Testing software product lines

da Mota Silveira Neto, Paulo Anselmo; Runeson, Per; Machado, Ivan do Carmo; de Almeida, Eduardo Santana; de Lemos Meira, Silvio Romero; Engström, Emelie

Published in:
IEEE Software

DOI:
10.1109/MS.2011.90

2011

Software Product Lines Testing and Industrial Perspective

Paulo Anselmo da Mota Silveira Neto[3], Per Runeson[4],
Ivan do Carmo Machado[2,3], Eduardo Santana de Almeida[2,3]
Silvio Romero de Lemos Meira[1,3], and Emelie Engström[4]
[1] Federal University of Pernambuco (UFPE), Brazil
[2] Federal University of Bahia (UFBA), Brazil
[3] Reuse in Software Engineering (RiSE), Brazil
[4] Lund University (LU), Sweden

I. Introduction

Software Product Lines (SPL) has received growing attention for its potential in fostering reuse of software artifacts. Evidence also indicate that they enable organizations to develop applications with less effort, in shorter time, or with higher quality when compared to development of single systems. Nevertheless, all of these benefits do not come for free, and to achieve the improvements promised, a high quality assets intended for reuse is essential. Therefore, quality assurance in general and testing in particular are still the most commonly applied quality assurance technique in industry becoming a crucial part of the product line effort.

The literature on SPL distinguishes between two sets of processes, labeled *domain* and *application* engineering, respectively [7], and deliver two types of software, *core assets* or *platform*, and *product*, respectively [6]. Testing is a fundamental activity applied over the whole SPL life-cycle, which includes testing the core assets software, the product specific software, and their interactions. While the domain testing process aims at ensuring that core assets are working properly, the application testing process determines whether the product being produced is the product specified by the requirements [3]. Several approaches address the interactions between these two processes, for example, developing test artifacts in domain engineering and then reusing these artifacts in application engineering [8].

In order to investigate state-of-the-art testing practices, synthesize available evidence, and identify gaps between required techniques and existing approaches available in the literature, two systematic mapping studies [1], [2] were carried out in parallel, addressing different SPL testing topics. One study was conducted by Engström and Runeson in Sweden and one by Silveira et al in Brazil, below referred to as *study.se* and *study.br*, respectively.

II. The Mapping Studies

Systematic mapping studies aim at giving an overview of a field of research. They are as systematic as systematic literature review (SLR) studies, but are conducted when the field of study is not sufficiently mature to comprise a set of comparable empirical studies. Instead it provides an overview of conducted research [4].

The *study.se* started off broadly, to get an overview identified challenges for SPL testing and the topics already studied. It also aimed at identifying the academic in which the research is published and which type of research is conducted. The search was conducted in an iterative manner, 1) starting with an exploratory search, 2) extending by the “snow-balling” process (following up the references of the papers found), 3) screening main conference proceedings, 4) validating the result through database searches, and 5) validating against a smaller systematic review [5]. This process resulted in 64 papers, published between 2001 and 2008.

The *study.br* was based on topics addressed by previous research on SPL testing and discussions with expert researchers and practitioners. In order to gather information about the topics, a set of nine research questions were established, each addressing a different issue in the SPL testing field.

Three main steps composed the search process, which was conducted in opposite order, compared to *study.se*: 1) an automatic search was performed using different search engines, 2) a manual search, were the most important and relevant conference and journals were visited, and 3) the “snow-balling” process was applied. This process resulted in a set of 45 papers, published between 1998 and 2009. Table I shows the defined research questions for both studies.

A. Industrial Insights

Some points should be highlighted when considering the practitioners perspective. They are described following.

1) Testing Strategy: When testing a SPL the variability and commonality should be considered in the overall testing levels. Based on our findings five different strategies could be suitable to the SPL assessment. Considering a SPL of critical systems, which need to be stressed, for example, in the medical domain, airplane software and so on, the *testing product by product* strategy seems to be perfectly suitable. In this strategy all products are tested independently, increasing the testing reliability. On the other hand, when working in a different domain (not critical), aspects as fast time to market

Study.br
Which testing strategies are adopted by the SPL Testing approaches?
What are the existing static and dynamic analysis techniques applied to the SPL context?
Which testing levels commonly applicable in single-systems development are also used in the SPL approaches?
How do the product line approaches handle regression testing along software product line life cycle?
How do the SPL approaches deal with tests of non-functional requirements?
How do the testing approaches in an SPL organization handle commonality and variability?
How do variant binding times affect SPL testability?
How do the SPL approaches deal with test effort reduction?
Do the approaches define any measures to evaluate the testing activities?
Study.se
Which challenges for testing software product lines have been identifed?
In which fora is research on software product line testing published?
Which topics for testing product lines have been investigated and to what extent?
What types of research are represented and to what extent?

