# Next-Generation Model-based Variability Management: Languages and Tools


Mathieu Acher, Raphaël Michel
PReCISE Research Centre
University of Namur, Belgium
{mac,rmi}@info.fundp.ac.be

Patrick Heymans
PReCISE Research Centre
INRIA Lille-Nord Europe
University of Namur, Belgium
Université Lille 1 – LIFL – CNRS , France
phe@info.fundp.ac.be



**Abstract**

Variability modelling and management is a key activity in a growing number of software engineering contexts, from software product lines to dynamic adaptive systems. Feature models are the defacto standard to formally represent and reason about commonality and variability of a software system. This tutorial aims at presenting next generation of feature modelling languages and tools, directly applicable to a wide range of model-based variability problems and application domains. Participants (being practitioners or academics, beginners or advanced) will learn the principles and foundations of languages and tool-supported techniques dedicated to the model-based management of variability.



**Résumé**

La modélisation et gestion de la variabilité est une activité essentielle dans un nombre croissant de contextes liés au génie logiciel, des lignes de produits logiciels aux systèmes dynamiques apatatifs. Les modèles de caractéristiques sont le standard defacto pour formellement représenter et raisonner sur la commonalité et la variabilité d'un système logiciel. Ce tutorial a pour objectif de présenter la nouvelle génération de langages et outils de modélisation de caractéristiques, directement applicable à un large ensemble de problèmes de variabilité basés sur les modèles et de domaines d'application. Les participants (qu'ils soient industriels ou académiques, débutants ou avancés) vont apprendre les principes et fondements des langages ainsi que des techniques, supportées par des outils, pour la gestion de la variabilité basée sur les modèles.



KEYWORDS: *Feature Models, Model Management, Variability Modelling, Automated Reasoning, Language, Model Composition and Differences*

MOTS-CLÉS: *Modèles de Caractéristiques, Gestion de Modèles, Modélisation de la Variabilité, Raisonnement Automatique, Langage, Composition et Différences de Modèles*

The length of the tutorial[1] is between 2 hours (minimum) and 5 hours (maximum), depending on the slot allocated by the organisers of the CIEL conference or/and the interest of the audience. Specific details about the topic, goal, schedule and supporting material of the tutorial are given in Section 1. Target audience and any pre-requisite background required by attendees are described in Section 2.

# 1 Description of the Tutorial

## 1.1 Topic

In many domains, systems heavily rely on software and have to be efficiently extended, changed, customized or configured for use in a particular context (e.g., to respond to the specific expectations of a customer) [22, 21]. The challenge for software practitioners is to develop and use the right *models*, *languages*, and *tool-supported techniques* to produce and maintain multiple similar software products (variants), exploiting what they have in common and managing what varies among them. A lack of flexibility in the reusable artifacts or a scope that is too large may have severe consequences on the model-based engineering process. Therefore the modelling and management of *variability* in software intensive systems, for example *Software Product Lines* (SPLs), is a critical task [10].

*Feature Models* (FMs) are the defacto standard for modelling and reasoning about commonality and variability of a software system (e.g., an SPL). FMs are widely used to compactly define the valid combinations of features supported by an SPL, each valid combination of features (called configuration) corresponding to a given product. Numerous model-based SPL approaches rely on FMs for various purposes: generation of product configurators, customisation of other artifacts (e.g., code, models) to derive members of an SPL [14, 15, 19, 20], reasoning about properties of an SPL [13, 12], etc. FMs have been intensively studied by academics during the last two decades. The research output is notably a formal semantics, the development of various automated reasoning operations and benchmarks, tools and languages [9]. Industrial tools (pure::variants, Gears) have also been developed and support FMs to fulfill the growing interest of practitioners. FMs will also be part of the Common Variability Language (CVL) standard[2].

The effort still continues since FMs are becoming increasingly complex (e.g., 6000+ features for Linux), handled by several stakeholders or organizations, and used to describe features at various levels of abstraction. Yet, barriers hindering a wider adoption of FMs, for example in industrial contexts, remain and have been reported (e.g., scalability [9], inadequate mechanisms to achieve separation of concerns [6], lack of relationships among languages [17]). To overcome current limitations, we have developed a set of complementary languages (TVL and FAMILIAR) and tool-supported techniques.

