Industrial Study on Test Driven Development: Challenges and Experience

Adnan Čaušević
School of Innovation, Design and Engineering
Mälardalen University
Västerås, Sweden
adnan.causевич@mdh.se

Rakesh Shukla
Infosys Labs
Bangalore, India
rakesh_shukla@infosys.com

Sasikumar Punnekkat
School of Innovation, Design and Engineering
Mälardalen University
Västerås, Sweden
sasikumar.punnekkat@mdh.se

Abstract—Conducting empirical studies in industry always presents a major challenge for many researchers. Being a graduate student does not make things any easier. Often due to the lack of experience, credibility or just very limited networking, graduate students do not receive many opportunities to directly collaborate with industry and experiment their theoretical models in a realistic environment. On the other hand, empirical research conducted in an academic settings is often criticised for using students as subjects and working with a small sample size, thus creating major validity threat of the published results.

In this paper we are presenting an experience report from an industrial empirical study conducted at Infosys Ltd., India with the support of their global internship program for graduate students, InStep. Focus of the paper is to present several challenges arisen before, during, and after the study, requiring an immediate attention in order to have a successful experiment completion. We also discuss and elaborate the data analysis results and its implication to our current research activities.

Index Terms—Test Driven Development, Industrial experiment, Test Efficiency

I. BACKGROUND

During the last few years, we have been conducting research on analysing the quality of testing within the contemporary development paradigms such as agile, test driven development, etc. by performing several empirical studies in the form of controlled experiments. While analysing the data from one of these studies, an academic experiment [1] designed to investigate the relation between developer’s testing ability and the extent to which test driven development practice is used currently, we noticed that participants wrote significantly less negative test cases compared to the overall number of tests they submitted. By a negative test case, we refer to a test case that was created for the purpose of exercising a program in a way that was not explicitly specified in the requirement. In literature, the phenomenon of writing less negative tests is known as a positive test bias [2], [3] where testing is done using more positive or specification defined inputs. This finding was even more interesting due to the fundamental nature of how test driven development is performed. Test driven development (TDD) [4] requires from developers to construct automated unit tests in the form of assertions to define code requirements before writing the code itself. In this process, developers evolve the systems through cycles of test, development and refactoring. Essentially, by writing test cases before the code, developers use tests to guide them in the correct implementation of the required functionality, and thus potentially create a threat of an inherent positive test bias within TDD.

To confirm existence of a positive test bias in TDD, we have designed an empirical study and performed a pilot experiment in an academic settings [5]. This experiment was designed in such a way to allow researchers to easily distinguish between positive and negative test cases once they are submitted by the experiment participants. Also, this experiment enforced the usage of the same programing interface among the participants, allowing us to execute test cases of one individual participant on the code of all other experiment participants. This was important since test cases, created by developers using TDD, are essentially used as a safety net of the implemented functionality. Developer’s test cases can detect eventual refactoring issues in the current software, but they are not as efficient in finding defects in their own implementation. In order to realistically measure the quality of developer testing we need to essentially gain access to an ideal test suite which is capable of finding all the defects in any given implementation. Our approach here was to approximate such an ideal test suite by merging all the test suites created by several individual developers working on the same problem. Given such a set of multiple implementations and associated test suites, we were able to cross-compare the ability of test cases to find defects. More importantly, by being able to distinguish between negative and positive test cases, we could as well compare defect detecting ability of both negative and positive test cases.

This approach was used in our academic experiment with fourteen master level students randomly assigned in two groups (test-first and test-last). They were given a task to develop a solution for Bowling Game Score problem and in total they created 127 positive and 58 negative test cases. When comparing defect detecting ability of the provided test cases, positive tests were able to detect 226 errors, while negative tests revealed 348 errors. Even though there were around 50% less negative test cases, they could detect 50% more errors than positive ones. This means that negative test cases (31%) were contributing as much as 61% to the overall defect detecting ability of all the tests in this experiment.
Validity threats of our academic study do not come as a surprise. We had a rather small number of participants and we used master level students as experiment subjects. Some might argue that inexperience of our students could be the single main cause of finding a positive test bias effect in our study results. However, this study served for the purpose of gaining more understanding of the positive test bias effect and what benefit developers might have by implementing greater number of negative test cases. Our intention was not to draw any conclusions from our academic study, but rather collect more insights into the identified problem.

