Analyzing the Effectiveness of a System Testing Tool for Software Product Line Engineering

Crescencio Rodrigues Lima Neto\textsuperscript{1,2,3}, Ivan do Carmo Machado\textsuperscript{2,3}, Vinicius Cardoso Garcia\textsuperscript{5}, Eduardo Santana de Almeida\textsuperscript{2,3,4}

\textsuperscript{1}Federal Institute of Bahia
\textsuperscript{2}Computer Science Department - Federal University of Bahia (DCC/UFBA)
\textsuperscript{3}Reuse in Software Engineering Labs (RISE)
\textsuperscript{4}Fraunhofer Project Center (FPC) for Software and Systems Engineering
\textsuperscript{5}Center for Informatics - Federal University of Pernambuco (CIn/UFPE)

\{crescencio, ivanmachado, esa\}@dcc.ufba.br, vcg@cin.ufpe.br

Abstract—This paper reports on a serie of empirical evalua-
tions of a proposed testing tool for SPL, in order to understand
how useful it might be consider. The study is a serie of controlled
experiment involving over forty subjects, which analyzed the
use of a proposed tool in an SPL project. The results show that
the effort spent during the test case design decrease significantly
when the tool was used. The subjects preferred to work with the
tool, especially users without experience in testing. The collected
insights are reported for further studies and the gathered data
will serve as baseline values for future experiments, since there
is a lack of work in this direction.

Keywords—Software Testing; Software Product Lines; Soft-
ware Reuse; Testing Tools; Empirical Evaluation

I. INTRODUCTION

Automated tools are available to support testing in every
stage of the software development life cycle \cite{1}. Some orga-
nizations have used testing tools to manage, store and handle
the tests, including their execution results. Nakagawa et al. \cite{2}
encourage the use of supporting tools to make testing a more
systematic activity, which may lead to reductions in cost and
time consumed, and also minimizing errors caused by human
intervention.

However, when narrowing the observation scope to only
consider testing tools for Software Product Line Engineering,
the number of existing testing tools decrease drastically, lead-
ing to the need of tools to support testing in such a develop-
ment paradigm, so as to enable practitioners to experience the
same benefits, as the ones mentioned earlier.

Within the set of existing tools for SPL testing \cite{3}, either
the reports lack details of empirical assessments that endorse
their usefulness, or there is really few evidence that support
in the large-scale use.

In earlier investigation, we developed SPLMT-TE, a tool to
support the design of system test cases from SPL use cases
\cite{4}, \cite{5}. In this paper, we report on a serie of experimental
studies performed in order to evaluate its usage effectiveness,
namely an in-vitro experiment, and its replication, in a distinct
environment.

The experimental study consisted of three phases. We
initially performed a pilot study \cite{6}, aiming at analyzing
the experiment feasibility. Then, we performed the actual
experiment. Next, we carried out the replications.

The experimental study was conceived and structured based
on the concepts of experimental software engineering and the
evaluation of methods and tools provided by Wohlin et al. \cite{7}.
The replications were motivated by the desire of achieving
more significance and confidence on the results \cite{8}.

The remainder of this paper is structured as follows. Section
II introduces the proposed testing tool. Section III presents
the experimental study. Section IV discusses the related work.
Section V concludes the paper and presents future research.

II. TOOL SUPPORT FOR SYSTEM TESTING IN SPL

In previous research \cite{3}, we carried out a mapping study of
tools for testing software product lines, involving the investiga-
tion of thirty-three research papers, dated from 1999 to 2011.
Among other goals, such a mapping aimed at identifying the
functionalities supported by existing tools, and also to leverage
others an SPL testing tool is expected to support. The findings
of such an investigation served as input for the analysis and
design of our proposal, called SPLMT-TE \cite{4}.

The SPLMT-TE was developed using Django, a Python Web
Framework\cite{1}. The tool supports the design of system test cases
from use case, and manages assets such as test cases, suites,
and plans. It intends to reduce the effort and the cost associated
with the SPL testing process. We performed an initial evaluation
in \cite{5} to assess the tool’s effectiveness.

III. EXPERIMENTAL STUDY

This Section details the experimental study setup and mea-
urements used to investigate the effects of using our proposed
tool. We reported the study by following the guidelines for
conducting experiments in software engineering defined by
Wohlin et al. \cite{7}, and reported as suggested by the guidelines
described in \cite{9}.

1http://www.djangoproject.com/
A. Experimental Planning

We applied the GQM method [10] in order to define quantitative measures that could provide objective assessment of the tool’s effectiveness. It was structured as follows:

1) Goal.: The objective of this experimental study is to analyze the SPLMT-TE tool for the purpose of evaluation with respect to its efficiency and effectiveness from the point of view of the potential users (testers) in the context of an SPL testing project in an academic environment.

