



HOCHSCHULE  
KONSTANZ  
TECHNIK, WIRTSCHAFT  
UND GESTALTUNG

FORMALE METHODEN DER  
SOFTWAREMODELLIERUNG

# Modellierung

ZUSAMMENFASSUNG: GRUNDLAGEN UND FOMALE  
METHODEN

*Ismail Zeybek*

June 17, 2019

---

# Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>3</b>
<b>1 Einführung in die Modellierung</b>	<b>4</b>
<b>2 Software</b>	<b>5</b>
2.1 Probleme bei Software-Entwicklung . . . . .	7
2.2 Definitionen System . . . . .	7
2.3 Beispiele: Modellierung im Alltag . . . . .	8
2.4 Aspekte eines Modells . . . . .	9
<b>3 Algebraische Spezifikation</b>	<b>10</b>
3.1 Modellierung von Datenstrukturen . . . . .	11
3.2 Relationen und Funktionen . . . . .	15
3.3 Algebraische Spezifikation von Datenstrukturen . . . . .	19
<b>4 Logik</b>	<b>23</b>
4.1 Relationen und Funktionen . . . . .	24
<b>5 Anhang</b>	<b>32</b>

## Abbildungsverzeichnis

1	The Need for Change . . . . .	6
2	CycleOfDemand . . . . .	6
3	Projektion Model . . . . .	8
4	Abtraktion in der Modellierung . . . . .	8
5	Abtraktion von Gebäuden . . . . .	9
6	links Intesion, rechts Extension . . . . .	12
7	Mengenoperationen . . . . .	12
8	3 Möglichkeiten . . . . .	15
9	Zweistellige Relationen - Eigenschaften . . . . .	16
10	Funktionen - Eigenschaften . . . . .	17
11	Sweet and white button . . . . .	22
12	Bewertung von Formeln . . . . .	25
13	Wahrheitstafel . . . . .	25
14	Umformungsregeln . . . . .	26
15	Weitere Umformungsregeln . . . . .	26
16	Wahrheitstafel . . . . .	27
17	Es regnet Beispiel . . . . .	28
18	Es regnet Beispiel . . . . .	28
19	Beispiel . . . . .	29
20	Weitere Beispiele . . . . .	29
21	Wirkungsbereich von Quantoren . . . . .	30

# 1 Einführung in die Modellierung

Das Modellieren ist eine für das Fach Informatik typische Arbeitsmethode, die in allen Gebieten des Faches angewandt wird. Aufgaben, Probleme oder Strukturen werden untersucht und als Ganzes oder in Teilespekten beschrieben, bevor sie durch den Entwurf von Software-Algorithmen, Daten oder Hardware gelöst bzw. implementiert werden. Mit der Modellierung einer Aufgabe zeigt man, ob und wie sie verstanden wurde. Das Modell ist Voraussetzung und Maßstab für die Lösungen und liefert meist auch den Schlüssel für einen systematischen Entwurf. Als Ausdrucksmittel für die Modellierung steht ein breites Spektrum von Kalkülen und Notationen zu Verfügung. Sie sind spezifisch für unterschiedliche Arten von Aufgaben und Problemen. Deshalb werden in den verschiedenen Gebieten der Informatik unterschiedliche Modellierungsmethoden eingesetzt. In den entwurfsorientierten Gebieten, wie Software-Technik und Hardware-Entwurf, ist die Bedeutung der Modellierung und die Vielfalt der Methoden besonders stark ausgeprägt. Mit dieser Zusammenfassung soll eine Übersicht über die Kalküle der Informatik und ein grundlegendes Verständnis für jeden der vorgestellten Kalküle in Kurzfassung vermittelt werden. Praktische Beispiele zu den jeweiligen Themen wurden zur Übersichtshaber weggelassen. Nichtsdestotrotz werden zu gewissen Themengebiete Beispiele in dem Vorlesungsskript vermerkt.

Dass angebotene Spektrum von Kalkülen ist als Grundausstattung zu verstehen, die noch wesentlich vertieft und verbreitert werden kann. Deshalb wird hier von jedem Kalkül nur der innere methodische Kern präsentiert und auf die Vorstellung von Erweiterungen und weniger grundlegenden Kalkülen verzichtet. Durch das Buch sollen Sie lediglich den Nutzen von klaren und präzisen Beschreibungen erkennen.

Außerdem sollte diese Zusammenfassung als eine Brücke zwischen dem Buch "Modellierung Grundlagen und formale Methoden" und der Vorlesung Formale Methoden der Softwaremodellierung betrachtet werden.

Die Zusammenfassung ist folgendermaßen gegliedert: in der...

