Fomal Methods and Analyses in Software Product Line Engineering
(Track Summary)

Ina Schaefer[1] and Maurice H. ter Beek[2]

[1] Technical University of Braunschweig, Germany
[2] ISTI–CNR, Pisa, Italy

1 Motivation

Software product line engineering (SPLE) [5,11] aims to develop a family of software-intensive systems via systematic, large-scale reuse in order to reduce time-to-market and costs and to increase the quality of individual products. In order to achieve these goals, formal methods offer promising analysis techniques, which are best applied throughout the product-line lifecycle so as to maximize their overall efficiency and effectiveness.

While some analysis approaches (e.g. for feature modeling and variant management) and formal methods and automated verification techniques and tools (e.g. CSPs, SAT solvers, model checkers and formal semantics of variability models) have already been applied to SPLE (cf. [12,3,13] and the references therein), a considerable potential still appears to be unexploited. In fact, despite the work that we just mentioned, the respective communities (SPLE, formal methods and analysis tools) are only loosely connected.

2 Goals

This track brings together researchers and practitioners interested in raising the efficiency and effectiveness of SPLE by applying formal methods and innovative analysis techniques. Participants review the state-of-the-art and practice in their respective fields, identify further promising application areas, report practical requirements and constraints from real-world product lines, discuss drawbacks and complements of the various approaches, or present recent emerging ideas and results. The two long-term objectives of the FMSPLE workshop series are:

1. to raise awareness and to find a common understanding of practical challenges and existing solution approaches in the different communities working on formal methods and analyses techniques for SPLE, and
2. to create a broader community interested in formal methods and analysis techniques for SPLs in order to keep SPLE research and tools up-to-date with the latest technologies and with practical challenges.

T. Margaria and B. Steffen (Eds.): ISoLA 2014, Part I, LNCS 8802, pp. 253–256, 2014.

While in the previous four years, FMSPLE has successfully been held as a workshop affiliated with the international Software Product Line Conference (SPLC), its 5th edition is held as a track at ISoLA in order to facilitate discussions with other application domains of formal methods, verification and validation. Its 6th edition will be held as a workshop at ETAPS 2015. Because of the highly interactive format of ISoLA tracks, locating FMSPLE as a track at ISoLA offers an excellent opportunity for exchanging results and experiences of applying formal methods and analysis techniques between SPLE and other application domains.

3 Contributions

The contributions of this track are separated into two parts. The first part consists of formal modeling approaches for variable software. The second part considers formal analysis, testing and verification techniques for variant-rich software systems and SPLs.

Part 1: Formal Modeling. Iosif-Lazar et al. [9] present a core calculus for separate variability modeling. The approach is inspired from the Common Variability Language (CVL), but aims at unifying other variability modeling approaches such as Delta Modeling and Orthogonal Variability Modeling (OVM). The introduced language, Featherweight VML, contains a single kind of variation point to define transformations of software artifacts in object models. Its semantics comprehensively formalizes variant derivation, encompassing feature models, variation points, implementation artifacts and transformations.

Collet [6] focuses on the modeling and management of multiple and complex feature models. This paper reports on the development and evolution of the FAMILIAR domain-specific language (for FeAture Model scrIpt Language for manIpulation and Automatic Reasoning) and toolset. The author presents the FAMILIAR language and discusses its various applications with advantages and drawbacks. Furthermore, he identifies challenges for feature modeling and management in the near future.

Damiani et al. [7] present a programming language approach for SPLs that builds on their existing work on delta-oriented programming and trait-based implementation of SPLs. In this approach, program modifications are expressed by delta modules which rely on the trait composition mechanism. This smooth integration of the modularity mechanisms provided by delta modules and traits constitutes a new approach for programming SPLs which is particularly well suited for evolving SPLs.

Broch Johnsen et al. [4] focus on deployment variability in virtualized product lines. Their approach is based on the ABS language which supports deployment models with a separation of concerns between execution cost and server capacity. This allows the model-based assessment of deployment choices on a product's quality of service. In this paper, the authors combine deployment models with the delta-oriented variability modeling to capture deployment choices as features when designing a family of products.

