Design-by-Contract für funktionale Sprachen mit verzögerter Auswertung

Stefan Wehr
Universität Freiburg
wehr@informatik.uni-freiburg.de

(In Zusammenarbeit mit Markus Degen und Peter Thiemann)

15. April 2007

Hintergrund *Design-by-Contract* [2] ist eine Methodologie, um das Erstellen von korrekten Programmen zu erleichtern. Ein *Contract* ist dabei ein Prädikat, welches bestimmte Eigenschaften des Programms kodiert. Typischerweise wird die Gültigkeit von Contracts durch *Contract Monitoring* zur Laufzeit überprüft. Dabei sollen zwei Eigenschaft gelten: (1) Falls ein Programm keinen Contract verletzt, soll Contract Monitoring die Bedeutung des Programms nicht ändern. (2) Contracts sind idempotent, d.h. es ist egal ob ein Contract einmal oder mehrmals angewandt wird.

Problemstellung Die ursprüngliche Umsetzungen von Design-by-Contract für funktionale Sprachen [1] ist im Umfeld von Scheme anzusiedeln, einer Sprache mit strikter Auswertung. Überträgt man nun diesen Ansatz auf eine Sprache mit verzögerter Auswertung, ergibt sich durch Contract Monitoring das Problem, dass Contracts möglicherweise auf noch unausgewertete Teile des Programms zugreifen. Um nun die beiden oben aufgeführten Eigenschaften nachzuweisen, genügt es daher für solche Sprachen nicht, lediglich gewisse Seiteneffekte wie etwa Nichtterminierung aus den Prädikaten der Contracts zu verbannen (was für strikte Sprachen ausreichend ist), sondern es müssen zusätzlich auch die Seiteneffekte der Argument eines Contracts (d.h. der Ausdrücke, die mit einem Contract versehen sind) eingeschränkt werden. Damit wird das Programmieren mit Contracts starkt eingeschränkt.

Lösung Ausgehend von einem Typ- und Effektsystem für eine funktionale Sprache mit verzögerter Auswertung und Contracts wird die ursprüngliche, problematische Form des Contract Monitorings formalisiert und die angesprochenen Eigenschaften 1 und 2 nachgewiesen. Anhand des Beweises kann man nachvollziehen, welche Effekteinschränkungen nötig sind. Auf dieser Erfahrung aufbauend wird eine neue Form des Contract Monitorungs für Sprachen mit verzögerter Auswertung entwickelt, welche sich besser mit der Auswertungsstrategie verträgt, für praktische Zwecke gut geeignet ist und für die die genannten Eigenschaften gelten. Eine Haskell Implementierung für die neue Form des Contract Monitorings liegt vor.

Literatur

[1] Robert Bruce Findler and Matthias Felleisen. Contracts for higher-order functions. In Simon Peyton-Jones, editor, *Proc. Intl. Conf. Functional Programming 2002*, pages 48–59, Pittsburgh, PA, USA, October 2002. ACM Press, New York.

[2] Bertrand Meyer. *Object-Oriented Software Construction*. Prentice-Hall, second edition, 1997.