

Ausgewählte Kapitel der Software- Technologie: Spezifikation und Testen

WS 2006/07

Lehrstuhl für Angewandte Telematik / e-Business
Fachbereich Mathematik und Informatik der Universität
Leipzig

Volker.Gruhn@informatik.uni-leipzig.de

0341 / 97 323 31

4.5 Terme

- Σ -Grundterm
 - Signatur: Syntax
 - Algebra: Semantik
- Terme: Wörter der Sprache
 - Beschreibt syntaktische Struktur auf Ebene der Signatur
 - Interpretation durch die jeweilige Algebra
 - z.B. `succ(add(n1,n2))`
 - Grundterme: Terme ohne Variablen
 - Terme: Terme mit Variablen

4.5 Terme

- Σ -Grundterm (Definition)

Sei $\Sigma = (S, F)$ eine Signatur.

Die Familie der Mengen $T(\Sigma, S)$ von Σ -Grundtermen besteht für alle $s \in S$ aus der kleinsten Teilmenge $T(\Sigma, S)$ von Wörtern über dem Alphabet $(S \cup F \cup \{“, ”, “, ”\})$,

für die gilt

Konstanten

- $c \in T(\Sigma, S) \forall c: \rightarrow s \in F$

Operations-
symbole +
zusammen-
gesetzte
Terme

- $f(t_1, \dots, t_n) \in T(\Sigma, S)$
 $\forall f: s_1 \dots s_n \rightarrow s \in F$ und alle $t_i \in T(\Sigma, S_i), i \in \{1, \dots, n\}$

4.5 Terme

- Σ -Grundterm (Beispiel)

- Sei gegeben $\Sigma\text{-Nat} = (S, F)$ aus vorherigem Beispiel

- $T(\Sigma\text{-Nat}, \text{Nat})$

- Stufe 0: one
- Stufe 1: succ(one), add(one,one)
- Stufe 2: succ(succ(one)), succ(add(one,one)),
add(succ(one),one), add(one,succ(one)),
add(add(one,one),succ(one)),...
- Stufe 3: succ(succ(succ(one))),...
-

- Vereinfachte Schreibweise per Rekursion

- $T(\Sigma\text{-Nat}, \text{Nat}) = \{\text{one}\} \cup \{\text{succ}(n) | n \in T(\Sigma\text{-Nat}, \text{Nat})\} \cup \{\text{add}(n1, n2) | \{n1, n2\} \subseteq T(\Sigma\text{-Nat}, \text{Nat})\}$

4.5 Terme

- Auswertung von Σ -Grundtermen (Definition)

Sei $\Sigma = (S, F)$ eine Signatur und A eine Σ -Algebra.
Die **Auswertung** (Evaluation/Interpretation) $\text{eval}(A)$
der Terme zur Signatur Σ in A
ist eine Familie von Abbildungen

- $\text{eval}(A) = (\text{eval}(A)_s : T(\Sigma, s) \rightarrow A_s)_{s \in S}$

Sie ist definiert für alle $s \in S$ durch:

Konstanten

Operations-
symbole +
zusammen-
gesetzte
Terme

- $\forall c : \rightarrow s \in F : \text{eval}(A)_s(c) = c_A$
- $\forall f : s_1, \dots, s_n \rightarrow s \in F$ und $\forall t_i \in T(\Sigma, s_i) :$
 $\text{eval}(A)_s(f(t_1, \dots, t_n)) = f_A(\text{eval}(A)_{s_1}(t_1), \dots, \text{eval}(A)_{s_n}(t_n))$

4.5 Terme

- Auswertung von Σ -Grundtermen (Beispiel)

