1 Zusammenfassung

Die *komponentenbasierte Softwareentwicklung* ist in aller Munde und weckt vielversprechende Erwartungen wie eine verbesserte Wiederverwendung, Überprüfbarkeit und Wartbarkeit, eine höhere Produktivität und Qualität in der Erstellung von Software. Doch zum jetzigen Zeitpunkt fehlt immer noch eine solide Basis auf der sich solche Erwartungen wirklich gründen könnten. Schlimmer noch, es liegt nicht einmal ein Konsens darüber vor, was eine *Komponente* eigentlich ist und welche Konzepte und Technologien für sie benötigt werden. Unzweifelhaft ist allerdings, das die *Komposition*, das heißt der Baustein-artige Zusammenbau von vorgefertigten Softwareeinheiten, das zentrale Konzept einer komponentenbasierten Sichtweise auf die Applikationserstellung bildet.

Die vorliegende Arbeit leistet einen Beitrag auf dem wünschenswerten Weg zu einer industriellen Softwareproduktion mit ihren entsprechenden Merkmalen, indem konzeptionelle Grundlagen und technische Mechanismen untersucht bzw. erarbeitet werden, die eine weitgehend Computer-gestützte und automatisierte Komposition vollständiger Anwendungen aus vorgefertigten, unter Umständen über Netzwerke bereitgestellten Einzelkomponenten erlauben.

Im Gegensatz zum Komponentenbegriff gilt das Konzept des *Dienstes* im Gebiet der Verteilten Systeme seit einigen Jahren als tiefgehend erörtert und gut verstanden bzw. etwa im Rahmen des Referenzmodells für offene verteilte Verarbeitung (RM-ODP) auch wohldefiniert. Die kommunikationstechnisch notwendige Betonung der *Schnittstellen* solcher Dienste hat auch zu einer deutlichen Fokussierung auf die Betrachtung von Schnittstellenkonzepten – die auch als *innere Systemgrenzen* bezeichnet werden können – in der übrigen softwaretechnologischen Diskussion geführt, wo diese Konzepte mit den Grundsätzen der Objektorientierung Entsprechungen finden. Die in einer typischen Client / Server-Sicht jedoch von einem höheren Isolationsgrad geprägten und entsprechend gut abgrenzbar vorliegenden Dienste kommen einer bausteinartigen Systemsicht allerdings zunächst auf natürliche Weise näher, insbesondere wenn deren *gemeinsame* Nutzung – im Sinne eines Mehrwertdienstes – angestrebt wird.

Die Ausgangssituation der vorliegenden Arbeit liegt dabei in der Forderung nach einer *direkten Verzahnung* der fachlichen Modellierung solcher Systeme mit der Realisierung (Implemen-

tation) im Rahmen des Entwicklungsprozesses zur Erstellung von Softwarelösungen. Der Autor ist der Ansicht, dass erst ein solch direkter Übergang die wünschenswerten Aspekte der theoretischen Behandlung von Softwaremodellen für die konkrete Entwicklungspraxis *erhält*. Die übliche Abbildung, die fach- bzw. domänenspezifische Konzepte erfahren, sobald sie von Entwicklern mehr oder weniger methodisch in konkrete Software überführt wird, ist in der Regel nicht verlustfrei. Dies führt in der Realität beispielsweise häufig dazu, dass im Modell noch formal zugesicherte Randbedingungen und Annahmen über die Software in der realen Applikation nicht mehr wirksam sind und somit etwa nicht zur Robustheit, Wartbarkeit und Portierbarkeit der erstellten Lösung beitragen können.

Daher resultiert aus dem Wunsch nach möglichst wenigen expliziten »Zwischenphasen« im Entwicklungszyklus, die Forderung nach einer weitgehenden *Automatisierung* der Softwareentwicklung. Ziel dieser Arbeit ist daher die Untersuchung der Möglichkeit eine *modellgetriebene Anwendungskonstruktion* zu etablieren und deren Randbedingungen und Implikationen zu betrachten. Im Vordergrund steht dabei der praktische Nachweis, dass die dabei zu Grunde gelegte generative und Architekturzentrierte Sicht der Anwendungserstellung sinnvoll realisiert werden und einen wesentlichen Beitrag zu einer produktiven Entwicklung von Anwendungssystemen leisten kann.

Wesentliche These ist hierbei, dass eine komponentenbasierte Sichtweise ausgezeichnet geeignet ist, um das gewünschte Konstruktionsprinzip im Rahmen der Leitbilder »Produktlinien« und »Anwendungsfamilien« realisieren zu können. Die Arbeit kann dabei selbstverständlich nicht den gesamten Themenkomplex abdecken, sondern konzentriert sich auf die beiden Schwerpunkte der *Typisierung* und der *Adaption* von Softwarekomponenten als Grundlage für die Funktionsweise automatischer Konstruktionsmechanismen.

Diese Schwerpunkte werden begleitet von der Betrachtung von Werkzeugen zur fachnahen Architekturspezifikation und der Technologie-unabhängigen Generierung der entstehenden Softwareapplikationen auf der technischen Ebene sowie konsistenzerhaltenden Kompositionsmodellen auf der theoretischen Ebene.