BERUFSAKADEMIE MANNHEIM
STAATLICHE STUDIENAKADEMIE
FACHRICHTUNG INFORMATIONSTECHNIK

Seminar Informationstechnik
Dozent: Prof. Dr. Tischhauser

Grundlagen algebraischer Spezifikation

von

Christian Hasselbach
&
Michael Krimgen

29. April 2004

Ehrenwörtliche Erklärung

Wir versichern hiermit, die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt zu haben. Des Weiteren sind alle Stellen, die den benutzten Quellen wörtlich oder inhaltlich entnommen wurden, als solche kenntlich gemacht worden.

Mannheim, den 29. April 2004

Christian Hasselbach	Michael Krimgen

Inhaltsverzeichnis

1 Einleitung und Begriffsdefinition

1.1 Algebraische Spezifikation

Der Begriff "algebraische Spezifikation" setzt sich aus den Wörtern "Algebra" und "Spezifikation" zusammen. Das Wort "Algebra" enstand aus dem Titel eines Buches des arabischen Mathematikres Al-Khowarizmi. Dieser lebte um 780 nach Christi Geburt in Bagdad und versuchte die axiomatische, griechische Mathematik mit der algorithmischen Hindu-Mathematik zu verbinden. Eine solche Verbindung zwischen der allgemeinen Algebra und der Informatik soll durch die algebraische Spezifikation entstehen. Hiermit soll erreicht werden, dass das Axiomensystem der Algebra auf abstrakte Datentypen (ADT) und Algorithmen in der Informatik angewandt werden kann um eine qualitative und effiziente Software zu erreichen. Dies kann erreicht werden, indem das zu Grunde liegende Axiomsystem auf die formale Beschreibung der Software angewandt wird, sodass die Korrektheit mathematisch *verifiziert* werden kann. Man kann daher sagen, dass die algebraische Spezifikation ein Teilgebiet der allgemeinen Algebra ist und ihre Anwendung im Software-Engineering findet. In diesem Dokument sollen zunächst einige Begriffe, die im Zusammenhang mit der algebraischen Spezifikation von abstrakten Datentypen wichtig sind, erläutert werden. Anschließend werden verschieden Beispiele abstrakter Datentypen vorgestellt und die Vorgenhensweise bei deren Implementierung beschrieben. Zum Schluß wird noch ein Verfahren zur Programmverifikation anhand abstrakter Datentypen vorgestellt sowie Probleme und Einschränkungen algebraischer Spezifikation zusammengefasst.

1.2 Begriffe

Die folgenden Begriffe werden in vielen mathematischen Abhandlungen verwandt. Da einige auch hier verwendet werden, sollen diese zum besseren Verständnis kurz erläutert werden:

Axiom (*griechisch: Grundwahrheit*) Eine als selbstverständlich angenommene Aussage. Diese braucht nicht bewiesen zu werden, andernfalls würde die Beweiskette niemals enden.

Definition (*lateinisch: de=ab, finis=Grenze*) genaue Beschreibung eines Begriffes durch Erklärung und Beschreibung.

Lemma Satz, der als Hilfssatz für einen anderen dient.

Satz/Theorem Eine aus den Axiomen hergeleitete Erkenntnis, die eine gültige Aussage innerhalb einer mathematischen Theorie darstellt.

Korollar Sammlung von Feststellungen und Folgerungen, die sich aus einem Satz oder einer Definition ergeben. Diese Folgerungen sind meist trivial.

2 Allgemeine Algebren

Allgemeine Algeberen sind vergleichbar mit Datentypen in der Informatik. Sie bestehen aus folgenden drei Komponenten:

- einer oder mehreren Mengen,
- auf diesen Mengen definierten Operationen (0 bis m),
- auf diesen Mengen definierten Relationen (0 bis n).

Operationen sind hierbei die arithmetischen und Relationen die logischen Verknüpfungen. Es gibt nur eine bestimmte Anzahl an möglichen Operationen und Relationen, da diese teilweise Abhängig von der verwendeten Menge sind. Ein Beispiel für allgemeine Algebren ist $[\mathbb{N}; +, *; <]$, wobei $\mathbb{N}$ für die Menge der natürlichen Zahlen, + und * für Operationen und $<$ für eine Relation steht.