- $\text{eval}(A)_{\text{Nat}}(\text{succ}(\text{add}(\text{one}, \text{one})))$
 $= \text{succ}_A(\text{eval}(A)_{\text{Nat}}(\text{add}(\text{one}, \text{one})))$
 $= \text{succ}_A(\text{add}_A(\text{eval}(A)_{\text{Nat}}(\text{one}), \text{eval}(A)_{\text{Nat}}(\text{one})))$
 $= \text{succ}_A(\text{add}_A(\text{one}_A, \text{one}_A))$
 $= \text{succ}_A(\text{add}_A(1, 1))$
 $= \text{succ}_A(2)$
 $= 3$
- $\text{eval}(A-2)_{\text{Nat}}(\text{succ}(\text{add}(\text{one}, \text{one})))$
 $= \text{succ}_{A-2}(\text{eval}(A-2)_{\text{Nat}}(\text{add}(\text{one}, \text{one})))$
 $= \text{succ}_{A-2}(\text{add}_{A-2}(\text{eval}(A-2)_{\text{Nat}}(\text{one}), \text{eval}(A-2)_{\text{Nat}}(\text{one})))$
 $= \text{succ}_{A-2}(\text{add}_{A-2}(\text{one}_{A-2}, \text{one}_{A-2}))$
 $= \text{succ}_{A-2}(\text{add}_{A-2}(2, 2))$
 $= \text{succ}_{A-2}(4)$
 $= 6$

4.5 Terme

- Signatur mit Variablen (Definition)

- Sei $\Sigma = (S, F)$ eine Signatur.
Eine Familie $X = (X_s)_{s \in S}$ von Mengen,
die durch S indiziert ist, heißt S -sortiert (S -indiziert).
Deren Elemente heißen Variablen.
- $\forall s \in S: X_s \cap F = \{\}$
- Dann heißt $\Sigma = (S, F, X)$ **Signatur mit Variablen**

4.5 Terme

- Allgemeine Terme (Definition)

- Sei $\Sigma = (S, F, X)$ eine Signatur mit Variablen.
Die Familie der Mengen $T(\Sigma, S)(X)$ von (**allgemeinen**) Σ -Termen
besteht für alle $s \in S$ aus der kleinsten Teilmenge $T(\Sigma, S)(X)$ von
Wörtern über dem Alphabet $(S \cup F \cup X \cup \{“, ”, “, ”, “, ”\})$, für die gilt

Variablen

- $x \in T(\Sigma, S)(X) \forall x \in X_s$

Konstanten

- $c \in T(\Sigma, S)(X) \forall c: \rightarrow s \in F$

Operations-
symbole +
zusammen-
gesetze
Terme

- $f(t_1, \dots, t_n) \in T(\Sigma, S)(X)$
 $\forall f: s_1 \dots s_n \rightarrow s \in F$ und alle $t_i \in T(\Sigma, S_i)(X), i \in \{1, \dots, n\}$

- Variablen spielen dieselbe syntaktische Rolle wie
Konstantensymbole, unterscheiden sich jedoch in der Semantik
 - Konstanten haben in Algebra feste Bedeutung
 - Variablen müssen für eine TermAuswertung erst belegt werden!

4.5 Terme

- Auswertung von allgemeinen Σ -Termen (Definition)

- Sei $\Sigma = (S, F, X)$ eine Signatur mit Variablen,
A eine Σ -Algebra und
 $\text{ass}: X \rightarrow A$ eine Variablenbelegung.

Die erweiterte Auswertung $\text{xeval}(\text{ass}): T_{\Sigma}(X) \rightarrow A$ der Terme zur Signatur Σ in A (bezgl. ass) ist eine Familie von Abbildungen

- $(\text{xeval}(\text{ass})_s: T(\Sigma, s)(X) \rightarrow A_s)_{s \in S}$

Sie ist definiert für alle $s \in S$ durch:

Variablen

Konstanten

Operations-
symbole +
zusammen-
gesetzte
Terme

- $\forall x \in X_s : \text{xeval}(\text{ass})_s(x) =_{\text{def}} (\text{ass})_s(x)$
- $\forall c: \rightarrow s \in F : \text{xeval}(\text{ass})_s(c) =_{\text{def}} c_A$
- $\forall f: s_1, \dots, s_n \rightarrow s \in F$ und $\forall t_i \in T(\Sigma, s_i)(X) :$
 $\text{xeval}(\text{ass})_s(f(t_1, \dots, t_n)) =_{\text{def}} f_A(\text{xeval}(\text{ass})_{s_1}(t_1), \dots, \text{xeval}(\text{ass})_{s_n}(t_n))$