A larger industrial-scale study was needed to confirm if this effect actually exists currently in industry and how it could be potentially addressed. It is evident that graduate students do not receive many opportunities to directly collaborate with the industry, often due to the lack of experience, credibility or just very limited networking opportunities. But, in May 2012 an opportunity had arisen for the first author of this paper to apply for an internship at Infosys Ltd.\(^1\), India using their InStep\(^2\) global internship program. Infosys internship program provided us with an opportunity to perform a large scale empirical study, helping us replicate our previous experiment in an industrial settings with their employees as subjects. With this industrial study our aim was to confirm eventual existence of positive test bias in an industrial context, and measure if the quality of negative test cases is the same as that of a positive test cases, once they are all created by professional developers.

II. Preparation for the Study

Upon receiving a positive feedback from the InStep internship program, the first author was introduced to a person internally referred as a Project Mentor. Project Mentor is an Infosys employee responsible for all the technical aspects of the work conducted by the student during the internship period and it is in her/his sole responsibility to define an acceptance criteria of the internship outcome results. In our case, the second author, Rakesh Shukla from Infosys Labs department was assigned as a Project Mentor for this internship.

We presented our recent research activities to Project Mentor, with the main focus on our last academic experiment and the idea to fully replicate it in a controlled environment where at least 100 Infosys employees would participate. However, initial discussions with the Project Mentor revealed that this may not be feasible for the following reasons:

1) It represented a great organisational challenge for many managers to plan ahead, the schedule of their employees, who could participate in the experiment on a specific day almost 4 months in advance. A more flexible estimate, which will not fix a specific day but rather some day in a specific week, was more likely to be an outcome of negotiations with managers.

2) Infosys has a very huge training facility that could easily accommodate 100+ developers working at the same time on the experiment task, helping us to achieve a highly controlled environment for the experiment. Having a controlled experiment only with new recruits may not satisfy the fundamental goal of evaluating research finding in an industrial settings. Moreover, since this training facility is located in the city of Mysore, India, where no actual production development work is done, bringing employees from other development centres (DC) in India to Mysore was not a feasible option either.

These reasons required to modify the original experiment design in order to still have a high number of participants in the study. A joint decision was made to conduct a semi-open experiment where participants could access and complete the experiment task at any time or date they find it suitable, as long as it is within the internship period and stay in India of the first author. Also, due to the distributed environment in which Infosys employees are working, developers will have to conduct the experiment task at their own workplace using their own equipment. However, a set of tools and instructions would be provided to experiment participants by the researcher to avoid using different programming languages and thus create unusable data for the analysis.

In order to seamlessly distribute those tools and instructions to the experiment participants, a dedicated server had to be setup within an Infosys internal network infrastructure. Project Mentor assigned one person (Shruti Bansal) to aid the researcher with this task before his arrival to Infosys. Our plan was to use this server as well for a version control system (SVN) as it was used in our academic experiment. But, this would require that Infosys employees should have an SVN client software installed on their own office computers, which is not an easy task for many employees due to the very strict internal policies in place. Since we planned to use Eclipse [6] development environment for solving the experiment task, we noted after some researching that Eclipse has an internal history system which could be used for the same purposes as SVN. The only requirement is that experiment participants submit a full Eclipse workspace for the analysis and not just a project folder.

Once the infrastructure problems were sorted out, the following issue was how to provide sufficient training to some employees who are not familiar with test driven development practice or the usage of JUnit [7] testing framework. This problem was mitigated by the decision to create a set of video tutorials by the researcher, explaining main concepts of the Eclipse, JUnit and TDD through theory and examples.

At the same time, while the researcher was sorting out infrastructure issues and working on the creation of video tutorials, the Project Mentor started an internal promotion of the experiment through various channels of communication. Managers, developers and testers from several divisions were individually contacted by phone or email, and a general email invitation was sent out to various focus groups mailing lists. By the time the researcher arrived in Bangalore, India to start his internship, a list of around 100 email addresses of Infosys employees, willing to participate in the experiment,
was collected by the Project Mentor. Once this effort was completed, the actual experiment execution phase could take place.