2) Questions.: To achieve this goal, we used the following questions defined in [5] and we added the question Q6:

- Q1. Is the amount of test cases increased when the tool is used? [5]
- Q2. Is the time required to design system test cases reduced when the tool is used? [5]
- Q3. Is the time required to execute the designed test cases reduced when the tool is used? [5]
- Q4. Is the amount of errors detected increased when the tool is used? [5]
- Q5. Is the effectiveness of test cases improved when the tool is used? [5]
- Q6. Is the time required to find errors improved when the tool is used?

3) Metrics.: After defining the questions, they need to be mapped to a measurement value, in order to characterize and manipulate the attributes in a formal way [11]. Hence, the metrics used in this analysis are next described next:

- M1. Designed Test Cases (DTC)[12], [5];
- M2. Efficiency in Test Case Design (ETCD)[12], [5];
- M3. Efficiency in Test Cases Execution (ETCE)[12], [5];
- M4. Number of Errors Found (NEF) [12], [5];
- M5. Test Cases Effectiveness (TCE) [13], [5];
- M6. Efficiency in Finding Errors (EFE): It refers to number of errors found (M4) reported over the amount of time spent to execute test cases (TSE).

\[
EFE = \frac{NEF}{TSE}
\]

4) Design.: In this experimental study, we applied One factor with two treatments, as the design type. We compared the two treatments against each other [7]. In this sense, the factor was the use of the proposed tool, and the treatments were: (1) Creating test cases with the tool support and (2) Create test cases manually. The subjects were divided in two groups, each one addressing a treatment.

5) Experiment materials.: The experiment used Consent Form, Background and Feedback Questionnaires, a set of Test Assets, Use Cases, Test Case Creation Form, Test Case Execution Form, Defect Reporting Form, application to be tested and the proposed tool.

6) Subjects.: We applied convenience sampling [7] in this experimental study. Initially, we performed a pilot with fourteen students from the Software Engineering course at Federal University of Bahia, Brazil. Next, the subjects were graduate students (seven M.Sc. Students and five Ph.D. Students) from Federal University of Bahia and Federal University of Pernambuco, Brazil. Then, additional nineteen students from the Software Engineering course at Federal University of Bahia participated in the experiment replications.

7) Hypotheses.: We set up six hypotheses for this experimental study. The Null Hypothesis \((H_0)\) considered that there was no benefit of using the proposed tool.

Conversely, the Alternative Hypothesis \((H_1)\) stated the opposite values. The alternative hypothesis determined that the proposed tool produced benefits that justify its use.

We used the same hypotheses defined in [5]. Moreover, we added the Null Hypothesis \(H_{06} : \mu_{EFE_{manual}} = \mu_{EFE_{tool}}\) and the Alternative \(H_{16} : \mu_{EFE_{manual}} \neq \mu_{EFE_{tool}}\).

8) The Experimental Study Project.: We selected two projects for the experimental study. The first one consisted of a product line in the mobile devices games domain called Arcade Game Maker (AGM) Pedagogical Product Line\(^2\) produced by the SEI\(^3\). The second project consisted of the NotepadSPL, a product line built upon the well-known word processor Notepad. This product line was extracted from the FeatureVisu\(^4\).

B. Execution

The subjects carried out the activities involved in the experiment. When the subjects filled out the background questionnaire, we could organize them into two groups. The balancing strategy was applied to harmonize the groups in terms of expertise. Both groups performed the experiment using the tool in one phase and without using the tool in the other phase.

1) Procedure.: The experiment took place at Federal University of Bahia, Brazil in July of 2011. The replications were also carried out at the same place, namely in November 2011, and in June 2012.

Initially the subjects became aware of the experiment purpose and associated tasks. Next, training sessions were performed with practical exercises.

The first training session consisted of theoretical and practical classes on fundamentals of software testing. Both groups learned how to create a good test case, how to execute test cases, and how to report errors. It was not necessary to offer training classes on software product line engineering, given that all subjects had been involved in research in such a field.

Next training session involved the use of the SPLMT-TE tool. At the end, they had to fill out the feedback questionnaire. This process was also performed at the pilot study and replications execution.

2) Operation.: All subjects were asked to fill out a background questionnaire, as the initial task in the experiment. It served for us to investigate their experience. It was comprised of a set of items, described next:

- Participation in industrial development/testing

\(^2\)http://www.sei.cmu.edu/productlines/ppl/index.html
\(^3\)http://www.sei.cmu.edu
\(^4\)http://fosd.de/FeatureVisu
- Experience in programming
- Experience with testing tools

The execution was performed in two phases. One group used the tool while the other worked without tool support. The pilot study execution was limited to 2 hours of test cases creation and execution because of the discipline schedule. However, at the experiment and replications, we did not restrict the time of creation and execution.

