Vergleichende Analyse von Technologien der generativen Programmierung

Das Paradigma der generativen Programmierung ist mittlerweile etabliert [3], und Technologien, die den aktuellen Stand dieses Ansatzes repräsentieren, bieten mächtige Abstraktionsmechanismen zur Entwicklung von Softwaresystem-Komponenten. Die mögliche Relevanz des Paradigmas für die gezielte Erstellung wiederverwendbarer Lösungen von Architektur-horizontalen Problematiken ist prinzipiell verstanden, wird aber erst seit kurzem [1] konsequent in Beispielsystemen untersucht und demonstriert.

Die vorliegende Veröffentlichung identifiziert verschiedene typische Problemstellungen, die einer softwaretechnisch hoch qualitativen Lösung mittels generativer Programmierung zugänglich sind. Dazu gehören als instruktives Beispiel das Erzeugen von Zugriffsoperationen für objektorientierte Datenkapseln ebenso wie die komplexen Probleme der Erzeugung einer transparenten Datenhaltungszugriffsschicht für relationale Datenbanken oder die Erzeugung von Stellvertreterobjekten im Rahmen eines verteilten Fernauruf-Mechanismus. Für diese Probleme werden mögliche Lösungen mittels repräsentativer Technologien wie *Jasper* oder dem aktuellen C++-Template-Mechanismus vorgestellt und kritisch analysiert.

Als Mehrwert dieser vergleichenden Analyse wird ein wechselseitig vertieftes Verständnis von Problemen und den verschiedenen Lösungsansätzen der generativen Programmierung gebildet. Darüberhinaus wird als Beitrag ein allgemeines strenges Typsicherheitskriterium für Schablonen-Programme formuliert, das über die unmittelbare Typsicherheit des Generats hinausgeht indem es die prinzipielle Typsicherheit aller Generate in allen Wiederverwendungen garantiert. Es wird herausgearbeitet, daß dieses Typsicherheitskriterium unter geeigneten Einschränkungen der Ausdrucksmächtigkeit von Schablonen-Mechanismen prinzipiell statisch sichergestellt werden kann. Deswegen kann argumentiert werden, daß diese im folgenden als *überspannend* bezeichnete Typsicherheit als Eckpfeiler einer streng getypten generativen Programmierung gewürdigt werden kann. Die Relevanz eines Konzepts überspannender Typsicherheit wurde schon sehr konkret im Rahmen der Entwicklung einer streng getypten Technologie [2] zur dynamischen Generierung von Web-Seiten erkannt.

Die zuvor angestrengte vergleichende Analyse wird ausgedehnt auf die Fragestellung der Unterstützung zur Sicherstellung überspannender Typsicherheit. In weitergehenden Untersuchungen soll aus der kritischen Diskussion ein eigener Vorschlag für einen Programmschablonen-Mechanismus abgeleitet werden.

Literatur

[1] Attardi, G., Cisternino, A. "Template Metaprogramming an Object Interface to Relational Tables". In: Proceedings of Reflection 2001, LNCS 2192, 266-267, Springer-Verlag, Berlin, 2001.

[2] Draheim, D., Weber, G. "Strongly Typed Server Pages". In: Proceedings of the Fifth Workshop on Next Generation Information Technologies, voraussichtlich August 2002.

[3] Czarnecki, K., and Eisenecker, U. "Generative Programming: Methods, Tools, and Applications". Addison Wesley Publishers, 2000.

[4] Nizhegorodov, D. "Jasper: Type Safe Compile-Time Reflection Language Extensions and MOP Based Templates for Java". In: Proceedings of ECOOP'2000 Workshop on Reflection and Metalevel Architectures, 2000.