Eine Sprache zur Beschreibung planarer aktiver Systeme

Hermann von Issendorff
D-21745 Hemmoor, Hauptstrasse 40
Email: hviss@issendorff.com

Zusammenfassung:

Die Sicherheit und Zuverlässigkeit programmierbarer Systeme hängt wesentlich von der Kompatibilität der Hard- und Software ab, d.h. davon, ob die Hardware genau das tut, was die Software vorschreibt. Dies läßt sich am besten durch eine formale Sprache erreichen, die die Hard- und Software gemeinsam beschreibt. Eine solche Sprache wird hier vorgestellt und gezeigt, daß sich aus ihr durch Abstraktion und Spezialisierung bekannte Programmiersprachen bzw. eine neue Layout-Sprache ableiten lassen.

In der Vergangenheit habe ich mehrfach über die Fortschritte bei der Entwicklung einer Aktionalgebra genannten Termalgebra zur Beschreibung digitaler Hardware berichtet, z.B. in [1]. Diese Aktionalgebra hat sich in den letzten Monaten zu einer Sprache für aktive planare Systeme gemausert. Aktive Systeme, darmit sind jede Art von Systemen gemeint, die räumliche Abmessungen haben und auf denen irgendwelche Vorgänge oder Abläufe stattfinden. Die Systemkomponenten werden durch Aktionen dargestellt, das sind bedingte Funktionseinheiten mit einer Eingabe und einer Ausgabe. Aktive Systeme sind z.B. analoge oder digitale Schaltungen, Rechnernetze oder aber auch Verwaltungs- und Produktionssysteme. Planar bedeutet, daß die Aktionen eine flächenhafte Anordnung haben, wobei ihre Vernetzung kreuzungsfrei auf zwei benachbarten Ebenen realisiert wird.

Das Besondere der Aktionsprache gegenüber anderen Programmiersprachen ist ihre kausal begründete Semantik, die eine diskrete raumzeitliche Topologie definiert. Die kausale Abhängigkeit von zwei Aktionstermen definiert zwischen ihnen eine räumliche wie auch eine zeitliche Richtung, die sich beide zu einer Ablaufrichtung zusammenfassen lassen. Die Ablaufrichtung definiert damit eine Ordnung der kausalen Abhängigkeit. Die kausale Unabhängigkeit von zwei Termen definiert eine zur Ablaufrichtung orthogonale Dimension, auf der zwischen links und rechts unterschieden werden kann, und damit eine Ordnung, die sowohl für die Terme wie auch ihre Eingabeelemente und Ausgabelemente gilt. Die kausale Abhängigkeit und Unabhängigkeit spannen eine Fläche auf, in der jedes Akton und jedes Element der Ein- und Ausgabe eine eindeutige topologische Position hat. Sind die Abmessungen der Aktionen bekannt, wird aus der räumlichen Topologie eine Raummetrik Sind die Generierungszeiten der Ausgabeelemente der Aktionen bekannt, wird aus der temporalen Topologie eine Zeitmetrik.

Die Aktionsprache wird zu einer Layout-Sprache, wenn man von der Funktionalität abstrahiert. Andererseits kann sie schrittweise in spezielle Software- bzw. Hardware-Sprachen abstrahiert werden, wobei die Einführung algebraischer Axiome eine besondere Rolle spielt. Das Assoziativaxiom zerstört die kausale Abhängigkeitsordnung und zwingt zur Einführung quantifizierter Verbindungsoperatoren. Das Distributivaxiom schnürt die kausale Abhängigkeit in einen einzigen Kontrollpfad. Zusammen ergibt sich daraus eine Schaltbildsprache in der Art von B. Möller [2]. Fordert man zusätzlich noch Synchronisation für den Kontrollpfad, erhält man eine Ablaufplansprache wie die von Stefanescu [3]. Das Kommutativaxiom zerstört die Unabhängigkeitsordnung und zwingt zur Einführung individueller Bezeichner. Die verbleibende Semantik genügt als Basis für Parallelsprachen. Einführung von Transitivität über der kausalen Abhängigkeit schließlich führt auf die sequentiellen Programmiersprachen.

[1] Hermann von Issendorff: Vollständige formale Beschreibung digitaler Schaltwerke; PARS-Mitteilg. 15 (1996)
[2] Bernhard Möller: Deductive Hardware Design - A Functional Approach; LNCS 1546 (1998)
[3] Manfred Broy, Gheorghe Stefanescu: The Algebra of Stream Processing Functions, TUM-19620 (1996)