3 Signaturen

Definition: Eine Menge Ω an Operationssymbolen bildet zusammen mit der Stelligkeitsabbildung $\alpha : \Omega \to \mathbb{N}$ die Signatur $\Sigma = (\Omega, \alpha)$.

Signaturen gehören zur Syntax elementarer Sprachen und stellen in der allgemeinen Algebra einen Typ einer Menge Ω von Operationssymbolen mit zugehöriger Stelligkeitsabbildungen α dar. Das Symbol für eine Signatur ist das griechische, große Sigma Σ. Daraus ergibt sich $\Sigma = (\Omega,\ \alpha)$, wobei $\alpha : \Omega \to \mathbb{N}$ heißt, daß-Ergebnis der jeweiligen Stelligkeitsbestimmung ist eine natürliche Zahl ist. Die Operationssymbole untergliedern sich in drei verschieden Zeichentypen, und zwar in Relationszeichen, Funktionszeichen und Individuenzeichen. So sind in der Menge $\Omega = \{+, *, <, 0, 1\}$ der elementaren Arithmetik + , * Funktionszeichen, $<$ ist ein Relationszeichen und 0 , 1 sind Individuenzeichen. Hierbei kommt jedem dieser Zeichen eine besondere Stellenzahl zu, so ist beispielsweise $\alpha(+)=2$ da zwei Objekte mit diesem Zeichen verbunden werden. Daraus folgt, es handelt sich bei dem + um ein zweistelliges Funktionszeichen. Das * in dem obigem Beispiel verhält sich äquivalent dazu. Bei dem $<$ handelt es sich ebenfalls um ein zweistelliges Zeichen, so dass es ein zweistelliges Relationszeichen ist. 0 und 1 haben als Stellenzahl jeweils die Null, da es sich bei diesen um Individuenzeichen also um Konstanten handelt. Ein typisches einstelliges Funktionszeichen ist beispielsweise das $\neg$ (nicht), welches zur Negierung von Ausdrücken dient.

4 Syntax / Semantik

Die Syntax dient der Untersuchung der (mathematischen) Sprache und deren Regeln. Jede Aussage einer mathematischen Theorie besteht aus Grundbegriffen, den so genannten Prädikaten (Zeichen) und Funktionen (Ausdrücken). Für diese einzelnen Elemente werden Symbole definiert, die in ihrer Gesamtheit die Signatur der gegebenen Theorie bilden. In der Syntax ist nicht die Bedeutung der einzelnen Zeichen und Ausdrücke relevant, sondern vielmehr die Korrektheit der einzelnen Elemente und deren Zusammensetzungen entscheidend. Die Semantik beschreibt aufbauend auf einen syntaktischen Ausdruck, welche Bedeutung den einzelnen Elementen zuzuschreiben ist. Das heißt er stellt die Beziehung zwischen den mathematischen Ausdrücken und die davon abzuleitende Schlussfolgerung der durchzuführenden Operation dar. In der Informatik gibt die Syntax die Struktur an in der ein bestimmter Befehl oder eine Anweisung formuliert werden muss, um später über die Semantik eine entsprechende Bedeutung und ein resultierendes Ergebnis zu bewirken. Syntax und Semantik sind somit für jedes Problem und jede Logik grundlegend, sei es sprachlicher, mathematischer oder informatischer Natur, da es sowohl das Alphabet, resultierende Zusammensetzungen daraus und weiterhin abzuleitende Schlussfolgerungen, durch den Bedeutungen, beinhaltet.

