Dynamische Metamodellierung als Methode zur Definition einer operationalen Semantik für die UML

Reiko Heckel, Stefan Sauer
Universität Paderborn, Fachbereich Mathematik – Informatik, D 33095 Paderborn
{reiko|sauer}@uni-paderborn.de

Im Vergleich zum Stand der Technik bei der Definition und Implementierung von (textuellen) Programmiersprachen ist die Situation im Bereich visueller Modellierungssprachen wie der Unified Modeling Language (UML, [3]) weitgehend unbefriedigend. Es gibt z.B. keine vergleichbaren Methoden zur Syntax- und Semantikbeschreibung, die genutzt werden könnten, um Compiler oder Interpreter zu generieren.

Das hat u.a. zur Folge, daß Sprachen wie UML keine formale Ausführungssemantik haben. Neben allgemeiner Verunsicherung über die Bedeutung gewisser Modellierungskonzepte (insbesondere im Bereich der Nebenläufigkeit) verhindert das Fehlen einer operationalen Semantik die formale Analyse von Modellen. Im Zusammenhang mit existierenden Implementierungen zur Simulation und Codegenerierung aus Modellen stellt sich darüber hinaus die Frage, ob es dem Entwickler der entsprechenden Systeme überlassen werden sollte, die offenen semantischen Entscheidungen zu treffen.

Als Ansatz zur Definition der operationalen Semantik (der ausführbaren Anteile) von UML schlagen wir das Konzept der dynamischen Metamodellierung vor. Das statische Metamodell, das in Form eines Meta-Klassendiagramms gegeben ist und die abstrakte Syntax der Modellierungssprache angibt, wird dabei u.a. um Operationen erweitert, die einen Interpreter für UML-Diagramme implementieren. Die Ausführung dieser Operationen wird mit Hilfe einfacher, auf Graphtransformationsregeln beruhender Kollaborationsdiagramme beschrieben.

Damit kombinieren wir zwei erfolgreiche Ansätze zur Semantik nebenläufiger Systeme mit dem Konzept der Metamodellierung in visuellen Modellierungssprachen. Die SOS-Methode (Structured Operational Semantics, [4]) wurde entwickelt, um die Semantik von Programmiersprachen zu definieren. Zunächst wird dazu die abstrakte Syntax der Programme um Zustandsinformationen erweitert. Auf den so gebildeten Ausdrücken wird mit Hilfe logischer Ableitungsregeln eine Zustandsüberführungsrelation definiert, die das Verhalten eines abstrakten Interpreters repräsentiert.

Um dieses Konzept zur Spezifikation eines abstrakten UML-Interpreters verwenden zu können, wird die textuelle Darstellung von Programmen und Programmzuständen durch eine graphische Darstellung ersetzt, die durch das statische Metamodell definiert wird. Die formale Grundlage für diesen Übergang von einer textuell basierten zu einer graphischen Methode bildet die regelbasierte Graphtransformation [2], ein formaler Ansatz zur visuellen Spezifikation zustandsbasierter Systeme, die sich vor allem im Bereich von Datenbanken, objektorientierten oder verteilten Systemen bewährt. Die aus der Kombination resultierende *graphische SOS-Methode* wird in [1] formal definiert. Sie gestattet eine Spezifikation der dynamischen Semantik von UML auf der Grundlage des Standard-Metamodells, die sowohl mathematisch präzise als auch anschaulich ist, und die ohne besondere Kenntnis formaler Methoden z.B. von den Entwicklern von Werkzeugen genutzt werden kann.

