Chapter 5 - Evaluation

5.1 Subject characterization

5.2 Experimental setup

5.3 Results

5.4 Discussion

5.5 Threats to Validity

5.5.1 Construct validity

5.5.2 Internal validity

5.5.3 External validity

EVALUATION

In this section, we present the evaluation of the techniques, which are approaches combined with automated testing tools for checking product line refinement. We evaluated them in evolution scenarios applied to product lines ranging up to 32 KLOC and more than a thousand product configurations.

5.1 SUBJECT CHARACTERIZATION

We evaluated 38 subjects. All of them consisting of SPL evolution scenarios. 35 of them is randomly selected in the TaRGeT’s SVN repository. TaRGeT SPL, is a tool that automatically generates functional tests from use case documents written in natural language. It has more than 32 KLOC on its last release and it is an Eclipse Rich Client Platform (RCP) application. [1]

As TaRGeT SPL was initially created as a single product, we had to find out the range of commits where we genuinely had a family of software-intensive systems. Being more specific, we had to discover the first commit on the SVN history where we do not only have source code, but also mandatory artifacts of Software Product Line as Feature Model, Configuration Knowledge and Component Model.

We implemented a python script to walk through SVN commits and check which point the project had already SPL artifacts associated with the code.

We implemented a python script to walk through SVN commits and check which point the project had already an SPL Architecture (i.e. a set of software-intensive systems that share a common, managed set of features developed from a common set of core assets in a prescribed way [5]).

After that, we randomly select pairs of commits to be evaluated.

The python script selects one revision not used before and creates the first branch.

In the second step, the script selects one more revision 3 commits above to compound the evolution pair.

Automatically run our tool set for checking this evolution pair.

We analyze these evolutions and identify which of them are refinements or not, considering changes in FM, CK and source code, based on developers’ notes in the SVN commits and using the product line refinement notion [3][4].

We also analyze 3 evolution pairs of Mobile Media [2], a product line for applications that manipulates music, video and photo on mobile devices.

The SPL developers believe these are safe evolution scenarios, and we use our tools to compare the four tools with respect to their performance and effectiveness for checking SPL refinement. In particular, we want to evaluate if they are able to correctly identify unsafe evolution scenarios.

~~We use our tools to check performance and effectiveness,~~ and then contrast their outcomes with the results we expect after manually comparing the evolutions. With these results we analyze the precision and recall of our techniques.

We also statistically compare the performance of our four techniques and we analyze if the time spent to check is significantly different between them.

**REFERENCES**

[1] Felype Ferreira, Lais Neves, Michelle Silva, and Paulo Borba. Target: a model based product line testing tool. In Tools Session at Brazilian Symposium on Software Engineering, Salvador, Brazil, 2010.

[2] Eduardo Figueiredo, N´elio Cacho, Claudio Sant’Anna, M´ario Monteiro, Uira Kulesza, Alessandro Garcia, S´ergio Soares, Fabiano Ferrari, Safoora Khan, Fernando Filho, and Francisco Dantas. Evolving software product lines with aspects: an empirical study on design stability. In ICSE, pages 261–270, 2008.

[3] Paulo Borba, Leopoldo Teixeira, and Rohit Gheyi. A theory of software product line refinement. In ICTAC, pages 15–43, 2010.

[4] Paulo Borba, Leopoldo Teixeira, and Rohit Gheyi. A theory of software product line refinement. Theoretical Computer Science, (To appear), 2012.

[5] Paul Clements and Linda Northrop. Software Product Lines: Practices and Patterns. Addison-Wesley Professional, August 2001.