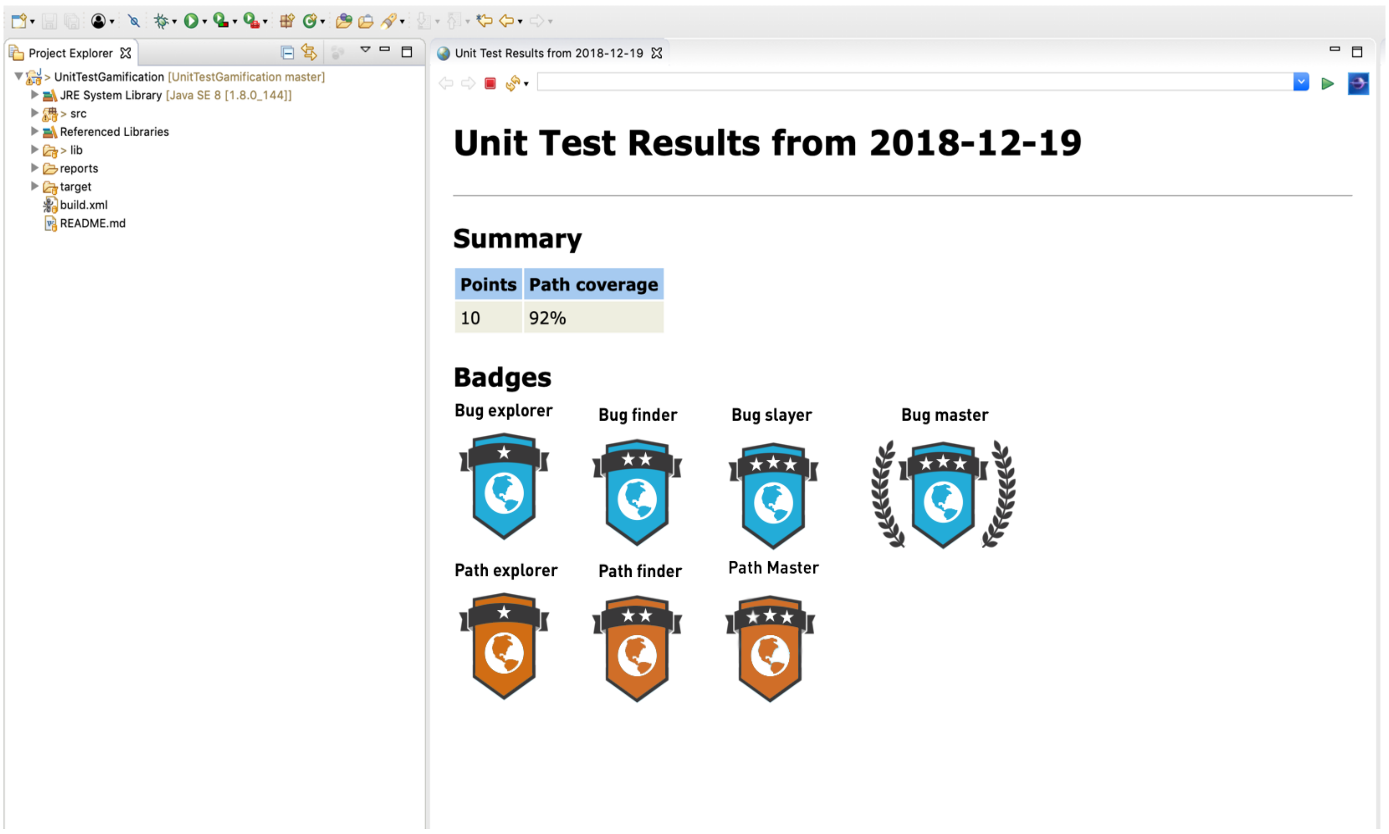
CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG


Project Explorer

UnitTestGamification [UnitTestGamification master]
- JRE System Library [Java SE 8 [1.8.0_144]]
- src
- Referenced Libraries
- lib
- reports
- target
- build.xml
- README.md

Unit Test Results from 2018-12-19

Unit Test Results from 2018-12-19

Summary

Points	Path coverage
10	92%

Badges

The impact of gamification in unit testing

A controlled experiment

Bachelor of Science Thesis in Software Engineering and Management

Shafiq Saloum
Fredrik Rissanen

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

The impact of gamification on unit testing
A controlled experiment to determine whether the introduction of gamification in unit testing affects the motivation and interest on developers.

SHAFIQ SALOUM
FREDRIK RISSANEN

Supervisor: JAN-PHILIPP STEGHÖFER
Examiner: RICHARD SVENSSON BERNTSSON

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

[Cover:
A preview of the unit testing gamification tools. Image and explanation can be found in the result section on page 6.]

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

The impact of gamification in unit testing
A controlled experiment

Fredrik Rissanen
Computer Science and Engineering
Gothenburg University
Gothenburg, Sweden
Email: gusrisfr@student.gu.se

Mohamad Shafiq Saloum
Computer Science and Engineering
Gothenburg University
Gothenburg, Sweden
Email: gussalomo@student.gu.se

***Abstract*—[Context] Unit testing is one of the most widely used tools to find bugs in software projects. The task of writing unit test is by some considered to be a tedious task. When finding a task tedious there may be a lack in motivation which can reduce the quality of the tests resulting in less bugs found. [Objective] In this research, we examine gamification and its effect on developer motivation and quality of unit tests. [Method] We have conducted a controlled experiment with 14 subjects that wrote unit tests to find bugs where we measured the motivation levels, the number of bugs found and the percentage of path coverage. [Results] The results did show a statistically significant difference between the control group and the experiment group where the experiment group was more motivated and found more bugs. However, the results did not show a statistically significant difference for path coverage. [Conclusion] Our research showed statistically significant difference when using gamification for motivation and number of bugs found, not when looking a path coverage. However, further research is required with a larger number of subjects and over a longer period of time to find more conclusive results.**

***Keywords*-gamification; unit testing; path coverage; motivation; bugs;**

I. Introduction

Unit testing is a verification method to reveal program deficiencies. Unit testing is either performed by a developer or by a dedicated tester. The developers and or testers are responsible for ensuring that the functionality of the code is correct [1]. Testing is one of the most widely used software validation techniques [2] due to its capability to find many bugs in software [3], however, the task of writing unit tests can be a repetitive and time-consuming task that many developers feel is tedious [4].

