Olaf Zwintzscher: Komponentenbasierte & generative Software-Entwicklung; Generierung komponentenbasierter Software aus erweiterten UML-Modellen

Promotion: Ruhr-Universität Bochum
Erstgutachter: Prof. Dr.-Ing. Helmut Balzert
Zweitgutachter: Prof. Dr.-Ing. Peter Liggesmeyer
Datum der Prüfung: 25. Juni 2003
Veröffentlichung: W3L-Verlag, Herdecke-Dortmund, 2003; Online-Shop: www.W3L.de; ISBN 3-937137-50-5

Kurzfassung:
Obwohl die komponentenbasierte Softwareentwicklung in einigen Bereichen, wie z.B. bei der Entwicklung von Benutzungsoberflächen, sehr erfolgreich praktiziert wird, hat sie sich insgesamt jedoch nur mäßig etabliert. Ein Grund hierfür ist in erster Linie in der Komplexität der jeweiligen Komponententechniken zu sehen. Vor allem die Komposition von Softwarekomponenten sollte jedoch so einfach wie möglich sein, denn gerade dieses Prinzip stellt einen entscheidenden Erfolgsfaktor für den Einsatz einer Komponententechnik dar, wie das Beispiel der Benutzungsoberflächen belegt.

Das Ziel dieser Dissertation ist es daher, einen Ansatz zu realisieren, mit dessen Hilfe Komponenten auf einfache Art und Weise in ein Softwaresystem integriert werden können. Die Grundidee hierbei ist, den generativen Ansatz mit dem komponentenbasierten zu kombinieren: Komponenten werden in ein leicht zu verstehendes Modell auf hohem Abstraktionsniveau eingefügt und anschließend auf Knopfdruck in das Softwaresystem hineingeneriert. Hierzu wurde zunächst eine Notation entwickelt, die es ermöglicht, UML-Modelle um kompositorische Aspekte zu erweitern, denn bisher bietet die UML, die sich mittlerweile als Standardnotation in der Industrie etabliert hat, in diesem Bereich keine ausreichenden Ausdrucksmittel an.

Da moderne Anwendungssysteme in der Regel aus mehreren Schichten bestehen, die auf unterschiedlichen Computersystemen ablaufen und über das Netzwerk miteinander kommunizieren, ist es auch häufig erforderlich, eine Komponente entsprechend dieser Architektur in Teilkomponenten aufzuteilen. Da Komponenten dieser Art in der Regel einen kompletten Geschäftsprozess oder ein Geschäftsobjekt in Benutzungsoberfläche, Fachkonzept und Datenhaltung realisieren, werden sie daher als *Business Components* bezeichnet.

Während sich die übrigen generativen Ansätze auf die Betrachtung einer Schicht beschränken, wird in dieser Arbeit ein Ansatz realisiert, mit dessen Hilfe *Business Components* als Ganzes in die Einzelteile – Client und Server – einer dreischichtigen Anwendung hineingeneriert werden können.

Ein weiterer Grundgedanke dieser Arbeit ist es, dass auch Funktionalität existiert, die sich nicht sauber in einer Komponente lokalisieren lässt. Von der aspektorientierten Programmierung beeinflusst wurde daher eine Generatorkomponente entwickelt, die durch Generierungsschablonen und *Weaving Points* parametrisiert werden kann.

Zusätzlich wird hierdurch das Problem gelöst, dass auch für Komponenten mit einer spezifischen, benutzerdefinierten Schnittstelle eine Anbindung generiert werden kann: Der Komponentenentwickler realisiert neben der Komponente eine Generierungsschablone, die zusammen mit der Komponente ausgeliefert wird. Wird die Komponente in das Modell eingefügt und der Generierungsprozess angestoßen, liest der Generator die Generierungsschablone ein und erzeugt eine für die jeweilige Komponente spezifische Anbindung.