TABLE I
RESEARCH QUESTIONS.

and cost reduction can be important factors to be considered. For example, mobile phone companies in some cases reduce the test coverage in order to ship a product early in the market. It can be used as market strategy, avoiding concurrence. In this context, *incremental testing of product lines*, *opportunistic reuse of test assets*, *design test assets for reuse* and *division of responsibilities* strategies, seems to be suitable.

All aforementioned strategies could be combined, however no evidence was found regarding this aspect, only brief indications of factors, such as: software development process model, languages used, company and team size, delivery time, budget, etc, should be considered, as earlier as possible, i.e. planning phase, in order to decide which strategy or combination is better suitable to a specific context.

2) Testing Levels: Each testing level has its importance regarding the type of fault found. While unit testing is responsible for testing different units which constitutes a software component or classes, the integration testing should be designed to revel faults in the integration of theses different units. In the SPL context, it still valid, however in a more critical way. As the assets are commonly reused among different applications, it is extremely important that the faults should be discovered in earlier levels. An error revealed in the latest phases increase not only the cost of determined product, but the overall SPL since the products share a common base.

It still valid for static analysis, in which the non-executable software portion is validated against the previous defined specification. It is important to identify the errors before it implementation in the code, reducing the maintenance cost over the SPL lifecycle. Although the static analysis techniques often are dismissed as more expensive, in the SPL context, their costs are amortized over multiple products.

3) Variability and Traceability: The variability and also the traceability among the testing artifacts and the required documents used during their composition (for example, requirements and use cases) must be considered when testing different products. The traceability maintenance is important since the assets are always evolving and changing. Thus, these changes and enhancements should be reflected to the overall SPL artifacts, since they are in some sense integrated.

When this traceability is not maintained, some problems may arise, such as: *testing cost growth*, as the requirements and use cases changes are not reflected in the test cases, a certain amount of resources and time should be dedicated to update the test cases; *reduction on product quality*, as the test cases are not properly reflecting the specification, the test could not guarantee the software accordance; and *increasing on change request analysis*, more duplicated or invalid change requests are raised, since the tests are not clearly specified. This way, all the time spent analyzing duplicate or invalid change requests is lost, not only in a specific SPL phase but over the entire lifecycle. All these problems are intensified when dealing with large scale systems.

4) Effort Reduction: Testing is considered the bottleneck in SPL since the cost of testing product lines is becoming more costly than testing single systems. This high cost makes testing an attractive target for improvements, especially by defining test effort reduction strategies, which can have significant impact on profitability and productivity.

This effort reduction can be addressed through reuse of test assets. Asset repositories should be included in a product line project. An initial effort is required to fill the test asset repository but this effort is amortized as soon as the assets are reused. The systematic reuse enables effort reduction by reducing the redundant work avoided when deriving products.

The use of automation testing tools to support testing activities is another way to achieve effort reduction. In this effect, automatic test generation and execution are considered.

5) Test Organization and Process: The test organization and test processes must be adapted to the needs of product line development. The product-line adds another dimension of complexity, which must be handled. Arguments are raised for a mapping between the organizational structure and the product structure, including the division between product and platform.

An approach is to divide product line testing into two distinct instantiations of the V-model. However, this approach has several problems. Testing is product oriented and no efficient techniques for domain testing exist. Second, complete integration and system testing in domain engineering is not feasible, and thirdly, it is hard to decide how much we can depend on domain testing in the application testing.

III. CONCLUSIONS

Both mapping studies conclude that "software product line testing seem to be a 'discussion' topic" [2]. Sixty percent of the primary studies are of *solution proposal* or *conceptual proposal* type. Just a few studies report on experience from real software environments. In this respect, we point out that proposals should be put into real projects in order to evaluate their practical implementation so that it could guarantee, with practical evidences, the real benefit of the proposals.