4.5 Terme

- Auswertung von allgemeinen Σ -Termen (Beispiel)

- Signatur Σ -Nat, Algebra A
- $X_{\text{Nat}} = \{n_1, n_2\}$
- Variablenbelegung $v: X \rightarrow A$
 - $v_{\text{Nat}}(n_1) = 1909$
 - $v_{\text{Nat}}(n_2) = 5$

- $\text{xeval}(v)_{\text{Nat}}(\text{add}(\text{succ}(0), \text{succ}(n_2)))$
 $= \text{add}_A(\text{xeval}(v)_{\text{Nat}}(\text{succ}(0)), \text{xeval}(v)_{\text{Nat}}(\text{succ}(n_2)))$
 $= \text{add}_A(\text{succ}_A(\text{xeval}(v)_{\text{Nat}}(0)), \text{succ}_A(\text{xeval}(v)_{\text{Nat}}(n_2)))$
 $= \text{add}_A(\text{succ}_A(v_{\text{Nat}}(0)), \text{succ}_A(v_{\text{Nat}}(n_2)))$
 $= \text{add}_A(\text{succ}_A(0), \text{succ}_A(5))$
 $= \text{add}_A(1, 6)$
 $= 7$

Studentische Hilfskräfte gesucht!

- **Projekt:** Entwicklung eines Frameworks zur Steuerung von Dialogen in Web-Anwendungen
- **Aufgaben:** Konzeption, Implementierung von Mechanismen
 - zur Validierung von Dialogspezifikationen
 - zur Steuerung von Datenflüssen
 - zur Behandlung von Backtracking in Web-Anwendungen
- **Voraussetzungen:**
 - notwendig: gutes Vordiplom, Java-Programmierkenntnisse
 - hilfreich: Erfahrung mit XML, Servlets, JSP oder JSF
- Weiterführende Bearbeitung in Abschlussarbeiten möglich
- **Kontakt:** Matthias Book (book@ebus.informatik.uni-leipzig.de)

4.6 Algebraische Spezifikation

- Spezifikation (lat.): Auflistung
 - Auflistung syntaktisch formulierbarer Eigenschaften einer Algebra
- Signaturen repräsentieren die einfachste Form der Spezifikation
- Idee: Wie kann die Interpretation einer Signatur durch eine Algebra als Funktion dargestellt werden?
 - Bildung von Formeln über den Termen einer Signatur
- Spezifikationen erweitern also Signaturen um Formeln, die mit Hilfe von Termen syntaktisch formuliert werden
 - Algebraische Gleichungslogik

4.6 Algebraische Spezifikation

- Σ -Gleichung (Definition) / Grundgleichung
 - Sei A eine Algebra mit der Signatur $\Sigma = (S, F, X)$.
Für $t, t' \in T(\Sigma, X)_{s \in S}$ heißt
 $e \equiv t =_s t'$ eine Σ -Gleichung.
 - e heißt **Grundgleichung**, falls weder t noch t' Variablen enthalten
- Beispiel
 - Signatur Σ -Nat, Σ -Algebra A
 - $\text{eval}(A)_{\text{Nat}}(\text{zero}) = \text{eval}(A)_{\text{Nat}}(\text{add}(\text{zero}, \text{zero}))$ gültig
 - $\text{eval}(A)_{\text{Nat}}(\text{zero}) = \text{eval}(A)_{\text{Nat}}(\text{one})$ ungültig!
 - $\text{xeval}(\text{ass})_{\text{Nat}}(\text{add}(n_1, n_2)) = \text{xeval}(\text{ass})_{\text{Nat}}(\text{add}(n_2, n_1))$ gültig

4.6 Algebraische Spezifikation

- Menge der wohldefinierten Formeln (Definition)
 - Zu einer Algebra mit der Signatur $\Sigma = (S, F, X)$
und einer S -sortierten Variablenmenge X
wird die **Menge der wohldefinierten Formeln** über der Signatur Σ
mit $\text{WFF}(\Sigma)$ bezeichnet.