5 Axiomatik

Die Axiomatik hat zum Ziel einen Begriff oder eine Relation eines Gegenstandsbereiches axiomatisch zu erfassen, das heißt sie durch eine Menge von Aussagen zu charakterisieren. Diese Aussagen sind selbstverständlich und bedürfen deshalb keiner Begründung. Sie dienen als Grundlage für eine deduktive Theorie, das heißt es wird vom Allgemeinen auf das Spezielle geschlussfolgert. Dies ist auch der Grund, warum die Aussage nicht selbst durch diese Theorie begründet werden kann. Wenn eine Theorie aus begründeten Sätzen besteht, so muss es notwendigerweise solche Axiome geben, da sonst die Argumentationskette nie enden würde. Zur Verständlichkeit werden im folgenden drei Axiome exemplarisch vorgestellt und kurz kommentiert [1]:

- "Zu jeder Geraden und jedem Punkt, der nicht auf dieser Geraden liegt, gibt es genau eine Parallele durch diesen Punkt" Dieses Axiom der euklidischen Geometrie war immer als weniger klar und einleuchtend erschienen als die anderen; schließlich wurden um die Wende zum 19. Jahrhundert nicht-euklidische Geometrien konzipiert, die zumindest bewiesen, dass es logisch unabhängig ist.

[1] Diese 3 Beispiele wurden wortwörtlich aus der freien Enzyklopädie Wikipedia entnommen. Sie sind im Internet unter http://de.wikipedia.org/wiki/Axiom zu finden.

- “Zu jedem Prädikat P gibt es die Menge aller Dinge, die dieses Prädikat erfüllen.” Dies ist das ursprüngliche Komprehensionsaxiom der Mengenlehre Georg Cantors, das so klar und einfach, so selbstverständlich ist, dass es einen groen Schock bedeutete, als sich herausstellte, dass es nicht widerspruchsfrei zu den anderen Axiomen hinzugefügt werden konnte.
- “Jede natürliche Zahl n hat einen Nachfolger n + 1” ist ein offenbar nicht umstrittenes Axiom(enschema) der Arithmetik. Es ist plausibel, weil es die Zählbewegung simuliert, deren protomathematische Evidenz klar ist.

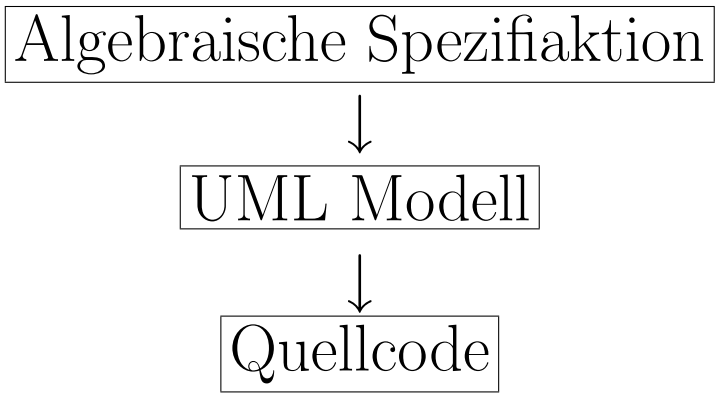
6 Abstrakte Datentypen

Abstrakte Datentypen (ADT) sind äquivalent zu den abstrakten Algebren. Sie bestehen aus folgenden vier Komponenten:

- einem oder mehreren Symbolen für Mengen,
- Symbolen für Operationen,
- Symbolen für Relationen,
- Aussage(formen) über den Symbolen (Eigenschaften und Beziehungen).

Bei den Symbolen für Operation und den Relationen handelt es sich um die Signatur. Die Aussage über die Symbole ist stark verknüpft mit der Axiomatik. Die Abstrahierung, das heißt die Verallgemeinerung, der allgemeinen Algebra geht soweit, daß durch die so entstehenden, übergreifenden Aussagen Einzelfälle ohne neue Überlegungen gelöst werden können. Diese abstrakten Datentypen sind in der Informatik als auch in der Mathematik unabhängig von jedweder Implementierung, das heißt sie sind anwendungsübergreifend.

Abstrakte Datentypen (ADT) werden zunächst mit prädikatenlogischen Formeln beschrieben. Mithilfe dieser Spezifiaktion kann der Datentyp auch ohne Kenntnis der genauen Implementation definiert werden. Diese Darstellung ist zunächst unabhängig von einer speziellen Programmiersprache. Sie kann aber vom Programmierer benutzt werden, um den Dantetyp zu implementieren. Die folgende Graphik zeigt die Reihenfolge bei der Erstellung eines Datentyps:



7 Beispiele Abstrakter Datentypen

7.1 Beispiel: Stack

Der abstrakte Datentyp Stack ist wie folgt spezifiziert:

