11] I. Sommerville, Software Enginnering. 9th ed, p. 773. Addison-Wesley, 2011.

[12] A. D. Lucia and A. Qusef, “Requirements engineering in agile software development,” Journal of Emerging Technologies in Web Intelligence. vol. 2, 2010, pp. 212-220.

[13] A. K. Massey, R. L. Rutledge, A. I. Anton, P. P. Swire, “Identifying and classifying ambiguity for regulatory requirements, ” Requirements Engineering Conference (RE), 2014 IEEE 22nd International, IEEE, 2014, pp. 83-92,

[14] E. L. Lehmann, and J. P. Romano, “Testing statistical hypotheses”. Springer, 2006.

[15] R. K. Yin, *Case Study Research: Design and Methods*, 3rd ed., ser. Applied Social Research Methods Series, L. Bickman and D. J. Rog, Eds. Sage Publications, 2003, vol. 5.

[16] IEEE Recommended Practice for Software Requirements Specifications," IEEE Std 830-1998 , pp.1-40, 1998.

[17] N. V. Zeferino and P. Vilain, “A model-driven approach for generating interfaces from user interaction diagrams,” Proceedings of the 16th International Conference on Information Integration and Web-based Applications & Services. ACM, 2014, pp.474-478.