$\text{WFF}(\Sigma)$ ist die kleinste Menge mit folgenden Eigenschaften:

 1. Jede Σ -Gleichung ist in $\text{WFF}(\Sigma)$.
 2. Mit $G, H \in \text{WFF}(\Sigma)$ sind auch $\neg G, (G \wedge H), (G \vee H) \in \text{WFF}(\Sigma)$.
 3. Falls $x \in X_s, G \in \text{WFF}(\Sigma)$, dann ist $(\forall x: G), (\exists x: G) \in \text{WFF}(\Sigma)$.

4.6 Algebraische Spezifikation

- Menge der wohldefinierten Formeln (Beispiel)
 - Zur Signatur Σ -Nat enthält $WFF(\Sigma\text{-Nat})$
 - jede Σ -Gleichung aus $T(\Sigma\text{-Nat}, S)$, z.B. $\text{add}(\text{zero}, n) = n$
 - Negationen, Disjunktion und Konjunktionen, z.B. $\neg (\text{add}(\text{zero}, n) = n)$
 - $\forall x : \text{add}(\text{zero}, x) = x$

4.6 Algebraische Spezifikation

- Algebraische Spezifikation (Definition)
 - Sei $\Sigma = (S, F, X)$ eine Signatur mit Variablen und E eine Menge von Σ -Gleichungen.

Dann heißt $SP = \langle \Sigma, E \rangle$ **algebraische Spezifikation**, falls $E \subseteq WFF(\Sigma)$.
- Beispiel
 - Algebraische Spezifikation Spec-Nat erweitert Signatur Σ -Nat um die Σ -Gleichungen
 - $\text{equal}_{\text{Nat}}(\text{zero}, \text{zero}) = \text{true}$
 - $\text{equal}_{\text{Nat}}(\text{succ}_{\text{Nat}}(n_1), \text{succ}_{\text{Nat}}(n_2)) = \text{equal}_{\text{Nat}}(n_1, n_2)$
 - $\text{succ}_{\text{Nat}}(n_1) = \text{add}_{\text{Nat}}(n_1, 1)$
 - $\text{succ}_{\text{Nat}}(\text{zero}) = \text{one}$

4.6 Algebraische Spezifikation

- **Modell (Definition)**

- Sei $SP = \langle \Sigma, E \rangle$ eine algebraische Spezifikation und A eine Σ -Algebra.

Dann heißt A **Modell** von SP (SP-Algebra), wenn für alle Σ -Gleichungen gilt

- $A \models e$ (e gültig in A),
 - d.h. $\forall \text{ass}: X \rightarrow A: \text{xeval}(\text{ass})(t) = \text{xeval}(\text{ass})(t')$
- Modelle sind heterogene Algebren mit der gegebenen Signatur, die die gegebenen Formeln erfüllen

4.6 Algebraische Spezifikation

- **Modell (Beispiel)**

- Zur algebraischen Spezifikation $\text{Spec-Nat} = (\Sigma\text{-Nat}, E)$ wird A ein Modell genannt, wenn A alle Formeln aus E erfüllt.
- Beispielhafte Überprüfung anhand eines Beispielaxioms
- $\text{xeval}(v)_{\text{Nat}}(\text{equal}(\text{succ}(n_1), \text{succ}(n_2))) = \text{xeval}(v)_{\text{Nat}}(\text{equal}(n_1, n_2))$
 - $\Leftrightarrow \text{equal}_A(\text{xeval}(v)_{\text{Nat}}(\text{succ}(n_1)), \text{xeval}(v)_{\text{Nat}}(\text{succ}(n_2))) = \text{equal}_A(\text{xeval}(v)_{\text{Nat}}(n_1), \text{xeval}(v)_{\text{Nat}}(n_2))$
 - $\Leftrightarrow \text{equal}_A(\text{succ}_{\text{Nat}}(v_{\text{Nat}}(n_1)), \text{succ}_{\text{Nat}}(v_{\text{Nat}}(n_2))) = \text{equal}_A(v_{\text{Nat}}(n_1), v_{\text{Nat}}(n_2))$
 - $\Leftrightarrow \text{equal}_A(\text{succ}_A(1909), \text{succ}_A(5)) = \text{equal}_A(1909, 5)$
 - $\Leftrightarrow \text{equal}_A(1910, 6) = \text{false}$
 - $\Leftrightarrow \text{false} = \text{false}$
- Also gilt diese Formel für die konkrete Belegung.
- Zu überprüfen bleibt: Alle Formeln für alle Belegungen; funktioniert offensichtlich nicht durch Nachrechnen!