The quality of unit tests varies and the quality is highly dependent on the skill of the engineer who wrote the test [5]. If the developer is under the pressure of strict deadlines the developer might not be motivated [5] to follow best practice guidelines [6] leading to increased maintenance cost [5].

Unit testing is mainly approached by providing the developers with guidelines on how to perform the tasks [5]. The guidelines often contain information such as recommended amount of testing and what format the tests should follow. The issue in treating unit testing like any other development task is that there might be a lack in motivation when performing a repetitive task such as unit testing [4].

When attempting to motivate and engage users, the concept of gamification has shown beneficial effects [7] [8]. Also, gamification has shown to assist in learning and to reduce the number of failures, in particular in cases where a new skill is used [9].

Research approach: This research follows an established framework called The Kaleidoscope of effective gamification [10] to identify suitable gamification elements , and integrate the prototype in to an environment where developers commonly work.

The field of gamification in software engineering has gotten some attention and research over the last decade [8] [7] [11]. Gamification is also research to some extent when it comes to software testing [12] [13] and we only found one paper that has a strict focus on gamification in unit testing [14]. These papers cover similar topics as this paper will. However, these papers are researched by multi-vocal literature ,case studies or design science where the participants subjectively state the difference before and after they were exposed to gamification. Our research is needed because there is still too little research in gamification for unit testing and this research will compare subjects that are exposed to gamification and subjects who are not. Another motivation for this study is that it follows an established framework as mentioned above. The benefit of using an established framework together with a controlled experiment is that it will be easy for other researchers to replicate this research to either confirm or disprove our findings. With the importance of unit testing and the risk that the developer feel that the task is tedious we believe that gamification is important to further research in relation to unit testing. To research this field the paper will aim to answer the following research question:

RQ1: Does gamification increase the motivation of developers and improve the quality of unit tests?

A controlled experiment was performed to answer the stated research question. The experiment was conducted with 14 participants. Seven of the participants wrote and executed unit tests in a gamified environment, the remaining seven wrote and executed unit tests in a regular testing environment. This research aims to test if gamification can increase developers' motivation and the quality of unit tests.

The remainder of the paper is structured as follows: Section II discusses the background of gamification and unit testing in software engineering. Section III explains the research methodology for this paper giving an understanding on how to replicate the research. Section IV contains the results from the experiment. In Section V the authors discuss the results and what implications they have. Section VI concludes the research.

II. Background and Related Work

This section covers the background of this research as well as previous, related studies. Unit testing is discussed and the use of gamification elements in software engineering.

A. Unit Testing in Software Engineering

The definition of unit testing is according to Whittaker: "unit testing tests individual software components or a collection of components. Testers define the input domain for the units in question and ignore the rest of the system." [15]. Unit testing is an attractive quality management tool where software components are tested in isolation [16].

The IEEE standard for software unit testing [6] explains how a unit test should be planned, executed, and evaluated. The standard is quite extensive, and it was used in this research to assist in what features of testing should be gamified. Even though the standard is a well-crafted document, it is not especially pedagogic. This research aims to implement the standard in to software with the help of gamification to deccrease the learning curve.

In the research by Yao et al. [1] the importance of unit testing is discussed. The authors stress the importance of unit testing as a verification method to find program deficiencies, and it is often the developers' responsibility that the code functions correctly. It is then important that the developers know how to write good unit tests, since the quality of the tests is highly dependent on the expertise of the developer [17]. The expertise of developers may vary though, as stated in the research by Bavota et al. [5] the authors state that not all developers follow best practice guidelines [6], mostly due to inexperience. The authors also state that not following these guidelines will eventually lead to increased maintenance cost, especially for unit tests, and as Clark et al. [2] stated, bugs found late are vastly expensive for companies.

Testing usually accounts for half of the research and development budget in software development organizations [3]. There is reaserch that tries to automate the creation of tests by different software solutions and decrease the cost, however, the automated software can still only find obvious faults [3] and there is still a need for a human to perform testing.

While these existing studies investigate the importance of testing and the cost of bugs, there is to our knowledge no studies from the perspective of unit testing that is dedicated to improve the unit testing process by making it more engaging.

B. Gamification in Software Engineering

The definition of gamification is according to Deterding et al. "the application of game design elements in non-game contexts." [18] where non-game context for this research will be unit testing. The impact of gamification in software development is analyzed by Platonova and Berzisa [7] and their research indicates that gamification could improve the quality of development, and also assists in learning new skills. The authors define relevant phases of development where gamification is appropriate such as definition of requirements, coding and testing.

In software development gamification is often designed ad-hoc [4] [8] leading to solutions that are hard to replicate and evaluate. This research aims to create replicateable gramification research and bases its design on previously studied frameworks from Garcia et al. [4] and Kappen & Nacke [10]. Gamification aims to foster engagement in tasks that developers may find tedious and boring such as unit testing [4] by rewarding them and making the work fun. However, it is important to not only focus on rewards and extrinsic motivation [8] [10], there must also intrinsic motivation in terms of clear goals and the clear connection to personal development for the developer.

The research by Pereira et al. [8] systematically maps and analyzes previous research in the field of gamification. The paper states that there are many benefits shown in gamification, mainly in motivation and in engagement. However, the paper concludes that previous research is only preliminary or even immature. The previous research is mainly considered to be immature in the terms of poor methodology description making the research hard to replicate. The lack of research methodology in previous research motivated this research to create a unit testing gamification method with a sound, replicable methodology to further extend to the scientific knowledge in gamification.

Gamification have been applied in several software areas such as educational, development and testing. In education research have shown promising results where the students became a lot more motivated and engaged in class than before [9] [11]. Gamification for development have also shown beneficial effects, not only for facilitating the work but also in learning [7]. In the research by Pedreira et al. [8] they argue that gamification is a trend that might not always be suitable, however, they do believe that gamification is suitable for unit testing. In the reseach by arnarsson and jóhannesson [14] they perform a case study with a company where gamification is used to achieve path coverage and detect unit test smells. In study the developers were awarded badges or points and placed in a leaderboard based on the number of unit test cases written, path coverage and unit test smell detected. Their results indicated that for almost all subject's gamification increased motivation and more test cases written. Their results indicate that leaderboards were the most appreciated tool, however they believe this might have been a result of an unproportionate badge system. In the reseach by G. Fraser

[13] on the other hand he believes that points is the most suitable tool.

