Describing Variability with Domain-Specific Languages and Models

Juha-Pekka Tolvanen
MetaCase
Jyväskylä, Finland
jpt@metacase.com

Steven Kelly
MetaCase
Jyväskylä, Finland
stevek@metacase.com

ABSTRACT

This tutorial will teach participants about domain-specific languages and models, where they can best be used (and where not), and how to apply them effectively to improve the speed and quality of product development within a product line.

CCS CONCEPTS

• **Software and its engineering → Software product lines**; **Model-driven software engineering**; **Domain specific languages**; *Abstraction, modeling and modularity*; System modeling languages; feature modeling;

KEYWORDS

Tutorial, domain-specific language, domain-specific modeling, product line variability, product derivation

ACM Reference Format:
Juha-Pekka Tolvanen and Steven Kelly. 2018. Describing Variability with Domain-Specific Languages and Models. In *SPLC '18: 22nd International Systems and Software Product Line Conference, September 10–14, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, Article 4, 1 page. https://doi.org/10.1145/3233027.3233059

1 INTRODUCTION

Variability is more than alternative or optional items. It also deals with behavior, ordering and various dependencies — often crossing several domains like hardware and software. Domain-Specific Languages (DSLs) allow covering this rich variation space, extending the capabilities of feature modeling and parameter tables [1]. With a DSL, the language definition itself defines the variation space and related rules, and language users follow this variation space when using the language to create variants. Domain engineers define the language and application engineers use it. In a fair number of cases the final products can be automatically generated from the high-level variant specifications.

Moving to language-based variation allows the development of variants based on common assets that would not be possible with other approaches. It also makes possible the creation of new variability that is not possible with other variability handling mechanisms [1, 3]. Perhaps even more importantly it allows application engineers to directly use the product concepts in the specification models and apply their own existing terminology.

2 TUTORIAL CONTENT

This tutorial introduces Domain-Specific Languages and models and looks at how they extend the other variability modeling approaches. This is followed by real-life examples (e.g. [2, 4]) from various industries and product lines, such as industry automation, smart watches, IoT devices and automotive systems. The main part of the tutorial addresses the guidelines for defining DSLs for product lines and for defining the generators that can automatically produce product variants.

On the language creation side, the success depends largely on identifying the variation space. Preparing for language definition comes down to conducting a thorough domain analysis. Domain engineers identify the abstractions for variation, the parts which are the same for all products within the product line, and the parts which can vary. Typically the key focus of language engineers is then to define a language which allows describing the varying parts within a given product line. In the tutorial we inspect some typical DSL patterns that are found from industry cases of product lines. On the variant generation side we look at different ways to implement generators to read the models and produce the full variant code or configuration.

The tutorial has a strong hands-on element, as after presenting how DSLs and related generators can be defined, small exercises will be done as group work. Results from the exercises will then be implemented for demonstration during the tutorial. By combining the theory, group work, and demonstration, the participants will learn practical skills for applying DSLs in product lines.

The target audience for the tutorial includes software and system architects, researchers, product managers, technology managers, consultants, and senior designers. The required expertise level is intermediate and no previous experience with modeling or code generation is needed.

REFERENCES

[1] Czarnecki, K., Eisenecker, U., Generative Programming, Methods, Tools, and Applications, Addison-Wesley, 2000.
[2] Kelly, S., Tolvanen, J.-P., Domain-Specific Modeling: Enabling Full Code Generation, Wiley-IEEE Computer Society Press, 2008.
[3] Tolvanen, J.-P., Kelly, S., Defining Domain-Specific Modeling Languages to Automate Product Derivation: Collected Experiences. Proceedings of the 9th International Software Product Line Conference, H. Obbink, K. Pohl, Springer-Verlag, LNCS 3714, 2005.
[4] Tolvanen, J-P., Kelly, S., Model-Driven Development Challenges and Solutions - Experiences with Domain-Specific Modelling in Industry. Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development, Science and Technology Publications, 2016.

SPLC '18, September 10–14, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-6464-5/18/09.
https://doi.org/10.1145/3233027.3233059