4.6 Algebraische Spezifikation

- **Semantik einer Algebra**

- Alles, was in der Semantik vorkommen soll, kann in der Signatur bezeichnet werden (**no-junk-Prinzip**)
 - Das heißt, eine Algebra, die Elemente in ihren Trägermengen hat, die wir mit Grundtermen nicht bezeichnen können, können wir mit der Signatur nicht „gemeint“ haben, denn sonst hätten wir eine „größere“ Signatur gewählt.
 - Alles was von der Signatur (mittels der Grundterme) nicht erfasst wird, wird als „junk“ bezeichnet.
- Alles, was in der Semantik vorkommen soll, kann eindeutig in der Signatur bezeichnet werden (**no-confusion-Prinzip**)
 - Das heißt, wir interpretieren die durch eine Signatur gegebene Spezifikation minimal. Sie beschreibt nichts „Überflüssiges“. Wäre eine Algebra mit der Signatur „gemeint“ gewesen, die ein Element in einer Trägermenge hat, das Bild zweier verschiedener Grundterme ist, hätte die Signatur „kleiner“ ausfallen sollen.

4.6 Algebraische Spezifikation

- **Semantik einer Algebra**

- Beide Sichten (No-Junk und No-Confusion) sind einschränkend!
- „No Junk“-Sicht verbietet beispielsweise die Spezifikation aller Algebren mit Trägermengen, die mächtiger als Nat sind, denn alle Termengen sind maximal so mächtig wie Nat . Es kann somit keine Signatur geben, die eine surjektive Abbildung eval von der Menge der Grundterme in z.B. die Menge der reellen Zahlen ermöglicht.
- „No Confusion“-Sicht verbietet beispielsweise alle Operationen, die auf den Daten arbeiten, statt sie lediglich zu konstruieren. Nat mit dem üblichen Interpretationen sind demnach z.B. keine „gemeinte“ Algebra zur Signatur $\{\text{nat}\}$, zero , succ , $+$. Denn sowohl der Term zero als auch $\text{zero}+\text{zero}$ bezeichnen beide 0 in der Standardalgebra der natürlichen Zahlen.
- Hinweis: Das Problem der „No Junk“-Sicht ist mit den diskutierten Mitteln nicht lösbar! Das Problem der „No Confusion“-Sicht ist durch die Mittel der Gleichungsspezifikation in den Griff zu bekommen.

4.6 Algebraische Spezifikation

- Semantik einer Algebra

Σ -NatB (ohne +)	Nat	INT	NATMOD3
Nat	N	Z	{0,1,2}
Zero: \rightarrow Nat	0	0	0
Succ: Nat \rightarrow Nat	$n \rightarrow n+1$	$i \rightarrow i+1$	$n \rightarrow (n+1) \bmod 3$
Signatur	Standardalgebra	Junk	Confusion

- INT enthält „junk“, da nicht alle Elemente der Trägermenge $INT_{\text{NAT}} = Z$ von Grundtermen bezeichnet werden können! Alle negativen Zahlen bleiben syntaktisch unbenannt.
- NATMOD3 enthält keinen „junk“, ist aber „confus“, denn verschiedene Grundterme der Signatur werden gleich interpretiert.