III. Research Methodology

A controlled experiment [19] was chosen as the research method for this paper.

A. Experiment Design

The experiment is designed as a one-factor experiment with two treatments. The factor is the inspection process Unit Testing. The first treatment is the use of software with gamification and the second is the use of software without gamification. The subjects using the gamified software are referred to as the experiment group and the group using regular software is called the control group. The experiment was run on the experimental unit which was a Java project with planted bugs for the subjects to discover with JUnit tests. The subjects applied the different testing techniques (testing with and without gamification) on the experimental unit. The research aims to answer the research question stated in section I:

> **RQ:** Does gamification increase the motivation of developers and improve the quality of unit tests?

In this research, the quality of unit tests is determined by the number of bugs found and the path coverage in the project. Quality of unit tests can be determined in many ways. To find bugs is the main goal when writing unit tests making it an obvious aspect to measure. Bugs were distributed across the system and the subjects did not know where. For more information about the system please refer to section III.C. In order for them to make sure that they have tested all outcomes path coverage was also found as a suitable aspect to measure. Based on the aspects of test quality and motivation the following dependent variables were formulated: the number of bugs found (*numBugs*), percentage of paths covered (*pathCoverage*), and the motivation to complete the task (*motivation*). This leads to the following hypotheses:

- H0*numBugs*: There is no significant difference in *numBugs* between the control group and the experiment group.
- H1*numBugs*: There is a significant difference in *numBugs* between the control group and the experiment group.
- H0*pathCoverage*: There is no significant difference in *pathCoverage* between the control group and the experiment group.
- H1*pathCoverage*: There is a significant difference in *pathCoverage* between the control group and the experiment group.
- H0*motivation*: There is no significant difference in *motivation* between the control group and the experiment group.
- H1*motivation*: There is a significant difference in *motivation* between the control group and the experiment group.

The experiment investigates whether there is a causal relationship between the use of gamification elements and the *motivation* of the developers. Also, if there is a correlation relationship between the variable *motivation* and the variables *numBugs* and *pathCoverage*.

B. Gamification Design

The design of the gamification elements was decided to follow kaleidoscope [10] a model for effective gamification. The kaleidoscope model consists of a core and four layers as seen in Fig 1. The core establishes the focal point of player experience [10] and represents the core objectives of the gamification design. For this experiment the core objectives are to increase the developer motivation and to improve the quality of unit tests. The Motivated behavior level focuses on the different intrinsic and extrinsic motivations, for this experiment it means to find what is an engaging gameplay experience for the intended end user.

Intrinsic motivations are the motivations that drives the developer without continuous external influence. This is facilitated by setting up goals for the individual developers for them to see how their competence level increases. Intrinsic motivation is hard to design for an experiment since it focuses on long-term effect. However, the experiment aims to facilitate the intrinsic motivation by setting up a reachable goal (finding all the 10 bugs and reaching 90% path coverage). 90% test coverage was decided instead of 100% to implicate that it is more important to find all the bugs than to get 100% coverage, coverage is after all not very useful if it does not find the bugs. The number of bugs planted was decided by the motivation of having a task that was not too easily achieved but still possible to achieve. It is important that it is possible to win in terms of extrinsic motivation [10].

The extrinsic motivations are external motivators such as badges, points and leaderboards. The Motivated behaviour level drives the Game experience level being rules and structure for the design. The experiments rules are conveyed in text form at the start of the experiment where the subjects will have time to read the goals and the challenges that they are facing. The game design process layer is the actual design that the subjects will see.

The features that were implemented are points and badges. Points were selected since it assists in adding up to a sum which is easily measurable and easy for the developer to understand. One point was given per bug found. Badges were selected since it works as a milestone for the developers to see where they are and to see that the points sum up to something. Two types of badges were implemented. The first type was given based on the sum of points received for finding bugs. The second type was given based on achieved path coverage. Leaderboards were considered but excluded, having a third element might be too much to grasp for 60 minutes of testing, and leaderboards bring a competitive element which was not sought after in this experiment. How a subject reacts to competition is very individual and possibly harder to measure, therefore the authors decided to only make the experiment with points and badges. When a test is executed a file will open displaying the current points and badges achieved. For

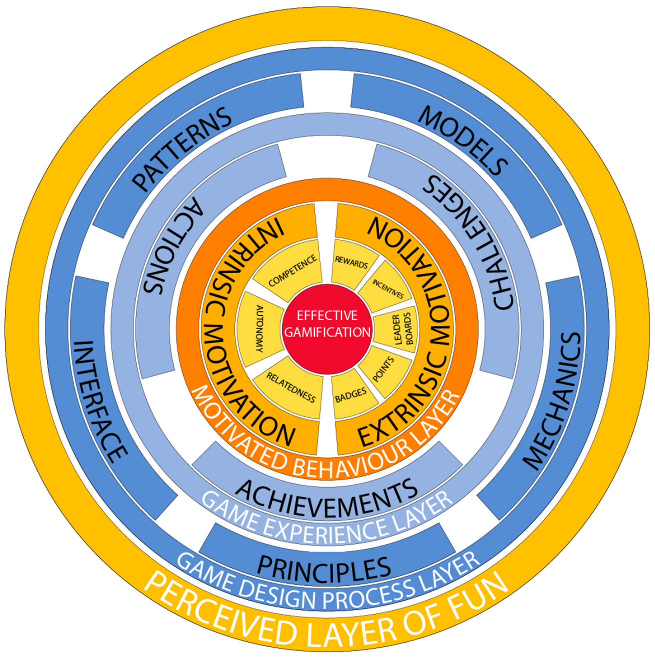


Fig. 1. Kaleidoscope of effective gamification.

implementation details please refer to section C. The badges implemented are called:

* Bug explorer (1 points)
* Bug finder (3 points)
* Bug slayer (6 points)
* Bug master (10 points)
* Path explorer (50% path coverage)
* Path finder (75% path coverage)
* Path master (90% path coverage)