## 1.2 Goal

During the tutorial, participants will learn solutions applicable to a wide range of model-based variability problems and application domains while being supported by concrete tools and languages. We will notably expose new capabilities offered to modellers and show that, without the advances made, some model-based analysis would not be made possible in several case studies. The languages and tools have already been applied in practical contexts (industrial

[1] Similar tutorial proposals have been recently submitted to SPLC'12 and MODELS'12 conference.
[2] http://www.omgwiki.org/variability/doku.php

and academic), in different application domains (printers, semi-conductors, medical imaging, video surveillance, etc.) and for various purposes (extraction of FMs from legacy software artifacts [2], SPL evolution [8], management of multiple models [6, 3], model-based validation of SPLs [7], large scale configuration of FMs [18], etc.)

The tutorial aims at introducing new feature modelling languages, tools and their underlying environments, describing their theoretical foundations and showing how their combined use can practically handle the whole process from variability modelling to product configuration (see Figure 1). Firstly, we give the participants an overview of TVL, a text-based feature modelling language. We emphasize its concise, natural, yet expressive syntax that subsumes most existing FM dialects [11]. Secondly, we present FAMILIAR (for FeAture Model scrIpt Language for manIpulation and Automatic Reasoning) a language dedicated to the management of multiple FMs [5]. We introduce practical techniques to import, export, compose, decompose, edit, configure, reason about several FMs and combine these operations to realise complex model-based variability management tasks. Thirdly, we will present a tool suite for distributing and scheduling a *Feature-Based Configuration* (FBC) process in different steps, so that each stakeholder can focus on different views of a typically large and complex FM [1]. As effort has been made to support interoperability between the two languages, their combine use can be achieved. At the end of the tutorial, participants of the tutorial will be able to go further with the languages and modelling techniques, and to reuse them in practical or academic contexts.

### 1.3 Plan

The tutorial is based on slides and supporting demonstrations. An archive will be provided to participants, including tutorial slides, related articles, TVL models, FAMILIAR scripts and packaged tools to interactively play with the models during the tutorial. A concrete, end-to-end demonstration is used for modeling FMs with TVL and performing management operations of FMs with FAMILIAR. The demonstration is based on the re-engineering of a car configurator, that includes the modelling of configuration options using TVL, the management of the TVL model(s) using FAMILIAR and the configuration of FMs in the dedicated FBC environment.

As stated early, the estimated length of the tutorial is comprised between 2 hours and 5 hours. More specifically:

1. The tutorial will start with a short explanation of variability modelling and FMs. We will present TVL, exposing its textual nature and formal semantics. Using the illustrative example, we will explain the main constructions (feature, group cardinalities, attributes) and modularity capabilities of the language. The resulting TVL models will then be analysed by FAMILIAR.
   This part is mandatory and typically lasts 1 hour.

2. In the second part, we will present some relevant analyses and operations for the illustrative example and show how to perform them using FAMILIAR. These analyses will include usual satisfiability checking, detection of dead features and wrong cardinalities, but also more advanced operations (aggregation, differencing, slicing and merging of FMs).
   This part is mandatory and typically lasts 1 hour (minimum) and 2,5 hours (maximum). In its shortest form, we will only present the main capabilties and operators of FAMILIAR. In its longest form, we will insist on the theory and report on practical applications.

3. In the third part, we will show how the FBC tool suite can realise complex configuration strategies (non linear, distributed) usually found in real world configurators. We will