Zur Veranschaulichung unseres Ansatzes betrachten wir ein Fragment der Zustandsdiagramme (Statechart Diagrams) in UML. Abb. 1(a) zeigt den entsprechenden Ausschnitt aus dem Meta-Klassendiagramm, das in den Klassen Event und Action um die Meta-Operationen occur bzw. perform erweitert wurde. Eine neue Meta-Assoziation current wurde zwischen Object und State zur Aufnahme des aktuellen Kontrollzustands eingefügt. Das hierzu konforme Kollaborationsdiagramm in Abb. 1(b) spezifiziert die Meta-Operation perform. Im Sinne einer Deduktionsregel für Übergänge zwischen Interpreterzuständen gestattet es, aus ei-

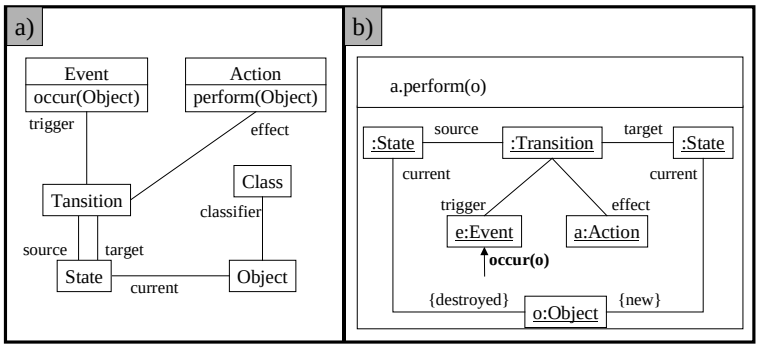

Abbildung 1: (a) Ausschnitt aus dem Klassendiagramm des Metamodells. (b) Darstellung einer graphischen SOS-Regel als Kollaborationsdiagramm.

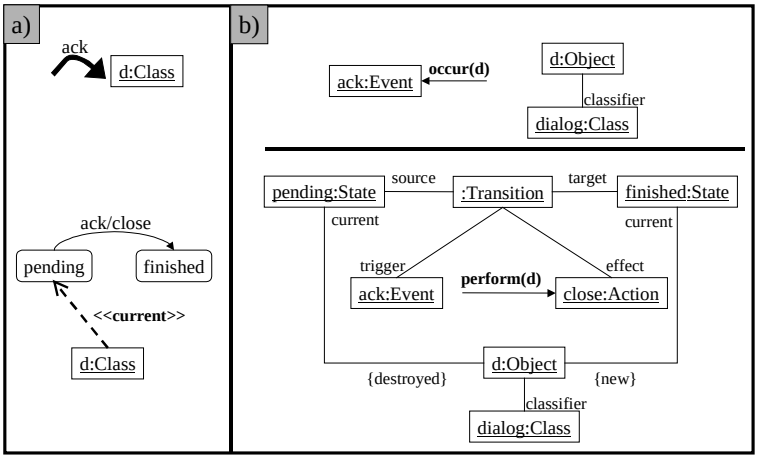

Abbildung 2: (a) Beispielhafte Darstellung in konkreter UML-Syntax mit Erweiterungen für das Eintreten von Ereignissen und für den aktuellen Objektzustand. (b) Deduktionsregel für dasselbe Szenario in abstrakter Syntax.

nem Übergang mit Label e.occur(o) einen Übergang mit Label a.perform(o) herzuleiten, bei dem der aktuelle Objektzustand von der Quelle zum Ziel der Transition wechselt. Prozedural bedeutet das die Ausführbarkeit der Meta-Operation perform, falls die im Kollaborationsdiagramm dargestellte Situation eintritt. Eine Instanz dieser graphischen SOS-Regel zu der in Abb. 2(a) gezeigten (erweiterten) konkreten UML-Syntax ist in Abb. 2(b) zu sehen. In der ausführlichen Darstellung ist die abstrakte Syntax der Regelprämisse (oben) und der Konklusion (unten) explizit dargestellt. Tritt das Ereignis ack bei Objekt d ein, so macht d, falls es im Zustand pending ist, einen Zustandsübergang nach finished durch und führt die Aktion close aus.

Literatur

[1] A. Corradini, R. Heckel, and U. Montanari. Graphical operational semantics. In *Proc. Graph Transformation and Visual Modelling Techniques, Geneva, Switzerland*, 2000.

[2] H. Ehrig, G. Engels, H.-J. Kreowski, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation, Volume 2: Applications, Languages, and Tools.* World Scientific, 1999.

[3] Object Management Group. Unified Modeling Language Specification, V 1.3, 1999.

[4] G. Plotkin. A structural approach to operational semantics. Tech. Rep. DAIMI FN-19, Aarhus University, 1981.