III. STUDY EXECUTION PHASE

The first author of this paper stayed at Infosys development center in Bangalore, India, during September 2012. During the first week of internship it was decided that the experiment will be executed from the 10th till 21st of September, 2012. The first week was used to finalise the training video material, instructions and survey questions for the experiment. Detailed information about the experiment task, instructions and video tutorials provided to experiment participants are provided on the first author’s webpage.

The 100+ Infosys employees, who indicated their willingness to participate in the experiment to the Project Mentor, were randomly distributed into the following three groups:

- "Test-last" (TL) group with 36 participants
- "Test-driven development" (TDD) group with 36 participants
- "Test-driven development with the support of negative testing" (TDD+) group with 35 participants

The third group of participants in this experiment (TDD+ group) represents a new dimension to the original academic experiment design. With this group of participants, we aimed at investigating if the effect of positive test bias could be leveraged in the test driven development process by directly informing participants to occasionally add a negative test case. Basically, they were instructed to follow TDD, but also to occasionally write a negative test case by not using an explicitly stated requirement. Design of such a test case would be based on a defined input space, domain knowledge, etc. Our goal was not to distract developers with the traditional usage of TDD but rather to test if the lack of negative tests could be overcome by a simple reminder to developers.

Participants were informed that their enrolment in this experiment would support the current research activities within Infosys Labs, but the exact details of the experiment, as well as the goal of the experiment, were not shared with them. Additionally, participants were explained that their source code and test cases would be analysed anonymously and results of this activity will not be used in any way for the internal employee evaluation.

For each participant, a dedicated Microsoft SharePoint Workspace was created with the training material and instructions placed in it. Decision to use SharePoint was made since it is the de facto standard at Infosys for synchronising documents between computers. Our attempts to propose a network share folder or even a dedicated intranet website for the experiment resulted in rejection from an internal communication & computer division. As a result, researcher had to send around 100 personal emails to experiment participants, each containing a private link for accessing a SharePoint workspace. This is where our first problems with the experiment occurred, since it was not possible to create more than 64 shared workspaces on the server. This required to immediately find some alternative ways to provide the necessary material to the experiment participants.

The first alternative was to obtain files from Internet, hosted at researchers home university website. In particular, the participants has to obtain Eclipse as well as an experiment instruction document. Video tutorials were also hosted outside the Infosys intranet and a dedicated link was provided to participants depending on the experiment group they belong to. However, not all participants had the ability to connect to Internet, due to the internal policy. In those particular cases, where both SharePoint and Internet access problem existed, a Microsoft Office Communicator (instant messaging software at Infosys) was used to transfer the required files.

Upon finishing their development task, participants uploaded software solution to their dedicated SharePoint Workspace. Participants, who did not have a dedicated SharePoint Workspace, sent their solutions by email to the researcher.

IV. ADDITIONAL DETAILS OF THE EXPERIMENT

The design of this experiment originates from the academic study, elaborated in [5], but modified to adapt to the industrial context in which it was executed. Main goal of the experiment was to investigate if the effect of a positive test bias could be identified when working with professional developers, regardless if the test last or test first approach was used. We wanted to be able to examine the existence of such an effect by calculating the effectiveness of the provided tests. As an additional goal, in case the positive test bias effect existed, we wanted to investigate if it is possible to eliminate such an effect by providing participants with the support for the negative tests.

Participants of this experiment were Infosys employees spread around several development centres in India and even some employees participating from the Infosys client’s (on-site) locations. The set of participant locations include: Bangalore, Beaverton, Brussels, Chennai, Hyderabad, Mangalore, Melbourne, Mysore, Pune, Trivandrum.

The Bowling Game Score Calculator problem was used for the experiment. The specification was based on the Bowling Game Kata (i.e., the problem also used by Kollanus and Isomöttönen to explain TDD [8]). From our experience, on average, 3 hours are needed to fully implement the problem and usually around 10 test cases are created during implementation when following TDD approach.