Regarding effort reduction – the most addressed topic according to the *study.br* findings – some studies present techniques or methods aiming at reducing effort in SPL testing. Most of these studies consider that reduction can be achieved by reusing test artifacts and execution results, and also through test automation. Considering more specifically the SPL aspects, effort can be reduced by taking advantage of the commonality among products.

The importance of defining a testing strategy is pointed out in the studies, but also no validation is presented regarding the combination of those strategies with testing levels. For example, they do not present which testing levels should be performed in each SPL process, *domain engineering* or *application engineering*. Although we have some studies addressing each of this points, no evidence was found regarding the combination of both aspects.

Test strategies are most critical for the *System and acceptance testing* – the most addressed topic according to the *study.se* findings – since testing of units and components are more similar to single system development, while the expected benefits of SPL are much higher at the system level. Model based testing come in different fashion, and to be useful for SPL testing, variability also must me considered in the models.

Although some studies advocate the use of static analysis to reduce the cost of tests, since as earlier you identify an error less effort will be spend to correct it, few studies provide static analysis techniques. The same holds true for non-functional testing, where most of the proposed approaches are dealing only with functional requirements.

Test *organizations* and *processes* must be tailored to product line testing. A mapping between the product and organizational structures are proposed, but this approach also involve several not yet solved problems. Separate processes are needed, but there are no clear evidence on how these should be designed.

The studies highlight several research topics that need further investigation, such as quality attribute testing considering its variations among products, ways to maintain and manage the traceability between development and test artifacts, and the management of variability through the whole development life-cycle. We also would like to again suggest research focus not only on proposing new solutions, but also on formal and systematic validation specially in industrial environment.

REFERENCES

[1] P. A. da Mota Silveira Neto, I. do Carmo Machado, J. D. Mcgregor, E. S. de Almeida, and S. R. de Lemos Meira. A systematic mapping study of software product lines testing. *Information and Software Technology*, 2011 (to appear).

[2] E. Engström and P. Runeson. Software product line testing - a systematic mapping study. *Information and Software Technology*, 53(1):2–13, 2011.

[3] J.D.McGregor. Building reusable test assets for a product line. In *Software Reuse: Methods, Techniques, and Tools*, pages 49–63, 2002.

[4] B. A. Kitchenham, D. Budgen, and O. P. Brereton. Using mapping studies as the basis for further research – a participant-observer case study. *Information and Software Technology*, 2011.

[5] B. P. Lamancha, M. P. Usaola, and M. P. Velthius. Software product line testing - a systematic review. In *ICSOFT (1)*, pages 23–30, 2009.

[6] J. D. McGregor. Testing a software product line. Technical Report CMU/SEI-2001-TR-022, Software Engineering Institute(SEI), 2001.

[7] K. Pohl, G. Böckle, and F. van der Linden. *Software Product Line Engineering – Foundations, Principles and Techniques*. Number ISBN: 978-3-540-24372-4. Springer, 2005.

[8] S. Reis, A. Metzger, and K. Pohl. A reuse technique for performance testing of software product lines. In *SPLIT' 05: Proceedings of the International Workshop on Software Product Line Testing*, Baltimore, Maryland, USA, 2006.

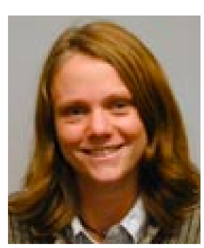
Paulo Silveira is a researcher from Reuse in Software Engineering Group (RiSE). Contact him at pamsn@rise.com.br

Per Runeson is a full professor at Lund University and research director for the industrial excellence center on Embedded Applications Software Enginering (EASE). Contact him at per.runeson@cs.lth.se.

Ivan Machado is a PhD student at Federal University of Bahia. Contact him at ivan.machado@dcc.ufba.br.

Eduardo Almeida is an assistant professor at Federal University of Bahia and head of the Reuse in Software Engineering (RiSE) Labs. Contact him at esa@dcc.ufba.br.

Silvio Meira is a full professor at Federal University of Pernambuco and chief scientist at Recife Center for Advanced Studies and Systems (C.E.S.A.R). Contact him at srlm@cin.ufpe.br.



Emelie Engström is a PhD student at Lund University. Contact her at emelie.engstrom@cs.lth.se.