The subjects were instructed to analyze the available components, create, and execute the test cases. Their assigned tasks were: (1) to analyze the available use case document; (2) to build test, i.e., writing the test cases manually or using the proposed tool; (3) to execute them manually step by step; and (4) to report the findings in the proper form. The replications focused only at the last two steps: test execution and reports.

At the end of the experiment, each participant completed the feedback questionnaires. The complete characterization of the subjects, as well as supplemental material of this experimental study such as complete descriptive statistics, questionnaires, reports, and forms can be reached in the paper’s resource website.

C. Analysis and Interpretation

In [5], we used the wrong hypothesis testing in some descriptive statistic analysis. For this reason, we analyzed all the artifacts produced by the subjects again, including the error report forms and the feedback questionnaires. The new analysis was performed based on descriptive statistics and hypothesis testing using t-test [7] (for parametric statistical hypothesis test) and Mann-Whitney-Wilcoxon [7] (for non-parametric statistical hypothesis test).

Since the replications’ subjects had the same profile, undergraduate students from a Software Engineering course inexperienced in testing, the two replications data were combined and analyzed in the same dataset.

Descriptive Statistic

1) Designed Test Cases.: According to Figure 1(a), the amount of designed test cases created with the tool is higher than without tool support. Moreover, since the subjects were inexperienced in testing, we decided not to consider this measure in the replications.

2) Efficiency in Test Case Design.: Figure 1(b) presents that the efficiency in test case design was higher with groups that used the tool, which enabled the creation of more test cases faster than without tool support. Hence, as this metric is related with the previous metric, we preferred not to consider it for the same reason.

3) Efficiency in Test Cases Execution.: Figure 1(c) presents that during the first experiment execution, the effort of test case execution were almost the same. On the other hand, during the replications, the tests created by the tool were more efficient than the tests created manually (see Figure2(a)).

4) Number of Errors Found.: Figure 1(d) presents that the subjects were capable to find more errors during the use of the tool. Figure 2(b) confirms that the number of errors found were higher when the subjects used the tests created by the tool.

5) Test Cases Effectiveness.: Figure 1(e) presents that during the experiment, the test case effectiveness was practically the same using the tool or not. The number of errors found per number of test cases was almost the same using and without using the tool. Which can be confirmed by the replications and it can be seen in Figure 2(c).

6) Efficiency in Finding Errors.: Figure 1(f) presents the amount of errors that were found in the experiment execution. The efficiency was practically the same between the amount of errors found using the tests generated by the tool and manually. On the other hand, during the replications more errors were found faster when the tests created by the tool were used (see Figure 2(d)).

Hypotheses Testing

Since the experiment has one factor with two treatments, completely randomized design, the data collected during the experiment were submitted to descriptive statistic examination. Firstly, in order to reduce the dataset errors, we eliminated the outliers. Secondly, we verified if the sample came from a normally distributed population through Shapiro-Wilk test.

Thirdly, when the sample was not “normal”, we used the non-parametric test (Mann-Whitney-Wilcoxon). On the other hand, when the sample was “normal”, we verified if it had equal variance through Levene test. Finally, when the sample had different variance, we used non-parametric test (Mann-Whitney-Wilcoxon), and, when the sample had equal variance we used parametric test (t-test).

The tests were primarily presented for a significance level of 5%. Table I presents the hypothesis testing results from the experiment and replications. The results are detailed next.

![Table I: Hypotheses Testing Results](http://www.crescenciolima.com/seke2013/)

Regarding Q1, the amount of test cases created using the tool is higher than without using it, $p-value = 0.0007$. The $p-value$ is smaller than the significance level, rejecting the null hypothesis. **Null Hypothesis $H_{01}$ is rejected, since $H_{01} : \mu_{DTC_{Manual}} \neq \mu_{DTC_{Tool}}$.**
Time spent creating test cases without the tool is higher than using it. For this reason, in order to answer the Q2, the Null Hypothesis $H_{02}$ is rejected, since $H_{02} : \mu_{ETCD_{manual}} \neq \mu_{ETCD_{tool}}$, $p$-value = 0.0001. This $p$-value allowed the rejection with high significance level.

Thus, the efficiency in test case execution supports the Null Hypothesis $H_{03}$. The question Q3 is answered, $p$-value = 0.03. As the $p$-value is lower than 0.05 the null hypothesis was rejected. The same result is confirmed in the replications, $p$-value = 0.04.