C. Technical implementation

To avoid bias from the author's a project that matched the criterion was downloaded from Github. The project is a Java project in a typical object-oriented modular way. The project was examined by the developers and then bugs were introduced. The bugs that we introduced are logical errors. An example of a bug that is introduced is in a statement to check if a is less than b instead of a is less than or equal to b making the statement provide a faulty output. The downloaded project contains the functionality of a food ordering system. It has a command line interface where the subjects can test the system and see how it works. The system works in the way that you can add either drinks or lunch. Drinks and lunch has in its' own categories with items. You can at any time add, remove, cancel or complete the order. When an order is placed the total is displayed. The structure of the project is split up in to models, views and controllers. The models contains all the different type of items available for ordering. The view contains the text output for the command line interface in which the subjects can test the system. The controller controls the logic in which the view is controlled. We added a skeleton test class with an example of a test that finds a bug and a structure that the subjects can use when writing tests. The project is approximately 600 lines of code and there are 10 planted bugs. Due to the limited time the developers have to perform the experiment the complexity of the placed bugs is decided to be kept low.

Integration of gamification into Eclipse is done by using the Java library ANT [20] to generate JUnit reports programmatically. The generated reports in HTML format includes the points and the badges achieved based on the test result.

D. Experiment operation

The experiment was run on two different occasions, once with the experiment group and once with the control group. The groups were decided to be kept apart so that they could not influence each other. Both groups had the same experimental environment: Windows and running the same version of eclipse. At the beginning of the experiment the subjects received an introduction to the experiment. The introduction covered background information about the project and an example of what it should do. The subjects received information on the location of the test skeleton files and we walked through a very basic example with a projector where we wrote a test case for a part of the code and found a bug. The introduction took 12 minutes for the control group and 15 minutes for the experiment group. In the extra three minutes for the experiment group we explained where to press to activate the gamification features and where the report landed after it was generated. After the introduction both groups had one hour of testing.

When questions arose, the question was repeated to all subjects to make sure that everyone received the same information. During the testing phase the subjects wrote unit tests in their skeleton test file to discover bugs and then they were supposed to resolve the bugs. For the experiment group they first received a point when they resolved a bug. Every time the experiment ran a test case to see if they had resolved a bug a new report was generated for the subject to look at and see their current points and badges. The subjects were distributed across a computer lab so that there would be no risk of looking at other participants' screens and compromising the results.

When the testing was done all subjects answered a survey for 15 minutes. The purpose of the survey was to gather qualitative data about the experiment. Mostly about how motivated the subjects were to complete the task. Quantitative data was gathered from the computers that the subjects used directly after the experiment was completed. The data we gathered came from source files and the generated reports for the experiment group. From the data we derived the number of bugs found and the path coverage.

E. Data collection and analysis

The sample consisted of 14 developers with an academic background in software engineering. The subjects chosen had been students within the past year and all had experience with both Java and Eclipse. Since all subjects just recently left

TABLE I
THE DISTRIBUTION OF SUBJECTS' EXPERIENCE FROM THE INDUSTRY

Experience in industry	**Control group**	**Experiment group**
One year	4	4
Two years	2	2
Three years	1	1
Total	7	7

school the homogenity is very high. Also, the high homogeneity of student knowledge yields higher internal validity [21].

We sent a pre-experiment survey to 30 people that had achieved a bachelor's degree in software engineering within the latest six months. Out of the 17 respondents we received 3 were filtered out. One was filtered out because of total lack of industry experience whereas all the other subjects had. One was filtered out because having a lot more experience than the rest of the subjects. And one was filtered out because of too much experience of unit testing, the potential subject was working as a tester. After the filtering we had an even amount for all the years of experience as can be seen in table I. So, for each group we had 4 participants with 1-year experience in industry, 4 participants with 2 years of experience and 2 participants with 3 years of experience. All of these subjects were working at the side of school. All subjects were working as junior software developers, 8 of them were working as java developers and 6 of them were working as c# developers. To divide up the control group and the experiment group we first divided the subjects in to years of experience, we then looked at their current role and programming language and made sure that they would be distributed equally. We then distributed the sub-groups in to the experiment group and control group through random sampling to even out difference in knowledge and skill of testing from the subjects.

To analyse that collected data we performed statistical analysis. The analysis was performed on the quantitative data but also on the answers from the questionnaire. To determine the most fitting statistical method in order to compare the two sample groups data was gathered in the experiment and presented in diagrams. The number of subjects was a determining factor to which method to be used. This study contained a relatively small sample group which in turn made the data non-normally distributed. The small sample made is disregard a T-test and instead focus on a fitting method for data which is not normally distributed. The Mann-Whitney-Wilcoxon U test [22]. test was used to produce a p-value in order to determine whether there is a statistical significance between the two data sets.

F. Validity threats

There are several threats to validity in this study to be considered. The following sections will present and discuss the validity threats for our study and how we have tried to mitigate them. The validity threats discussed are a selection from the threats from [23]. The biggest validity threat for this study is the number of subjects. The number of subjects makes it hard for us to trust the reliability of the statistical analysis. This threat makes this study to be considered as an indication to where the study of gamification in unit testing might lead and calls for more research to either confirm or reject our findings.

Conclusion validity: The experiment included humans as subjects. When working with humans there is always a possibility that the amount of knowledge, experience and skill in the area varies from individual to individual. These factors can affect the result of the conducted experiment by making one group looking better than the other. We believe that we minimized the risk of these factors affecting the conclusion validity through random sampling.

Construct validity: To make sure that the treatment corresponded to the cause we were interested in [23] we made sure that all participants had experience with unit testing and we explained clearly how the experiment would be conducted with examples. We believe that this mitigated the largest threat to construct validity. Also, to made sure that there were no misinterpretations on the instructions we walked around in the room monitoring screens and were available for questions.

Internal Validity: A threat to internal validity is when a factor other than the treatment affects the outcome of the experiment [23]. To mitigate this threat, we controlled all variables we could. The participants used computers that we provided with software that we had tested. The subjects received that same time to test, the same introduction except for the gamification features and the same questionnaires except for the gamification features. To make sure that the experiment was understandable we conducted it in two iterations with two different developer. One developer for each iteration. In the first iteration we improved the introduction and after the second iteration we were assured that there was no ambiguity.