4.6 Algebraische Spezifikation

- Semantik einer Algebra

- Die terminale Semantik ist durch das terminale Modell gegeben. Gilt eine Gleichung in einem Modell, so gilt sie auch im terminalen Modell. Das terminale Modell ist somit das maximale Modell.

- initiales Modell terminales Modell

Menge der erfüllten \subseteq \subseteq Menge der erfüllten
Gleichungen Gleichungen

Weiterführende Betrachtungen basieren auf Grundlagen der Kategorientheorie.
Vgl. dazu Kap 8 und 11 in Ehrig, Mahr, Cornelius, Große-Rohde, Zeitz

4.6 Algebraische Spezifikation

- Widersprüchlichkeit und Redundanz
 - Eine Algebra ist widersprüchlich, wenn sich $true = false$ beweisen lässt.
 - Eine Algebra ist redundant, wenn die Menge der beweisbaren Aussagen gleich bleibt auch dann, wenn ein Axiom gestrichen wird.
 - Beispiel: Wenn die „String“-Algebra um das Axiom $append(new(),s) = s$ erweitert wird, dann ist sie redundant, denn dieses Axiom ist ohnehin ableitbar und trägt somit nicht zu zusätzlichen beweisbaren Axiomen bei.

4.6.1 Von Datenstruktur zum Modell

- Datenstruktur ist aus formeller Sicht eine Algebra
 - Datenstruktur besteht aus verschiedenen Datenbereichen und einer Menge von Operationen auf diesen Bereichen
z.B. INT, REAL, NAT, STRING, STACK, QUEUE
1. Aufstellung einer Signatur Σ
 - Sorten (Datenbereiche) und Operationen (Methoden, Funktionen)
 2. Formulieren von Σ (Grund-)Termen
 - Ermöglichen die Bezeichnung der Elemente der Datenbereich
 - „No-junk“-Prinzip: Eine Algebra soll nur Elemente enthalten, die durch Σ -Terme bezeichnet werden können
 - Stellt unsere Signatur genügend syntaktische Ausdrücke bereit, um alle Elemente des Datenbereiches zu bezeichnen?
→ evtl. Signatur erweitern
 - Lassen sich alle Elemente der Implementierung (Algebra) durch die Signatur bezeichnen? → evtl. Algebra einschränken

4.6.1 Von Datenstruktur zum Modell

3. Aufstellen von Gleichungen

- „No-confusion“-Prinzip: Elemente der Datenbereiche lassen sich durch verschiedene Grundterme beschreiben
 - Alles, was nicht explizit als gleich definiert wird, wird verschieden interpretiert
 - Es muss definiert werden, welche Grundterme die gleichen Datenelemente bezeichnen
- Gesucht: eine Menge E von Gleichungen, um die Gleichheit von Elementen zu beschreiben (wegen „no-confusion“-Prinzip),
 - die in der Algebra gültig ist (Korrektheit von E) und
 - die Menge aller im Modell gültigen Grundgleichungen generiert (Vollständigkeit von E)

4.6.1 Von Datenstruktur zum Modell

4. Ableiten von Algebren aus Spezifikation

- Aus Signatur kann eine „Grundtermalgebra“ abgeleitet werden (initiale Semantik)
- Aus Spezifikation (Signatur + Gleichungen) kann eine „Quotiententermalgebra“ abgeleitet werden (initiale Semantik)
- Quotientenalgebra ist idelaerweise isomorph zu unserem Modell (Ausgangs-Algebra)

5. Homomorphismus und Algebra

- Wenn es eine strukturerhaltende Abbildung (Homomorphismus) von einer Quotiententermalgebra auf eine Algebra X gibt, dann entspricht X der Spezifikation
- Strukturelle Induktion als Werkzeug für den Beweis der Existenz einer solchen Abbildung

4.2 Ein Beispiel

Struktur der Spezifikation

algebra X

introduces sorts Y,Z;

operations

op1 : X -> X

op2 : X -> Y

constrains op1, op2 so that for all

x,y : X, z: Y

...

end X;