Participants assigned to the TDD group were instructed to use TDD steps to develop software solution. Instructions for TDD were given as prescribed by Flohr and Schneider [9]. Participants assigned to the TDD+ group were instructed to use the same TDD steps with the addition to the very first step. Basically, they were instructed to follow TDD, but also to occasionally write a negative test case based on the input space, domain knowledge, etc. (but not explicitly stated in the requirements). Participants assigned to the Test Last (TL)
group were instructed to use traditional (test-last) approach for software development and they were considered as a control group for this experiment.

V. DATA ANALYSIS PHASE

Due to the limited period of the internship at Infosys, a full data analysis of the experiment was performed at the researcher home university. Table I presents the number of solutions submitted for the experiment analysis by the participants of the experiment.

<table>
<thead>
<tr>
<th>Group</th>
<th>Submitted</th>
<th>Removed</th>
<th>Analysed</th>
</tr>
</thead>
<tbody>
<tr>
<td>TL</td>
<td>19</td>
<td>8</td>
<td>11</td>
</tr>
<tr>
<td>TDD</td>
<td>21</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>TDD+</td>
<td>20</td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td>Total</td>
<td>60</td>
<td>27</td>
<td>33</td>
</tr>
</tbody>
</table>

Once the submitted solutions were individually inspected (code was visually reviewed and tests were executed on the accompanied code), several of them had to be removed before the data analysis started.

Reasons for such a removal are listed below:

**Incomplete solutions**
Manually looking at the code it was possible to identify that some provided solutions were not completed.

**Own failing test cases**
Number of solutions had test cases which were failing on their own code. This was usually a sign of an incomplete solution.

**Small number of test cases (≤3)**
In case a submitted solution did not have a minimum of 3 test cases, in addition to the one test case provided with the code skeleton, such solutions would be removed from the analysis.

**Wrong test cases**
It is very important not to have false positives in the test cases. In case a test case is expecting a wrong result, the same was removed from the test suite, but if most of the test cases were wrong, then the complete solution was removed from analysis.

**Different programming interface**
Some solutions used a different programming interface which prevented executing other participants test cases on its code, or executing its test cases on other participants code.

For a few submissions it was also noted to contain a rather small number of test cases, but visual inspection revealed that several tests (assertion statements in jUnit) were grouped into one joint test method. There was no need to remove such submissions, but researcher had to put an additional effort of separating assertions into individual test methods before the data analysis. We did experience the same problem in our previous experiments, but we did not want to instruct our participants on how to write test cases. Our goal was not to have tests that will ease the process of analysis, but tests that are created to make developer feels confident of her/his work.

Around 40 employees did not submit their solution at the end of researcher internship in Infosys. One of the reasons for such a high drop-out was that many of employees were initially assigned to participate in the experiment by their managers, but at the time the experiment was executed, they were not able to contribute. It is interesting to discuss drop-outs with respect to the designation our participants have at Infosys. We noticed that mostly employees with a designation focused on testing were not able to fully complete their solution before the deadline. There are several interpretations of this phenomenon: (i) the actual work duties at Infosys are heavily deadline stressed for testers than for developers, (ii) testers find this empirical study more directed towards developers since TDD is a development practice, or (iii) testers have limited development skills.

Our previous academic experiment [5] was performed with a limited number of participants and although we could see some trends, it was difficult to evaluate statistical significance of the collected data. However, this industrial experiment enabled us to perform hypothesis testing using statistical methods on the data we collected. One of the first thing we wanted to investigate with this experiment is the existence of a positive test bias within our participants, defined as following null hypothesis.

\[ H^1_0 \text{ There is no difference between the total number of positive and negative test cases created by experiment participants.} \]

We used the Wilcoxon signed rank test for paired non-parametric data in order to test the \( H^1_0 \) null hypothesis with \( \alpha = 0.05 \). With a \textit{p-value} of 0.00000731 we could reject the null hypothesis and confirm that a difference between the created number of positive and negative test cases is significantly different for our industrial participants.

In addition, we wanted to compare defect detecting ability of both positive and negative test cases. By doing that we can explore further how lack of negative tests could potentially affect the overall testing effectiveness. The following null hypothesis was defined for that purpose:

\[ H^2_0 \text{ There is no difference between the number of failing assertions detected by positive and negative test cases.} \]

Again, the Wilcoxon signed rank test for paired non-parametric data was used in order to test the \( H^2_0 \) null hypotheses with \( \alpha = 0.05 \). We can reject the stated hypothesis since the \textit{p-value} is 0.00000302, confirming that there is a significant difference in the efficiency of positive and negative test cases.