Before the subjects were selected they had to answer a questionnaire about their experience level as a developer and their knowledge of Java and unit testing. We believe that by the similar knowledge in unit testing presented by the answers in the survey and by random sampling the experiment group and the control group that we have mitigated most of the internal validity threats. Also, since all the subjects volunteered to join with no external reward we believe that the risk was low of having subjects with very high respectively very low motivation levels.

External Validity:

Even though the subjects we chose came from the same education all of them now have jobs, most at different companies with different roles. However, we cannot generalize that we elicited all possible context information since some of the subjects had more experience than others in unit testing. Also, since the number of subjects we tested on was low, we cannot generalize the results.

A threat to the external validity is that all subjects recently finished the same education. The same education that the authors studied. This makes it hard to generalize the results since our education might have affected how the experiment was created. To mitigate this threat the project was download

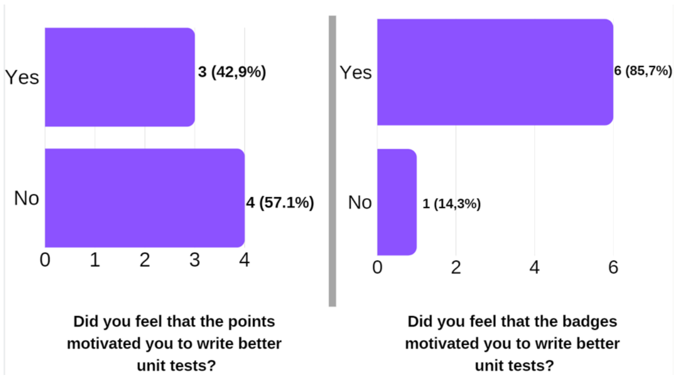


Fig. 2. Result from the post-experiment questionnaire where the experiment group is asked if badges and points motivated them to write better unit tests. The scale on the x-axis is the number of participants

from Github and not created by the developers. Also, we made sure that all subjects now had jobs at different companies and the subjects had different years of experience since some of them had been working while studying. We believe that this selection partly mitigated the threat to external validity.

IV. Results

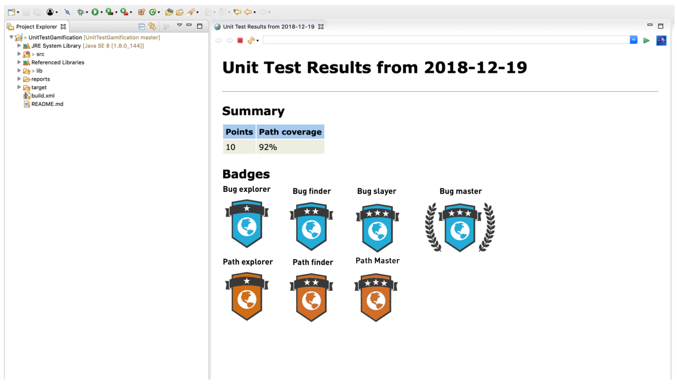
The result section provides the results gathered from the experiments. There are quantitative data gathered from the project we designed using Eclipse and qualitative data from the questionnaires.

A. Gamification features

This section provides the opinions gathered about the implemented gamification features. The implemented features are *badges* and *points* where the *points* summed up in to *badges* as stated in the methodology section. As can be seen in Fig 2 the *badges* motivated most of the subjects to write better unit tests while on the other hand there was a majority that felt that the *points* did not motivate them to write better unit tests. At this point the developers did not know if badges or points actually made them write better unit tests, but they stated that they were motivated to try to write better unit tests. There were indications from the free text comments in the questionnaire that the *badges* were more appreciated due to clearer and more playful graphical representation. The gamification features can be seen in Fig 3 where one *point* is awarded for every resolved bug. In the figure shown all bugs are resolved yielding all bug *badges* and a path coverage of 92% is achieved yielding all path coverage *badges*. When asking the subjects how hard it was to use the tool most answered that it was neither hard nor easy as seen in Fig 4.

B. Motivation to write unit tests

There are two charts displayed in Fig 5 the left one showing how motivated the experiment group was in general when writing unit tests and the right one showing how motivated they were in completing this task. The results incline that subjects were more motivated to complete the task with gamification features compared to when writing normal unit



Unit Test Results from 2018-12-19

Summary

Points	Path coverage
10	92%

Badges

Fig. 3. The gamification features as displayed to the subjects

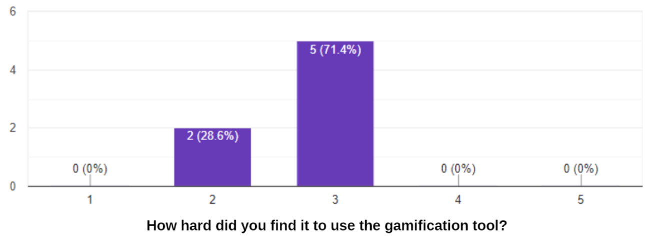


Fig. 4. Result from the post-experiment questionnaire from the experiment group showing the responses of how difficult the subjects believed the gamification system was. 1 is very easy and 5 is very hard

tests. If we compare the control group to the experiment group in motivation to complete the task as seen in Fig 6 it is visible that the experiment group who used the gamification were generally more motivated to complete the task.

When analysing the data from Fig 5 in table II the mean value of pre and post experimental results of the surveys together with a difference between them is displayed. A Mann-Whitney-Wilcoxon U test was used to find the p-value in order to determine the statistical significance. The result of the p-value is 0.11 which means that there is no statistical significance between the two sample groups. This analysis compared the experiment groups usual motivation to write unit

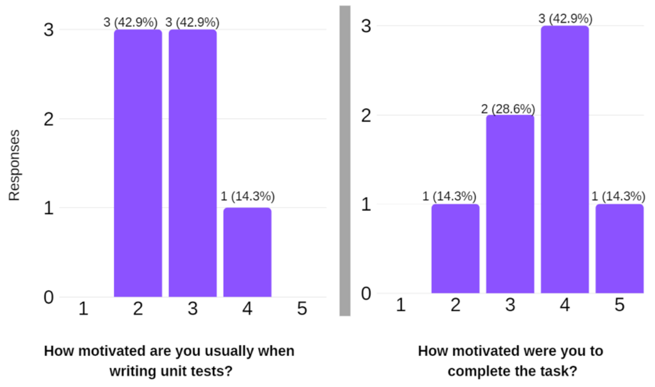


Fig. 5. The experiment group results from the pre-experiment questionnaire on the left and post-experiment on the right. 1 is ”Very unmotivated” and 5 is ”Very motivated”