Typ:	Stack			
Operationen:	empty:			Stack
	push:	int x Stack	→	Stack
	top:	Stack	→	int
	pop:	Stack	→	Stack
	isEmpty:	Stack	→	boolean
Axiome:	isEmpty(empty)	=	true	
	isEmpty(push(x,s))	=	false	
	top(push(x,s))	=	x	
	pop(push(x,s))	=	s	
	pop(empty)	=	error	
	top(empty)	=	error	

Hieraus ergibt sich für die Signatur:

$$\Sigma = (\Omega, \alpha) = (\{empty, push, top, pop, isEmpty\}, \{0, 2, 1, 1, 1\})$$

7.2 Beispiel: String

Der abstrakte Datentyp String ist wie folgt spezifiziert:

Typ:	String			
Operationen:	getChar:	String	→	A
	getString:	String	→	String
	insert:	A x String	→	String
Axiome:	-			

Hieraus ergibt sich für die Signatur:

$$\Sigma = (\Omega, \alpha) = (\{getChar, getString, insert\}, \{1, 1, 2\})$$

7.3 Beispiel: Dictionary

Der abstrakte Datentyp Dictionary[2] ist wie folgt spezifiziert:

Typ:	Dictionary (D), Value (V), Key (K)			
Operationen:	new:		→	D
	find:	K x D	→	V
	insert:	K x D x V	→	D
	delete:	K x D	→	D
Axiome:	delete(K,new())	=	new()	
	find(K,insert(K,V,D))	=	V	

Hieraus ergibt sich für die Signatur:

$$\Sigma = (\Omega, \alpha) = (\{new, find, insert, delete\}, \{0, 2, 3, 2\})$$

[2]Quelle: http://www.nist.gov/dads/HTML/abstractDataType.html

8 Programmverifikation - das Kalkül von Hoare

Das Kalkül von Hoare [3] ist die Grundlage zum Beweis der totalen Korrektheit einer Schleife. Hoare beschrieb die Axiome, die eigentliche Methode zum Beweis der absoluten Korrektheit stammt von Dijkstra [4]. Zunächst einmal müssen wir die Begriffe "Verifikation" und "Überprüfung" (durch Test) voneinander abgrenzen.

> **Test**: zeigt, dass das Programm P die Spezifikation S für die Testmenge T erfüllt.
>
> **Verifikation**: formaler Beweis, der zeigt, dass das Programm P die Spezifikation erfüllt.

Der Hoare-Kalkül bedient sich hierbei der Methoden der Prädikaten- und Aussagenlogik. Die Axiome und Regeln des Hoare-Kalküls werden nacheinander in den Quelltext der Schleife geschrieben. Ergeben sich hieraus keine Widersprüche, so ist bewiesen, dass das Programm unter der Vorbedingung V zur Nachbedingung N führt, das heißt es terminiert.

Der Bezug zur algebraischen Spezifikation besteht darin, dass die Bedingungen, die hierbei herangezogen werden direkt aus der algebraischen Spezifikation entnommen werden können.

Die Verifikation einer einzigen Schleife, welche aus wenigen Zeilen Quellcode besteht ist jedoch schon so umfangreich, dass er hier nicht dargestellt werden kann. Eine einzige Schleife die die z.B. den Rest bei einer Division bestimmt erfordert schon circa 4 Seiten Verifikation. Das Vorgehen bei der Verifikation ist in [2] anhand einiger Beispiele beschrieben.

[3] Charels Antony Richard Hoare: britischer Wissenschaftler. Vor allem bekannt durch den Sortieralgorithmus *Quicksort*

[4] Edsger Wybe Dijkstra: niederländischer Wissenschaftler (1930-220)

9 Zusammenfassung

Zusammenfassend läßt sich sagen, dass die algebraischen Spezifikationen eine Abstrahierung der allgemeinen Algebra sind. Durch diese extreme Verallgemeinerung ist es möglich, viele spezialisierte Probleme ohne nötige Neuüberlegungen zu lösen.

