Feature-Oriented Contract Composition

Thomas Thüm
TU Braunschweig, Germany

Alexander Knüppel
TU Braunschweig, Germany

Stefan Krüger
Paderborn University, Germany

Stefanie Bolle
TU Braunschweig, Germany

Ina Schaefer
TU Braunschweig, Germany

ABSTRACT

A software product line comprises a set of products that share a common code base, but vary in specific characteristics called features. Ideally, features of a product line are developed in isolation and composed subsequently. Product lines are increasingly used for safety-critical software, for which quality assurance becomes indispensable. While the verification of product lines gained considerable interest in research over the last decade, the subject of how to specify product lines is only covered rudimentarily. One challenge is composition; similar to inheritance in object-oriented programming, features of a product line may refine other features along with their specifications.

In our work [1], we present a comprehensive discussion and empirical evaluation of *how to specify product lines* implemented by means of *feature-oriented programming*. In feature-oriented programs, implementation artifacts, such as methods, are distributed over the set of feature modules and subsequently composed together when the respective features are selected. Similar to this idea, contracts could be modularized, too, and are subsequently composed together with their respective methods. In particular, we investigate how refinement and composition of such specifications can be established and derive a notion of *feature-oriented contracts* comprising preconditions, postconditions, and framing conditions of a method (i.e., following the *design-by-contract* paradigm).

While both design by contract and feature-oriented programming have been hot research topics for more than two decades, their combination had rarely been explored. When features refine methods, an important question is whether refinement of their contracts is inevitable or not. However, unlike method composition where only the order of features is relevant, it seems that contract composition has to be handled differently according to certain scenarios. Consequently, a diverse set of composition techniques is required. In total, we identify and discuss six mechanisms to perform contract composition between original and refining contracts. Moreover, we identify and discuss desired properties for contract composition and evaluate which properties are established by which mechanism. As proof-of-concept and to enable larger evaluations, we developed tool support for feature-oriented contracts and their composition in FeatureHouse and FeatureIDE.

We conducted an empirical evaluation, in which we specified 14 product lines by means of contracts. To evaluate product lines specified with feature-oriented contracts, we applied three strategies. First, we implemented five product lines and feature-oriented contracts from scratch. Second, we decomposed six existing, object-oriented programs, which were formally verified before, including their contracts into a product line. That is, we identified features of the program and separated them into feature modules. Third, we specified three existing product lines with feature-oriented contracts. Each of these creation strategies is a typical application scenario of employing feature-oriented contracts and may impose different requirements for contract-composition mechanisms.

We gained six insights from our work. First, the majority of contracts defined for product lines are not contained in all products (i.e., family-wide specification is not sufficient). Second, product-line specifications can be given by specifying each feature module and usually even without derivative modules (i.e., feature-based specification is sufficient). Third, most but not all method refinements establish behavioral subtyping, which means that the *Liskov principle* does not apply to feature specifications. Fourth, we identified that four of our six mechanisms were superior to all other mechanisms for certain contract refinements, and thus we conclude that these four mechanisms should be used in concert. Fifth, fine-granular contract refinements and alternative method introductions often cause specification clones. Finally, most contract refinements only refine the postcondition while the precondition and framing condition remain unchanged. In particular, only eleven out of sixty contract refinements modified the frame.

CCS CONCEPTS

• **Software and its engineering → Software product lines**.

KEYWORDS

Feature-oriented programming, Software product lines, Design by contract, Deductive verification, Formal methods

ACM Reference Format:
Thomas Thüm, Alexander Knüppel, Stefan Krüger, Stefanie Bolle, and Ina Schaefer. 2019. Feature-Oriented Contract Composition. In *23rd International Systems and Software Product Line Conference - Volume A (SPLC '19), September 9–13, 2019, Paris, France.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3336294.3342374

REFERENCES

[1] Thomas Thüm, Alexander Knüppel, Stefan Krüger, Stefanie Bolle, and Ina Schaefer. 2019. Feature-oriented contract composition. *Journal of Systems and Software* 152 (2019), 83–107.

SPLC '19, September 9–13, 2019, Paris, France

ACM ISBN 978-1-4503-7138-4/19/09…$15.00
https://doi.org/10.1145/3336294.3342374