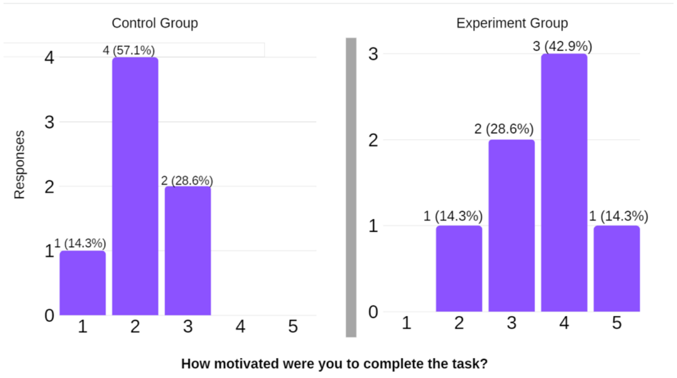


Fig. 6. Results from the post-experiment survey where the control group is on the left and the experiment group is on the right. 1 is ”Very unmotivated” and 5 is ”Very motivated”

TABLE II
DATA SHOWING THE MOTIVATION LEVEL FOR THE EXPERIMENT GROUP BEFORE AND AFTER THE EXPERIMENT

Experiment group	Value
Pre-experiment motivation	2.71
Post-experiment motivation	3.57
Difference	0.86
P-value	0.11

test compared to how motivated they were to complete this task.

When analysing the data from Fig 6 in table III it is visible that the difference between the mean values is 1.43. A Mann-Whitney-Wilcoxon U test was performed to find the p-value in order to determine if there was a statistical significance between the two sample groups. Results of the test shows that the p-value is 0.017 which means that there is a 98.3% chance that we are not rejecting a true null hypothesis. This data conflicts with the data from the paragraph above which stated that there was no statistical difference when comparing the experiment group before and after the experiment. However, since the hypothesis we wanted to test H0*motivation* is whether there is a statistically significant difference between the control group and the experiment group we can reject the null hypothesis.

By looking at the number of unit tests written and *pathCoverage* it is visible in Fig 7 that the gamification features motivated the subjects to write more tests and achieving high *pathCoverage*. By comments in the free text field we gained insights that some focused more on just covering all the code to get badges for *pathCoverage* instead of actually finding the bugs.

TABLE III
DATA SHOWING THE MOTIVATION LEVEL FOR THE CONTROL GROUP AND EXPERIMENT GROUP AFTER THE EXPERIMENT

Experience in industry	Control group	Experiment group
Mean (x bar)	2.14	3.57
Difference	1.43	
P-value	0.017	

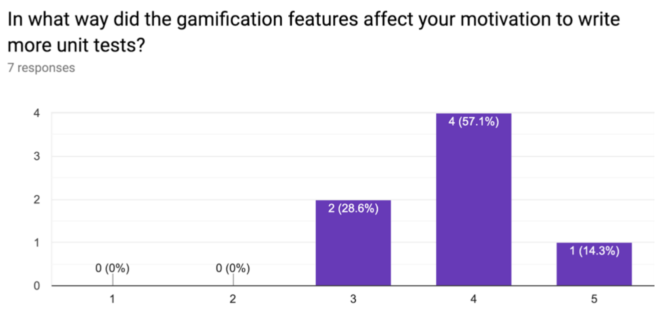


Fig. 7. Result from the post-experiment survey from the experiment group where 1 is ”Made me a lot less motivated” and 5 is ”Made me a lot more motivated”

TABLE IV
U-TEST ON QUANTITATIVE RESULTS GATHERED FROM ECLIPSE FOR BOTH THE CONTROL GROUP AND EXPERIMENT GROUP FOR NUMBER OF BUGS FOUND

Statistical values	Control group	Experiment group
Mean (x bar)	3.0	6.2
Standard deviation (σ)	1.63	2.87
p-value	0.045	

C. Quantitative data

It is observable in the Table IV that the experiment groups' mean was 6.2 for *numBugs* found while the control groups' mean was 3.0 for *numBugs* found. Table IV describes the statistical significance difference between the two data sets. The different number of detected bugs for both group was statistically significant under a Mann-Whitney-Wilcoxon test. The p value for the U-test came out to be 0.045, shown in table IV, which indicates that there is a statistical significance as stated before. A p value lower that 0.05 equates to a statistical significance in the given data sets when compared in a Mann-Whitney-Wilcoxon test. A boxplot further illustrates the bugs found in figure 8. Since we have been able to show a statistically significant difference between the control group and the experiment group we can reject the null hypothesis H0*numBugs*.

Table V showcases the mean, standard deviation and the p-value for both the control group and the experimental group. The mean was 40.6% for the control group and 63.9% for the experiment group. The Mann-Whitney-Wilcoxon test provided a p-value of 0.141 which is significantly higher than 0.05 which in turn mean that there is no statistical significance between the two samples. The boxplot in Fig 9 further helps illustrate the standard deviation for each set of samples. Since we have shown that there is no statistically significant difference between the control group and the experiment group we cannot reject the null hypothesis H0*pathCoverage*.

V. DISCUSSION

In this section we discuss the meaning of our results and how they relate to our research question:

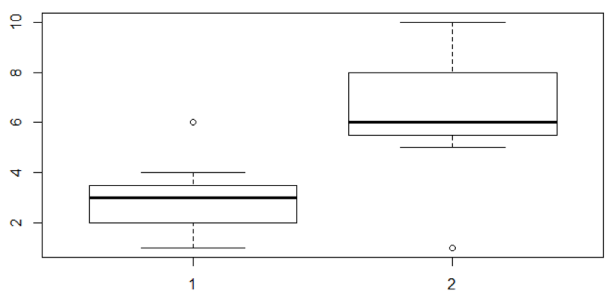


Fig. 8. Boxplot of both the sample groups compared to each other. Where the y-axis is in %, 1 is the control group and 2 is the experimental group.