When looking at the actual differences in the number of created positive and negative test cases, as shown in Figure 1, we can notice that a very similar 70%-30% ratio exists for all groups individually as well as for all participants test cases combined together.
Several benefits of this study, from an academic perspective, could be listed. The main benefit is, for sure, ability to use professional developers as subjects in the experiment. However, opportunity to have an active collaboration with an industry while being in the industry and to obtain a direct feedback on the industrial impact our research might have, are very high on the list of achieved academic benefits with this experiment. Not having any financial cost associated with this study is definitely worth mentioning. For the purpose of comparison, a study done by Bergersen et al. [10] was executed by hiring professional developers resulting in a cost of €40,000. In addition to free usage of resources, travelling expenses, accommodation, and daily allowances were as well provided to the researcher via InStep internship program.

During the execution of the experiment itself, we gained valuable experiences and lessons for future related studies. Employees are generally busy with their daily and project related duties, which of course does not come as a surprise. Their time is of greatest value and should be fully respected in the context of performing any experimentation. From our experience, if there is any opportunity to spare employees from wasting their time on tasks which are not directly contributing to the experiment itself, this should be done to the highest extent possible. From setting up infrastructure to providing video instructions and tutorials, everything that can save their time is of a great value.

However, we also learnt that our participants are willing to try out new things and are very open minded. This was noted from answers in the survey, which our participants filled after completing the experiment task. In addition to answering predefined demographic questions in the survey, employees had an opportunity to provide a comment on the experiment itself and their experience with it. It was completely unexpected to receive quite many positive comments about the experiment design and its execution. Some employees even suggested that their emails should be placed on a list for any future experiments Infosys Labs might conduct. Still, some employees raised concern that their work for the experiment might be used for their evaluation. We could be even more clear at the beginning of the experiment that all the data we collect will be analysed anonymously and thus remove any concerns with respect to this issue.

In November 2012, researcher conducting this internship was nominated and received a Best Intern award in recognition of outstanding contribution as Intern during 2012-13 term of InStep program.
VII. Conclusion

Based on our academic results from previous experiments as well as results from this industrial study, it is evident that positive test bias (i.e., lack of negative test cases) is present when test driven development approach is being followed. On an average, in our studies, around 70% of test cases were positive while 30% were negative. However, this effect was not only constrained to TDD since test last or traditional developers experienced the same problem as well. When measuring defect detecting ability of test cases, an opposite ratio was observed. Effectiveness of negative test cases were above 70% while positive test cases contributed around 30%. These results made evident what importance negative test cases have as part of a test suite.

Importance of these results could be even more highlighted once we consider the context in which they are investigated. In first study we have a fully controlled experiment performed in an academic environment using students as subjects. In the second one we have a semi-controlled experiment in an industrial settings using skilled developers as subjects. In both cases we can observe nearly the same behaviour.

We consider that many organisations could use industrial internships as a highly valuable and direct communication channel with the academia. By doing this, industry will increase their visibility within the academic community and gain an early-bird access to the latest research results. However, for this to happen, academia has to still work on defining how acceptable and valuable are industrial experiments performed in a semi-controlled settings with the ever growing list of validity threats and low ability of replication.

This empirical study helped to gain more understanding of the effect of lack of negative test cases and its impact on test driven development approach. Currently, our work is focused towards improving a modified model of test-driven development which will facilitate non-specified requirements to a much higher extent. Our aim with this model is to overcome built-in positive test-bias within test-driven development. Another academic and industrial experiment will be needed to confirm benefits of this model.

ACKNOWLEDGMENT

We would like to express our gratitude to all the Infosys employees who devoted their time and effort to participate in this experiment. Additionally, we would like to acknowledge Shruti Bansal for her enormous help in maintaining the infrastructure and InStep team for making the stay for the researcher in India easy and pleasant. Finally, we would like to thank Infosys for offering such a great opportunity for young researchers to cooperate with industry.

REFERENCES