# **Entwicklung einer Software-Produktlinie mit automatischer Code-Generierung der Varianten**

Mistra Forest Kuipou Tchiendja ([kuipoutc@th-brandenburg.de](mailto:kuipoutc@th-brandenburg.de))

An der technischen Hochschule Brandenburg werden Bestellsysteme in der Lehre benötigt. Lehrende wollen nicht über die Semester hinweg dieselben Systeme, sondern veränderte Varianten anbieten. Das Ziel der Lehrenden ist einerseits, verschiedene Systeme aus unterschiedlichen Domänen von einem einzigen Grundsystem abzuleiten. Andererseits geht es um das Vermitteln von Konzepten, die sich in Code-Strukturen widerspiegeln und weniger um die Anwendung als solche oder den Domäneninhalt. In diesem Kontext hat Grigarzik (2020) in seiner Abschlussarbeit ein bestehendes Bestellsystem nachimplementiert. Weitere Nachimplementierungen werden benötigt, um möglichst viele Variante zu bilden. Die manuelle Nachimplementierung ist zeitaufwendig und fehleranfällig. Eine mögliche Lösung besteht darin, die Variantenbildung unter einem Dach als Software-Produktlinie (SPL) zu entwickeln und zu verwalten.

Diese Alternative erweist sich als sinnvoll, denn unter einer SPL wird eine Gruppe von Produkten verstanden, die eine gemeinsamen Grundstruktur aufweisen, welche nur einmal für alle Produktvarianten entwickeln wird. Darüber hinaus werden Unterschiede unter den Varianten dadurch gekennzeichnet, dass Features separat entwickelt und konfiguriert werden, um spezifische Anforderungen einer bestimmten Domäne gerecht zu werden (vgl. (Siegmund, Pukall, Soffner, Köppen, & Saake, 2009, S. Article 4, 1–7)).

Diese Arbeit beschäftigt sich mit der Frage, wie Variantenbildung von Bestellsystemen, unabhängig vom Diskursbereich, systematisch entwickelt werden. Das Hauptziel besteht darin, eine SPL so zu entwickeln, dass der Sourcecode der Varianten automatisch generiert werden. Die zu vermittelnden Konzepten sollten sich in den generierten Codes wieder erkennbar sein.

SPL Engenieuring ist kein einfacher Prozess und kommt in vielen Bereichen der Industrie vor. Industrielle Produkte werden in verschiedenen Varianten serienweise hergestellt. Im Herstellungsprozess wird das Konzept der Wiederverwendbarkeit in jeder Produktlinie umgesetzt. Dubinsky et al. (2013, S. 25-34) haben in ihrer Forschung in sechs großen Unternehmen herausgefunden, dass die Wiederverwendbarkeit systematisch durch Klonen realisiert werden kann. Die Umwandlung geklonter Produktlinien in strukturierten SPL-Modellen spart viel Zeit. Dabei entstehen (Code-)Duplikate, die refaktoriert werden müssen (Dubinsky, et al., 2013, S. 25-34). Die herkömmliche Refaktorierungsmethode muss angepasst werden, um die Variabilität vor und nach der Refaktorierung zu erhalten (vgl. Schulze et al. 2012, S. 73–81). Fenske et al. (2017, S. 316-326) zeigen den Prozess der schrittweisen Umwandlung von geklonten Produkten zu Produktlinien durch sinnvolles Refactoring von Produktvarianten. Die Autoren heben in ihre Ausführung, eine wichtige Eigenschaft einer Refaktorierung hervor: Alle potenziellen (Software-)Produkte innerhalb einer SPL bleiben kompilierbar und behalten ihr bisheriges Verhalten bei. Während Setyautami & Hähnle (2021, S. 1–9) ihren Schwerpunkt darauf legen, Java-Code aus UML-Modellen zu generieren, zeigen Setyautami, Adianto & Azurat (2018, S. 274–278) in ihrer Untersuchung, wie aus UML übesetzte Java-Codes, Produkte als Java-Objekte mittels Gradle Task simuliert werden.

In dieser Arbeit wird der Schwerpunkt darauf gelegt, einen Generator zu implementieren, um eine Variantenbildung einer Produktlinie zu generieren. Dafür werden Erkenntnisse aus Setyautami & Hähnle (2021, S. 1–9) hinsichtlich der Modellierung durch UML angewendet, um ein abstraktes Metamodell mithilfe der AST (Abtract Syntax Tree) zu erstellen. Dabei wird nach der Methode der Bottom-up-Entwicklung vorgegangen. Das heißt das Metamodell soll vom bereits bekannten Zielcode generiert werden. Auf dieser Weise werden die zu generierenden Codes aus Grigarzik (2000) als Grundlage dienen, das Startmodell zu konstruieren. Methoden der FOP (Feature oriented programming) und Refactoring-Techniken aus Schulze et al. (2012 S. 73–81) werden dafür genutzt. Durch geeignete Mappings wird der Generator entwickelt, welcher für eine Modell-zu-Code-Transformation zuständig ist.

Darüber hinaus wird der Generator in der Lage sein, nicht nur Anwendung, sondern auch grafische User-Interface (GUI) sowie Tests zu generieren. Die Erweiterung auf GUI und Tests war in den bereits zitierten Literatur im Zusammenhang mit SPLE nicht betrachtet worden. Der Generator soll nicht nur auf Bestellsysteme beschränkt werden. Auch die Möglichkeit die Persistenz-Schicht künftig bei der Generierung anzubinden, soll berücksichtigt werden.

# **Literaturverzeichnis**

Dubinsky, Y., Rubin, J., Berger, T., Duszynski, S., Becker, M. & Czarnecki, K. (2013). An Exploratory Study of Cloning in Industrial Software Product Lines. *2013 17th European Conference on Software Maintenance and Reengineering*, S. 25-34. https://doi.org/10.1109/CSMR.2013.13.

Fenske, W., Meinicke, J., Schulze, S., Schulze, S. & Saake, G. (20-24. Febuar 2017). Variant-Preserving Refactorings for Migrating Cloned Products to a Product Line. *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, S. 316-326. https://doi.org/10.1109/SANER.2017.7884632

Grigarzik, P. (7. Dezember 2020). Analyse eines Bestellsystems für zukünftige Variantenbildung in der Lehre. *Unveröffentlichte Bachelorarbeit am Fachbereich Informatik und Medien. Brandenburg an der Havel: Technische Hochschule Brandenburg.*

Schulze, S., Thüm, T., Saake, G. & Kuhlemann, M. (2012). Variant-Preserving Refactorings for Migrating Cloned Products to a Product Line. (A. f. Machinery, Hrsg.) *Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems (VaMoS '12)*, S. 73–81. https://doi.org/10.1145/2110147.2110156

Setyautami, M. R. & Hähnle, R. (February 2021). An Architectural Pattern to Realize Multi Software Product Lines in Java. (A. f. Machinery, Hrsg.) *15th International Working Conference on Variability Modelling of Software-Intensive Systems*(Article 9), S. 1–9. https://doi.org/10.1145/3442391.3442401

Setyautami, M. R., Adianto, D. & Azurat, A. (1. September 2018). Modeling Multi Software Product Lines using UML. (A. f. Machinery, Hrsg.) *Proceedings of the 22nd International Systems and Software Product Line Conference, Volume 1*(SPLC '18), S. 274–278. https://doi.org/10.1145/3233027.3236400