TABLE V
U-TEST ON QUANTITATIVE RESULTS GATHERED FROM ECLIPSE FOR BOTH THE CONTROL GROUP AND EXPERIMENT GROUP FOR PATH COVERAGE

Statistical values	**Control group**	**Experiment group**
Mean (x bar)	40.6%	63.9%
Standard deviation (σ)	23.5%	29.5%
p-value	0.141	

RQ: Does gamification increase the motivation of developers and improve the quality of unit tests?

In the following sections the research question will be answered separately for motivation and quality of unit tests.

A. Statistical analysis

For the statistical analysis for motivation and quality of unit test we have the same issue as described in section III.F, there are few subjects for this study affecting the reliability of the results. Because of the small sample size our results may not be the most valid ones. By having a small sample our research is more vulnerable to individuals being very good or being very bad and individuals having strong opinions. However, when looking at the questionnaire there are no outliers that filled in the questionnaire different from the rest. When we look at the Quantitative data on the other hand we could see that there were some subjects who were a lot better than the rest and some who were worse. This is a serious threat to validity, but since we distributed the subjects based on their previous experience and skill and in the end random sampled them we believe that we got as reliable results as was possible for the sample size.

B. Motivation

Our results show that there is an inclination towards the experiment group feeling more motivated to finish the task than the control group. The results also shows that the experiment group had more motivation to complete the experiment with the gamification features than their usual level of motivation they had when writing unit tests. The motivation was measured for writing good unit tests, writing many unit tests and achieving high path coverage. The results show inclinations that the gamification features worked better for writing better unit test for actually finding the bugs than writing many unit tests. This could mean that the subjects believed that the number of bugs found was a better measurement than path coverage. The results also shows that the badges provided higher motivation levels than the points. This could mean that the developers were more motivated by the extrinsic rather than the intrinsic motivators, where the intrinsic motivators are what motivates you to do something because you like it and extrinsic motivation is when something external rewards the developer. To test this further research is necessary where a more robust introduction and goals are set for the individual where the intrinsic motivation could be more satisfied. Badges being preferred is conclusive with the work in [7] [8] but conflicts with the work in [14]. However, in the work by they state that they believe that they made the badges too hard to achieve which could explain the difference. Since this research did not include leaderboards i would encourage researchers that would replicate this study to add a leaderboard element as a gamification feature since points by it self was not considered to be especially good by its own according to the subjects.

Some of the subjects who found few bugs but achieved high path coverage stated that they felt that they could just use the gamification features to be rewarded even though they did not actually provide any value. We consider that for future research modifying the gamification features for path coverage so that you had to find some bugs to achieve the path coverage badges, that could be hard to implement in a real project since you would not know how many bugs there are of if any bugs even exist. With this in mind, path coverage should ether be improved or be considered to be omitted in future research. Our null hypothesis for motivation is H0*motivation* stating that there is no significant difference between the experiment group and the control group in motivation. As stated in the result section we found a statistically significant difference and rejected the null hypothesis H0*motivation*. There is data that could contradict this in table II. The fact that the experiment group was more motivated than the control group aligns with the work of [7] [8] [14] that shows that gamificaiton has

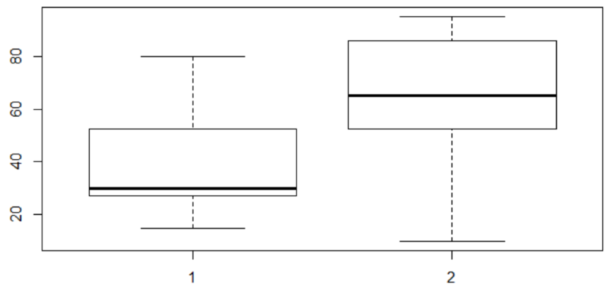


Fig. 9. Boxplot of both the sample groups compared to each other. Where the y-axis is in %, 1 is the control group and 2 is the experimental group.

beneficial effects when i comes to motivation. However, that the experiment group was not statistically significant more motivated after performing the experiment than before conflicts with the work in [14]. The conflict could have its grounds in that their research was conducted over time. The implications for academia at this point is still unclear and further research would be required where replications of studies are performed and conducted over time. It would be safe to use gamification features in industry since many studies have shown statistically significant benefits and our research have shown significance to some extent and also inclinations to the benefits.

We can also show inclinations that most of the experiment group felt motivated to write more unit tests. To understand the motivation of developers in relation to unit tests and gamification features more research is needed with more subjects and potentially other types of features and reward systems.

Since our research only took place during a limited time period the findings are inconclusive with regards to long term exposure to gamification. We can see tendencies towards higher motivation for short time exposure to gamification, however, we do not know the effect gamification have over time. Further research is needed to see how gamification would affect developers in a project over an extended period of time.

C. Quality of unit tests

Our results on the quality of unit tests are based on the quantitative data received from Eclipse. We determined quality by whether the test found a bug or not. For future research this definition could be iterated since a good unit test could be a test that protects against future bugs. This could be examined by research with gamification during an extended period of time where the developers would receive points for when a precautional test finds a bug. These findings are inconclusive with [14] since they state that they did not improve the quality of unit test but rather the number of tests. This could be a result of us giving points based on bugs resolved. This could be a recommended direction for further research or for implementations in companies. Solving bugs should be a higher priority than just findings bugs so it is a natural prioritization. However, this would not work in companies where there are dedicated testers who only find bugs and then send issues.

There was a statistically significant difference for the hypothesis H0*numBugs there is no significant difference in numBugs between the control group and the experiment group* so we could reject the null hypothesis. Since the subjects came from the same education and had approximately the same level of experience in development we believe that the motivation provided by the gamification features is what caused the tendencies towards higher number of *numBugs* found. However, we cannot exclude the possibility that the experiment group had their own interest in gamification and wanted to perform better to prove the usefulness of gamification, if that was the case then motivation would still be the cause of tendencies towards better results. That would still mean that the motivation would improve the results, just not from the intended gamification features.

There was no statistical difference for the null hypothesis H0*pathCoverage there is not significant difference between the control group and the experiment group*. So we could not reject the null hypothesis. This is opposite of the results from the research in [14] where they found the quality of unit tests to be pretty much the same but path coverage and number of tests written increasing. This might differ since our research also showed inclinations towards higher path coverage, but when running statistics we could not see a significant difference. Statistic analysis was not performed [14] which might be a reason for the difference. The data we received was very compatible for statistics and we would recommended future researchers to perform statistical analysis, especially from the quantitative data.