Part 2: Formal Analysis, Testing and Verification. Lochau et al. [10] propose a delta-oriented extension to the process calculus CCS, called DeltaCCS, that allows for modular reasoning about behavioral variability. In DeltaCCS, modular change directives, i.e. deltas, are applied to core processes in order to alter term rewriting semantics. Variability-aware congruences capture the preservation of behavioral properties defined by the modal μ-calculus between different CCS variants. A DeltaCCS model checker allows to efficiently verify the members of a family of process variants.

Devroey et al. [8] focus on coverage criteria for model-based testing of SPLs based on Featured Transition Systems (FTS). FTSs constitute a family-based representation of SPLs extending labeled transition systems such that transitions are moreover tagged with a feature. The authors define several FTS-aware structural testing coverage criteria and combine these with usage-based testing for configurable websites.

Ter Beek et al. [2] apply variability analyses on a small bike-sharing product line. To this aim, they adopt a chain of existing feature modeling and variability analysis tools (including S.P.L.O.T., FeatureIDE, Clafer, ClaferMOO and VMC) to specify a discrete feature model, non-functional quantitative properties and a behavioral model, and to perform a quantitative evaluation of the attributes of products and model checking over value-passing modal specifications.

Ter Beek and De Vink [1] present a proof-of-concept of a feature-oriented modular verification technique for analyzing the behavior of SPLs with the mCRL2 toolset. The behavioral model of a SPL is modularized into components, based on feature-driven borders, with interfaces that allow a driver process to glue them back together on the fly. This is a powerful abstraction technique that eases the model checking task, since it allows mCRL2 to concentrate on the relevant components (features) for a specific property, and moreover allows the result to be reused in other settings.

References

1. ter Beek, M.H., de Vink, E.P.: Towards Modular Verification of Software Product Lines with mCRL2. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014. LNCS, vol. 8802, pp. 368–385. Springer, Heidelberg (2014)
2. ter Beek, M.H., Fantechi, A., Gnesi, S.: Challenges in Modelling and Analyzing Quantitative Aspects of Bike-Sharing Systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014. LNCS, vol. 8802, pp. 351–367. Springer, Heidelberg (2014)
3. Borba, P., Cohen, M.B., Legay, A., Wąsowski, A.: Analysis, Test and Verification in The Presence of Variability (Dagstuhl Seminar 13091). Dagstuhl Reports 3(2), 144–170 (2013)
4. Johnsen, E.B., Schlatte, R., Tapia Tarifa, S.L.: Deployment Variability in Delta-Oriented Models. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part I. LNCS, vol. 8802, pp. 304–319. Springer, Heidelberg (2014)
5. Clements, P., Northrop, L.: Software Product Lines: Practices and Patterns. Addison Wesley, Longman (2001)
6. Collet, P.: Domain Specific Languages for Managing Feature Models: Advances and Challenges. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part I. LNCS, vol. 8802, pp. 273–288. Springer, Heidelberg (2014)

7. Damiani, F., Schaefer, I., Schuster, S., Winkelmann, T.: Delta-Trait Programming of Software Product Lines. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part I. LNCS, vol. 8802, pp. 289–303. Springer, Heidelberg (2014)
8. Devroey, X., Perrouin, G., Legay, A., Cordy, M., Schobbens, P.-Y., Heymans, P.: Coverage Criteria for Behavioural Testing of Software Product Lines. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part I. LNCS, vol. 8802, pp. 336–350. Springer, Heidelberg (2014)
9. Iosif-Lazăr, A.F., Schaefer, I., Wąsowski, A.: A Core Language for Separate Variability Modeling. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part I. LNCS, vol. 8802, pp. 257–272. Springer, Heidelberg (2014)
10. Lochau, M., Mennicke, S., Baller, H., Ribbeck, L.: DeltaCCS: A Core Calculus for Behavioral Change. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014, Part I. LNCS, vol. 8802, pp. 320–335. Springer, Heidelberg (2014)
11. Pohl, K., Böckle, G., van der Linden, F.: Software Product Line Engineering: Foundations, Principles, and Techniques. Springer, Heidelberg (2005)
12. Schaefer, I., Rabiser, R., Clarke, D., Bettini, L., Benavides, D., Botterweck, G., Pathak, A., Trujilol, S., Villela, K.: Software Diversity: State of the Art and Perspectives. STTT 14(5), 477–495 (2012)
13. Thüm, T., Apel, S., Kästner, C., Schaefer, I., Saake, G.: A Classification and Survey of Analysis Strategies for Software Product Lines. ACM Comput. Surv. 47(1), 1–6 (2014)