The Null Hypothesis $H_{04}$ cannot be rejected, since there is no significant difference $H_{04} : \mu_{NEF_{manual}} = \mu_{NEF_{tool}}$, although the number of errors found were higher using the tool support. The experiment $p$-value = 0.35 and the replication $p$-value = 0.058, are higher than 5%, which did not reject the null hypothesis, answering the Q4.

As a result, the Null Hypothesis $H_{05}$ cannot be rejected, since $H_{05} : \mu_{TCE_{manual}} = \mu_{TCE_{tool}}$, the experiment $p$-value = 0.34 and replication $p$-value = 0.5. Since the $p$-value are higher than the significance level the hypothesis cannot be rejected and no conclusion can be drawn. Regarding Q5, there is no significant differences between the effectiveness test cases values during the tool usage and without use it.

Finally, regarding the Q6, the efficiency in finding errors supports the Null Hypothesis $H_{06}$. The experiment $p$-value = 0.7, cannot reject the null hypothesis. On the other hand, the replications $p$-value = 0.03 is lower than the significance level, rejecting the Null Hypothesis.

D. Evaluation of results and implications

The results presented insights that enable us to consider that activities of test case creation were more productive when the tool was used as we can seen in the hypotheses $H_{01}$ and $H_{02}$. For this reason, we focused the replication analysis in the null hypotheses that could not be rejected.

During the replications, we identified that in some aspects, subjects without experience were more productive than the experienced ones. In other words, the tool could have hampered the performance of the subjects with experience in testing. Although, almost all the subjects answered the feedback questionnaire saying they preferred to work with the tool.

During study execution, we observed the importance of splitting the design and execution activities. Ideally, these activities should be performed by different subjects.

Perhaps, as the subjects gain confidence on the SPLMTE usage (depending on the time availability, to try different scenarios), the results might become better. But it is solely an assumption that should be tested, maybe applying in a larger context.
E. Threats to Validity

1) Maturation.: This is the effect that subjects react differently as time passes. Some subjects can be affected negatively during the experiment, and their performance may be below normal. In order to mitigate this boredom, two familiar domains were provided.

2) Gained Experience.: There were two scenarios: in the first phase group 1 performed the task using tool support, and in the second phase, without the tool. In the second scenario, group 2 executed the first phase manually, and in the second phase with tool support. To mitigate this risk, two distinct problems were analyzed.

3) Experimenter Expectations.: Surely the experimenter expectations may bias the results, and for that reason, all formal definition and planning of the experiment was carefully designed by the first author of this paper, and revised by the two remaining authors. We also had support of two senior researchers in the field of Software Product Line Engineering.

4) Generalization of subjects.: This is an effect of having a subject population not representative of the population we would like to generalize to. The replications were conducted with undergraduate students without knowledge about software testing. In this case, if these subjects succeed using the tool support, we cannot conclude that an experienced testing analyst would use it successfully too.

IV. RELATED WORK

As we mentioned before, there is few empirical evaluations of testing tools for SPL. According to [14], there is a lack of studies reporting on empirical evaluations in SPL testing, more specific in a testing tools viewpoint.

It is even harder to find empirical studies that evaluate SPL testing tools. From the papers we analyzed in the previous work [3], we found only one study considered as a similar study [15].

Nogueira et al. [15] presented the Test and Requirements Generation Tool (TaRGeT), which is a tool for automatic test case generation from use case scenarios written in Natural Language (NL).

V. CONCLUDING REMARKS AND FUTURE WORK

This paper presented the experimental studies conducted to evaluate the proposed tool. It included the definition, planning, operation, analysis and interpretation of a pilot study, experiment, and replications. The guidelines defined by [7] were used to perform the experiment.

The evaluations analyzed the SPLMT-TE effectiveness and efficiency, in order to gather empirical evidence. As one of the findings we had after performing this study, we observed that, for users with experience in testing, it could be more complicated to work with the tool support.

Probably, for these subjects, there was no gain using the tool to save time during the execution of test cases. In the analysis of test case effectiveness, it was not possible to reject the null hypothesis, concluding that there is no advantage in using the tool.

On the other hand, the tool proved to raise the productivity of subjects without experience in testing. Moreover, the number of test cases created and the number of errors found were higher when the subjects used the proposed tool. Finally, the effort spent during the test case design decrease when the tool was used.

As future work, we plan to compare the proposed tool with some other testing tools, which is a difficult task since it is necessary to adapt the tool use to SPL. We also intend to replicate this study in other environments.

ACKNOWLEDGMENT

This work was partially supported by the National Institute of Science and Technology for Software Engineering (INES), funded by CNPq and FACEPE, grants 573964/2008-4 and APQ-1044-1.03/10 and APQ-1037-1.03/08 and CNPq grants 305968/2010-6, 559997/2010-8, 474766/2010-1 and FAPESB.

REFERENCES