From the discussed results and the answered hypotheses we can answer the research question: **RQ1:** *Does gamification increase the motivation of developers and improve the quality of unit tests?*. Based on our results gamification does increase the motivation of developers and it did improve the quality of unit tests in terms of number of bugs found, but not in terms of path coverage.

VI. Conclusion

In this paper, we researched the effect gamification has on motivation to write unit tests and if the quality of the unit tests could be increased by gamification. We used an existing Java project downloaded from Github as a base in which we introduced bugs. The gamification features were integrated into Eclipse through ANT which generated a report with the subjects' score based on their test results. To test the impact of the gamification features we conducted an experiment with 14 participants comparing the writing of unit tests with and without the gamification features. The results did show a statistically significant difference between the control and the experiment for group higher levels of motivation and number of bugs found. However, there was no statistically significant difference shown for percentage path coverage in the group using the gamification features. The results also showed that the subjects preferred badges rather than points as a gamification feature and number of bugs found rather than path coverage for measurement.

For future research, we see a number of possible areas to dive in deeper into. The most prominent area is to see how gamification affects the motivation of developers over time. It is still unclear after our experiment whether the positive motivation effects gamification had would linger over time. The experiment should only serve as an initial step in the area to give a general indication of the potential effect gamification could have, it is necessary to commit to further research to fully understand the potential of gamification over time. Finally, our research was taken out of context from the developers, it would be necessary to test the effect of gamification inside a real project in the industry where the developers act with the gamification features added to their every-day tools.

References

[1] Z. Yao, Y. Jia, D. Wang, C. Steed, and S. Atchley, "In situ data infrastructure for scientific unit testing platform," *Procedia Computer Science*, vol. 80, pp. 587–598, 2016.

[2] P. J. Clarke, D. Davis, T. M. King, J. Pava, and E. L. Jones, "Integrating testing into software engineering courses supported by a collaborative learning environment," *ACM Transactions on Computing Education (TOCE)*, vol. 14, no. 3, pp. 1–33, 2014.

[3] P. Godefroid, P. de Halleux, A. Nori, S. Rajamani, W. Schulte, N. Tillmann, and M. Levin, "Automating software testing using program analysis," *IEEE Software*, vol. 25, no. 5, pp. 30–37, 2008. [Online]. Available: http://search.proquest.com/docview/215838022/

[4] F. García, O. Pedreira, M. Piattini, A. Cerdeira-Pena, and M. Penabad, "A framework for gamification in software engineering," *The Journal of Systems & Software*, vol. 132, pp. 21–23, 2017.

[5] G. Bavota, A. Qusef, R. Oliveto, A. De Lucia, and D. Binkley, "An empirical analysis of the distribution of unit test smells and their impact on software maintenance." IEEE, 2012, pp. 56–65.

[6] *IEEE Standard for Software Unit Testing (1008-1987)*. USA: IEEE, 1986.

[7] V. Platonova and S. Bērziša, "Gamification in software development projects," *Information Technology and Management Science*, vol. 20, no. 1.

[8] O. Pedreira, F. García, N. Brisaboa, and M. Piattini, "Gamification in software engineering – a systematic mapping," *Information and Software Technology*, vol. 57, no. 1, pp. 157–168, 2015.

[9] D. Charles, T. Charles, M. McNeill, D. Bustard, and M. Black, "Game-based feedback for educational multi-user virtual environments.(report)," *British Journal of Educational Technology*, vol. 42, no. 4, 2011.

[10] D. Kappen and L. Nacke, "The kaleidoscope of effective gamification: Deconstructing gamification in business applications," in *ACM International Conference Proceeding Series*. Association for Computing Machinery, 2013, pp. 119–122.

[11] M. R. d. A. Souza, K. F. Constantino, L. F. Veado, and E. M. L. Figueiredo, "Gamification in software engineering education: An empirical study," vol. 2017-. IEEE, 2017, pp. 276–284.

[12] K. Mäntylä, Mika V. Smolander, "Gamification of software testing - an mlr." Springer International Publishing, 2016, pp. 611–614.

[13] G. Fraser, "Gamification of software testing." IEEE, 2017.

[14] D. Arnarsson and i. H. Johannesson, "Improving unit testing practices with the use of gamification." Institutionen för data- och informationsteknik (Chalmers), Chalmers tekniska högskola, 2015, 94.

[15] J. Whittaker, "What is software testing? and why is it so hard?" *Software, IEEE*, vol. 17, no. 1, pp. 70–79, 2000.

[16] L. Bulej, T. Bureš, V. Horký, J. Kotrč, L. Marek, T. Trojánek, and P. Tůma, "Unit testing performance with stochastic performance logic," *Automated Software Engineering*, vol. 24, no. 1, pp. 139–187, 2017.

[17] A. Qusef, G. Bavota, R. Oliveto, A. De Lucia, and D. Binkley, "Scotch: Test-to-code traceability using slicing and conceptual coupling." IEEE Publishing, 2011, pp. 63–72.

[18] S. Deterding, K. O'Hara, M. Sicart, D. Dixon, and L. Nacke, "Gamification: Using game design elements in non-gaming contexts," in *Conference on Human Factors in Computing Systems - Proceedings*, 2011, pp. 2425–2428.

[19] N. Juristo and A. M. Moreno, *Basics of software engineering experimentation*. Springer Science & Business Media, 2013.

[20] "Apache ANT apache ant documentation," https://ant.apache.org/, accessed: 2018-11-05.

[21] D. Falessi, N. Juristo, C. Wohlin, B. Turhan, J. Münch, A. Jedlitschka, and M. Oivo, "Empirical software engineering experts on the use of students and professionals in experiments," *Empirical Software Engineering*, vol. 23, no. 1, pp. 452–489, 2018.

[22] "Mann Whitney U Test wilcoxon rank sum test," https://ant.apache.org/http://sphweb.bumc.bu.edu/otlt/mph-modules/bs/bs704_nonparametric/BS704_Nonparametric4.html, accessed: 2019-01-20.

[23] A. M. Robert Feldt, "Validity threats in empirical software engineering researchan initial survey," pp. 374–379, 2001.