4.2 Ein Beispiel

algebra String introduces sorts String, Char, Nat, Bool;

operations

new: \rightarrow String

append: String x String \rightarrow String

add: String x Char \rightarrow String

length: String \rightarrow Nat

isEmpty: String \rightarrow Bool

equal: String x String \rightarrow Bool

constrains new, append, add, length, isEmpty, equal so that

for all s,s1,s2: String, c: Char

isEmpty(new()) = true

isEmpty(add (s,c)) = false

...

equal (new(), new()) = true

equal (new(), add(s,c)) = false

equal (add(s1,c),add(s2,c)) = equal (s1,s2)

end X;

4.7 Zusammenfassung und Ausblick

Signatur $\Sigma = (S, F)$

S-sortierte Variablenmenge

Σ -Algebra

Σ -Terme

Grundterm

Σ -Gleichung

Interpretation

Formel/WFF

Σ -Homomorphismus

algebraische
Spezifikation

Modell einer algebraischen Spezifikation

Semantik

initiale/terminale Semantik

4.7 Zusammenfassung und Ausblick

- Forschungsgegenstände

- Systematische Ermittlung der minimalen Menge der generierenden Operationen
- Ermittlung der nötigen Axiome für die gewünschte / beabsichtigte Spezifikation

Anhaltspunkt:

oi_1, \dots, oi_n als Menge der inspizierenden Operationen

og_1, \dots, og_m als Menge der generierenden Operationen

Axiome: $oi_i(og_j(\dots)) = \dots \quad \forall i = 1, \dots, n \quad \forall j = 1, \dots, m$

- Automatische Identifikation von Widersprüchen
- Automatische Identifikation von Redundanz
- Von der Spezifikation zum Programm ??

4.7 Zusammenfassung und Ausblick

- Vorteile der algebraischen Spezifikation
 - Spezifikation weitestgehend implementierungsunabhängig
 - Vollständig formale Spezifikation, daher für Verifikation und automatische Werkzeuge einsetzbar
 - Theoretisch fundiert
 - Kann auf abstrakte Datenobjekte, Datentypen und Klassen angewandt werden
 - Wirkung von Zugriffsoperationen ergibt sich durch wechselseitige statische Abhängigkeiten, nicht durch isolierte Betrachtung
 - Mit einiger Erfahrung kann man algebraische Spezifikationen lesen
 - Leicht erweiterbar, da zusätzliche Operationen nur jeweils eine neue Syntaxregel sowie zusätzliche Axiome aufbauend auf Grundoperationen erfordern

4.7 Zusammenfassung und Ausblick

- Nachteile der algebraischen Spezifikation
 - Konstruktion algebraischer Spezifikationen nicht trivial
 - Fälle wie z.B. Grenzbedingungen können leicht übersehen werden
 - Schwierig auf Konsistenz und Vollständigkeit zu prüfen
 - Operationen mit mehreren Wertebereichen sind schwierig zu beschreiben
- In der Praxis hat sich die algebraische Spezifikation nicht durchgesetzt!

4.7 Zusammenfassung und Ausblick

- Warum algebraische Spezifikation, modellbasierte Spezifikation und formale Spezifikation im allgemeinen?
 - Der Spezifizierer muss Basistechniken und Sprachen kennen
 - Formale Spezifikation sind eine bedeutsame Klasse in der Menge aller Spezifikationssprachen
 - Man muss wissen, was man mit formaler Spezifikation erreichen kann UND was nicht!

4.8 Literatur

- Ehrig, Mahr: Fundamentals of algebraic Specification 1: Equations and Initial Semantics, Springer Verlag, 1985
- Ehrig, Mahr: Fundamentals of algebraic Specification 2: Module Specifications and Constraints, Springer Verlag, 1990
- Ehrig, Mahr, Cornelius et.al.: Mathematisch-strukturelle Grundlagen der Informatik, Springer Verlag, 1999
- Ehrich, Gogolla, Lipeck: Algebraische Spezifikation abstrakter Datentypen, Teubner Verlag, 1989