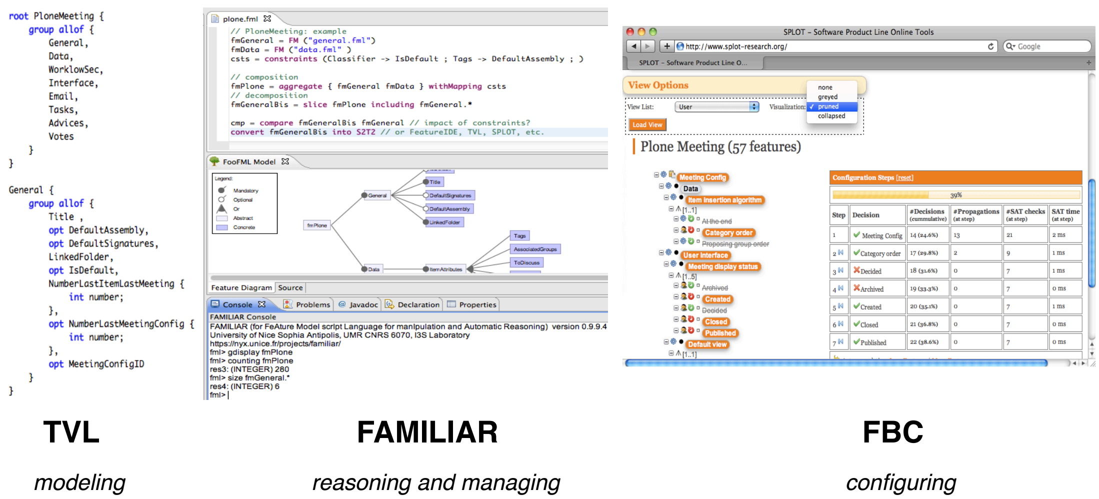


Figure 1: Languages and tools used in the tutorial

explain how the tool reacts and enforces the constraints defined in the FM. This part is optional (estimated length: between 30 minutes and 1 hour).

4. The session will end up with a short summary of what have been done, a broader overview of the applications already realised by our tool suite and a report on our experience with the design and development of the languages. It should strengthen discussions with the audience (tools information, applicability of the techniques, etc.).
   This part is optional (estimated length: between 10 minutes and 30 minutes).

## 1.4 Material and Experience

Numerous presentations of the languages and tools have already taken place in the past. An empirical evaluation of TVL on four real industrial projects has been published and presented at SLE'10 [16]. We plan to reuse introductory slides of TVL used in industrial contexts. TVL has also been presented in VaMoS'10. FAMILIAR has been presented in numerous venues: VaMoS'11 (tool demonstration), SAC'11 (formal presentation), french SPL day (tool demonstration), and ASE'11 (tool demonstration track). It is also currently being used in the french national project YourCast (`http://yourcast.unice.fr`, started in 2012) which aims at building a complex industrial strength SPL of model-based information broadcast systems.

Furthermore, the languages and tools are extensively used during an SPL course given to Masters' students at University of Namur. As a running project, students have to re-engineer car configurators using TVL, FAMILIAR and modelling techniques.

It should be noted that similar tutorials have been recently submitted to SPLC'12 and MODELS'12 conference. We plan to reuse the material prepared for those events.

The web page `http://www.info.fundp.ac.be/spltutorials/` gathers some available material we plan to reuse for the tutorial. It includes the slides and screencasts used during the aforementioned demonstrations or the teaching material and illustrative example used in the SPL course.

# 2 Audience

## 2.1 Target audience

The tutorial targets *software practitioners* – professionals working in industrial contexts: requirements engineers, domain analysists, software architects, or product line managers – that look for languages and tools to model and manage variability. For example, TVL has already been applied in four industrial case studies [16] while FAMILIAR can tackle numerous variability problems reported in the industry [6]. The tutorial also targets an *academic audience*. For example, the recent research results as well as state-of-the-art existing techniques (e.g., reasoning operations reported in [9]) are integrated in FAMILIAR. During the tutorial, academics will learn the design and developement of novel model-based operations (e.g., merging, slicing, differencing techniques for FMs) [5, 6].

## 2.2 Pre-requisite background

Attendees, being industrial practitioners or academics, may have basic knowledge about software product lines and variability, but it is not required. The tutorial targets both beginners and advances.

## 2.3 Target participants of CIEL'12

More specifically, PhD students or modellers unaware of variability modelling techniques will have the opportunity to learn a widely used modelling formalism (FM) as well as existing languages and tools they can directly use. Furthermore, advanced modellers will learn novel techniques that may be reused in their academical or industrial contexts. We particularly target people involved or interested in the development of *model management tools*. Our experience in several case studies has shown that syntactical mechanisms are likely to be insufficient for operations like the computation of differences of FMs [8] or the composition/decomposition of FMs [4, 6]. We have rather developed a set of semantical operations. We will present their foundations and report on our experience, which can benefit for the attendees.

# Acknowledgments

This work was partially funded by the Walloon Region under the NAPLES project, the IAP Programme, Belgian State, Belgian Science Policy under the MoVES project, the BNB, the FNRS, and the FP7 Marie-Curie COFUND program.

# References

[1] E.K. Abbasi, A. Hubaux, and P. Heymans. A toolset for feature-based configuration workflows. In *SPLC'11*, pages 65 –69. IEEE, 2011.

[2] Mathieu Acher, Anthony Cleve, Philippe Collet, Philippe Merle, Laurence Duchien, and Philippe Lahire. Reverse engineering architectural feature models. In *ECSA'11*, volume 6903 of *LNCS*, pages 220–235, 2011.