Besonders wichtig ist dieses Gebiet in der Informatik, da hier am ehesten danach gestrebt wird möglichst viele Problemstellungen und Algorithmen mit einem gemeinsamen Wortschatz abarbeiten zu können. Deutlich wird dieser Punkt vor allem bei den höheren Programmiersprachen wie C++ oder Java, bei denen mit einem eher kleinen Alphabet komplexe Sachverhalte gelöst werden können. Um solchen Programmiersprachen den Weg zu ebenen ist es daher von Nöten, sich mit der theoretischen Informatik auseinander zu setzen, was nichts anderes bedeutet als sich mit algebraischen Spezifikationen und abstrakten Datentypen zu beschäftigen.

Die Vorteile von algebraischen Spezifikationen sind vor allem darin zu sehen, dass sie aus einem formalen Verfahren mit mathematischer Semantik und entwickelten Theorien bestehen. Die Unterstützung von Rapid Prototyping ist ebenfalls ein großer Vorteil. Rapid Prototyping bezeichnet die Möglichkeit, Implementationen automatisch aus maschinell ausführbaren Spezifikationen zu generieren. Somit ist ein erster Prototyp eines Produktes mit verhältnismäßig geringem Aufwand aus ersten Entwurfsdaten erstellbar[5]. Die Automatisierbarkeit der Konsistenzprüfung und die mögliche automatische Generierung von Testdaten aus den Spezifikationen, sind ebenfalls zwei erwähnens Werte Vorteile.

Verständlicher Weise ergeben sich aus den algebraischen Spezifiaktionen auch einige Nachteile und Probleme. Algebraische Spezifikationen sind nicht generell verwendbar, dieses wird besonders bei Echtzeitsystemen, die in der Zukunft immer wichtiger werden, deutlich. Graphical User Interfaces (GUI) sind ebenfalls mit diesem System nicht anwendbar, so dass man hierbei auf andere Implementierungen ausweichen muss. Die generelle Anwendbarkeit der algebraischen Spezifikationen vor allem bei groen Problemen ist ebenfalls stark umstritten. Dies ist zurückzuführen auf die oben erwähnte starke Abstraktion, die sehr schwierig und komplex ist und nur bei Projekten bis zu einem gewissen Problemumfang Sinn machen, da man schnell die mathematische Fassbarkeit verlieren kann. Wichtig bei algebraischen Spezifikationen ist, das immer eine eindeutige Logik zu Grunde liegen muss. Aus dem Fakt, das unsere Welt für uns nur bis zu einem gewissen Punkt erklärbar ist, kann man ableiten, das nicht alle Probleme des Alltags auf diese Logik abstrahiert und somit gelöst werden können.

[5] http://de.wikipedia.org/wiki/Rapid_Prototyping

Literatur

[1] Mathias Kegelmann: Allgemeine Algebra für Informatik, im Internet: http://www.mathematik.tu-darmstadt.de/MathNet/Lehrveranstaltungen/Lehrmaterial/WS2002-2003/AllgemeineAlgebra/, 19.04.2004

[2] Michael Gellner: Der Umgang mit dem Hoare-Kalkül zur Programmverifikation, im Internet: http://wwwswt.informatik.uni-rostock.de/deutsch/Mitarbeiter/michael/lehre/pt_2001/Hoare_akt_008.pdf, 19.04.2004

[3] Hans-Jörg Kreowski: Algebraische Spezifikation, im Internet: http://www.informatik.uni-bremen.de/theorie/teach/lehre/AlgSpez/Skript.pdf, 19.04.2004

[4] Markus Hohenwarter: Algorithmen und Datenstrukturen, im Internet: http://informatik.uibk.ac.at/users/c703301/files/algdat01_2f.pdf, 19.04.2004

[5] Martin Klarner: Spezifikation und Implementation, im Internet: http://www8.informatik.uni-erlangen.de/IMMD8/Lectures/WEB/vorlesung/sources/11122002/Vorlesung9.pdf, 26.04.2004

[6] Wikipedia: Axiom, im Internet: http://de.wikipedia.org/wiki/Axiom 26.04.2004

[7] NIST: Abstract Data Type, im Internet: http://www.nist.gov/dads/HTML/abstractDataType.html 26.04.2004

[8] Wikipedia: Deduktion, im Internet: http://de.wikipedia.org/wiki/Deduktion 26.04.2004