Fostering a consistent SPL service ecosystem

José A. Galindo
Dept. Lenguajes y Sistemas Informáticos, University of Seville
Avda. de la Reina Mercedes s/n
Seville, Spain 41012
jagalindo@us.es

Pablo Fernandez
Dept. Lenguajes y Sistemas Informáticos, University of Seville
Avda. de la Reina Mercedes s/n
Seville, Spain 41012
pablofm@us.es

ABSTRACT

Nowadays, Software Product Line (SPL) researchers and practitioners have a diversity of Automated Analysis of Feature Models (AAFM) tools at their disposal. However, only a few applications are compatible between them. This, increases time to market of new applications and hinders application usage by researchers and practitioners. In this tutorial, we present how we can successfully create an ecosystem of SPL tools that can be integrated to offer a better user experience. Concretely, we will show how to i) easily provide a common REST interface to an SPL analysis tool thus, fostering application integration; ii) automatically offer a web graphical editor to interact with the tool, thus, promoting its usage by end users; and, iii) enable the governance of the applications and create a customized portal for pricing plans. Also, we show other benefits such as the automatic creation of demo sites for review purposes.

CCS CONCEPTS

• **Software and its engineering → Software product lines**; *Integrated and visual development environments*;

KEYWORDS

SPL, AAFM, web services govern, rest, ecosystem, tool

1 TOPIC

Variability-intensive systems (VIS) are those systems that need to cope with variability as part of its main functionality. Feature Models (FMs) [5] have become the *de facto* standard to represent common and variable characteristics in variability-intensive systems.

A comprehensive list of proposals and operations over feature models was presented by Benavides et al. [2] in 2010. Concretely, more than thirty operations over feature models were depicted. Those operations range from determining if a product is valid conforming to a feature model to the whole calculation of the number of different products in a product line. Nowadays, we can find multiple tools that enable different SPL analyses in different contexts and domains such as cloud computing [4]. Some of the most used ones are FaMa [3], FaMiLiAr [1] and FeatureIDE [6] among others.

However, this diversity of options makes the choice of the most appropriated tool a time-consuming activity. Users need to install, test and decide if it is the most appropriated tool for them. Moreover, there is no consistency regarding the interfaces between different applications making its integration difficult and costly.

Generally speaking, we find the following difficulties when using SPL related tools: *i*) There is no unified catalog of available tools and operations. *ii*) It is required to install and configure each operation; *iii*) There is no compatibility between two applications. *iv*) Graphical user interfaces are not always available. *v*) Its difficult to govern an SPL application according to a pricing plan.

In this context, we think that the proposed tutorial can help to bridge the gap between different users and practitioners while providing multiple benefits such as easing off the review process of software prototypes or the implementation of pricing plans for the different tools. First we will show how to enable interoperability and be ready-for-developers by easily offering a restful API (Rest wrapper). Then, we will see how we can provide a graphical interface to interact with our application (GUI Generator). Finally, we will present how to to model service agreements based on the number of features or number of rest calls to automatically create a customized portal for pricing plans (Application Govern). As a result we will end up with a central repository of SPL tools that enable ready to use applications and enable interoperability between them. Moreover, we will present how to show off our tool, prototype or application for SPL analysis with no extra coding thus, allowing to release usable software artifacts for reviewing purposes.

REFERENCES

[1] Mathieu Acher, Philippe Collet, Philippe Lahire, and Robert B. France. 2013. FAMILIAR: A domain-specific language for large scale management of feature models. *Science of Computer Programming (SCP)* 78, 6 (2013), 657–681.
[2] D. Benavides, S. Segura, and A. Ruiz-Cortés. 2010. Automated analysis of feature models 20 years later. *Information Systems* 35, 6 (2010), 615–636.
[3] David Benavides, Pablo Trinidad, Antonio Ruiz Cortés, and Sergio Segura. 2013. *FaMa.* Springer Berlin Heidelberg, Chapter FaMa, 163–171. DOI: http://dx.doi.org/10.1007/978-3-642-36583-6-11
[4] Jesús García-Galán, Omer F. Rana, Pablo Trinidad, and Antonio Ruiz-Cortés. 2013. Migrating to the Cloud: a Software Product Line based analysis. In *3rd International Conference on Cloud Computing and Services Science (CLOSER'13).*
[5] Kyo C Kang, Sholom G Cohen, James A Hess, William E Novak, and A Spencer Peterson. 1990. *Feature-oriented domain analysis (FODA) feasibility study.* Technical Report. DTIC Document. http://www.sei.cmu.edu/reports/90tr021.pdf
[6] Thomas Thüm, Christian Kästner, Fabian Benduhn, Jens Meinicke, Gunter Saake, and Thomas Leich. 2014. FeatureIDE: An extensible framework for feature-oriented software development. *Science of Computer Programming* 79 (2014), 70–85.