[3] Mathieu Acher, Philippe Collet, Alban Gaignard, Philippe Lahire, Johan Montagnat, and Robert France. Composing multiple variability artifacts to assemble coherent workflows. *Software Quality Journal (special issue: Quality Engineering for Software Product Lines)*, pages 1–46, 2011.

[4] Mathieu Acher, Philippe Collet, Philippe Lahire, and Robert France. Comparing approaches to implement feature model composition. In *ECMFA'10*, volume 6138 of *LNCS*, pages 3–19, 2010.

[5] Mathieu Acher, Philippe Collet, Philippe Lahire, and Robert France. A domain-specific language for managing feature models. In *SAC'11*, pages 1333–1340. PL Track, ACM, 2011.

[6] Mathieu Acher, Philippe Collet, Philippe Lahire, and Robert France. Separation of Concerns in Feature Modeling: Support and Applications. In *AOSD'12*, pages 1–12. ACM, 2012.

[7] Mathieu Acher, Philippe Collet, Philippe Lahire, Sabine Moisan, and Jean-Paul Rigault. Modeling variability from requirements to runtime. In *ICECCS'11*, pages 77–86. IEEE, 2011.

[8] Mathieu Acher, Patrick Heymans, Philippe Collet, Clément Quinton, Philippe Lahire, and Philippe Merle. Feature Model Differences. In *CAiSE'12*, LNCS. Springer, 2012. to appear.

[9] David Benavides, Sergio Segura, and Antonio Ruiz-Cortes. Automated analysis of feature models 20 years later: a literature review. *Information Systems*, 35(6), 2010.

[10] Lianping Chen, Muhammad Ali Babar, and Nour Ali. Variability management in software product lines: a systematic review. In *SPLC'09*, pages 81–90, 2009.

[11] Andreas Classen, Quentin Boucher, and Patrick Heymans. A text-based approach to feature modelling: Syntax and semantics of TVL. *Science of Computer Programming, Special Issue on Software Evolution, Adaptability and Variability*, 76(12):1130–1143, 2011.

[12] Andreas Classen, Patrick Heymans, Pierre-Yves Schobbens, and Axel Legay. Symbolic model checking of software product lines. In *ICSE'11*, pages 321–330, 2011.

[13] Andreas Classen, Patrick Heymans, Pierre-Yves Schobbens, Axel Legay, and Jean-François Raskin. Model checking lots of systems: efficient verification of temporal properties in software product lines. In *ICSE'10*, pages 335–344, 2010.

[14] Krzysztof Czarnecki and Micha Antkiewicz. Mapping features to models: A template approach based on superimposed variants. In *GPCE'05*, volume 3676 of *LNCS*, pages 422–437, 2005.

[15] Krzysztof Czarnecki and Krzysztof Pietroszek. Verifying feature-based model templates against well-formedness ocl constraints. In *GPCE'06*, pages 211–220. ACM, 2006.

[16] Arnaud Hubaux, Quentin Boucher, Herman Hartmann, Raphaël Michel, and Patrick Heymans. Evaluating a textual feature modelling language: Four industrial case studies. In *SLE'10*, volume 6563 of *LNCS*, pages 337–356. Springer, 2011.

[17] Arnaud Hubaux, Andreas Classen, Marcílio Mendonça, and Patrick Heymans. A preliminary review on the application of feature diagrams in practice. In *VaMoS'10*, pages 53–59, 2010.

[18] Arnaud Hubaux, Patrick Heymans, Pierre-Yves Schobbens, Dirk Deridder, and Ebrahim Abbasi. Supporting multiple perspectives in feature-based configuration. *Software and Systems Modeling*, pages 1–23.

[19] Vinay Kulkarni and Sreedhar Reddy. A model-driven architectural framework for integration-capable enterprise application product lines. In *Proceedings of the Second European conference on Model Driven Architecture: foundations and Applications*, ECMDA-FA'06, pages 1–12. Springer-Verlag, 2006.

[20] Gilles Perrouin, Jacques Klein, Nicolas Guelfi, and Jean-Marc Jézéquel. Reconciling automation and flexibility in product derivation. In *SPLC'08*, pages 339–348. IEEE, 2008.

[21] Klaus Pohl, Günter Böckle, and Frank J. van der Linden. *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag, 2005.

[22] Mikael Svahnberg, Jilles van Gurp, and Jan Bosch. A taxonomy of variability realization techniques: Research articles. *Softw. Pract. Exper.*, 35(8):705–754, 2005.