- **Einführung** werden Problematiken und die Modellbildung im Allgemeinen gezeigt
- **Modellierung mit Datenstrukturen** die Grundlagen der abstrakten und eindeutigen Schreibweise erklärt
- **Relationen und Funktionen** die Beziehungen von zwei Mengen und deren Abbildung erklärt
- **Algebraische Spezifikation von Datenstrukturen** verschiedene Algebren aufgelistet
- –Logik–
- **Aussagenlogik in der Spezifikation** verschiedene Regeln der Aussagenlogik in Bezug auf die Logik gezeigt.
- **Aussagenlogische Modellbildung** werden Modellierung von statischem Wissen erklärt
- **Prädikatenlogik** die Erweiterung um parametrisierte Elementaraussagen beigebracht
- –Zeichenfolgen–
- **Reguläre Ausdrücke**
- **Bakus Naur Form** (Kontextfreie Grammatik)

## 2 Software

Software bestimmt fast alle unsere Lebensbereiche und die Abhängigkeiten steigen exponentiell weiter, doch neben der rasant wachsenden Nachfrage müssen Entwickler gleichzeitig diese Nachfragen schnell wie möglich abdecken, wobei wir hier nur von der Konstruktiven Nachfrage sprechen. Dennoch werden diese Aufträge als Programmierer aufgrund des Zeitdrucks häufig nicht richtig bearbeitet bzw. falsch Programmiert, dessen Konsequenzen häufige Qualitätsmängel sind, die sogar einen Flugzeugabsturz hervorbringen können.

Häufige Qualitätsmängel bei Software sind:

- Systeme sind selten fehlerfrei
- Fehler in Software können Katastrophen als Auswirkung haben, z.B. Flugzeugabsturz

Programmieren ohne Modelle unter Zeitdruck hat zur Folge, dass man nicht nur den Überblick über das Projekt verlieren kann sondern auch bei "Bugs" häufig mehr Zeit in Anspruch nimmt den Fehler zu finden bzw. zu beheben. Wenn sie modellieren haben sie die gesamte Übersicht über die zu lösende Aufgabe und können eventuell später auftretende Probleme frühzeitig erkennen und von Anfang an alle Karten richtig auf den Tisch legen. Früher oder später werden sie auf diese Probleme zustoßen und erkennen das dass los programmieren in eine Sackgasse führt. Um dies zu vermeiden müssen Probleme bzw. kritische Bereiche frühzeitig mithilfe von Modellierungswerkzeugen erkannt werden, welche wir in dieser Zusammenfassung uns beibringen werden, aber dazu kommen wir erst später.

## Untersuchung der Fehlerquellen

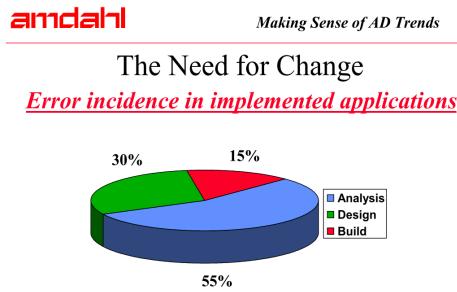


Abbildung 1: The Need for Change

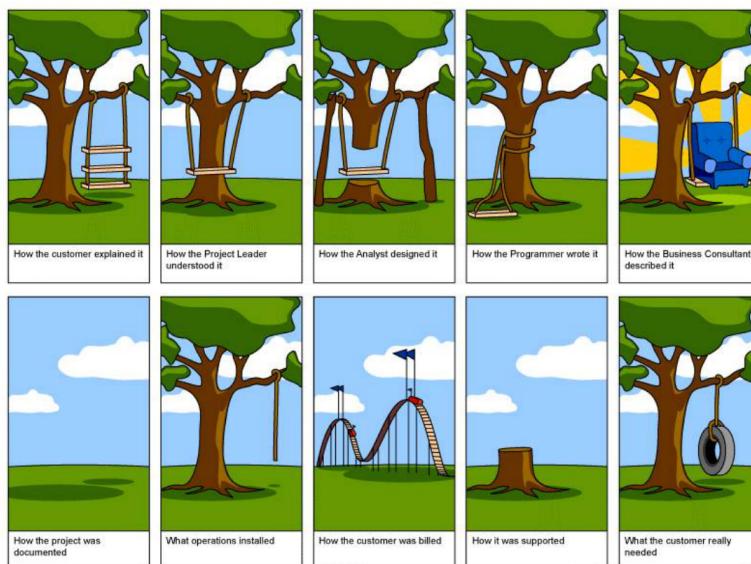


Abbildung 2: CycleOfDemand

Oben ist ein Graph abgebildet, dass den tatsächlichen Projekt-Aufwand in drei Gebiete unterteilt: Analysis, Design und Build. Man kann hier erkennen, dass der Analysis Abschnitt deutlich mehr Aufwand vorzeigt als die anderen beiden zusammen. Um dies zu vermeiden sollte man hauptsächlich mehr Zeit für den Design einplanen und daraus das Projekt quasi "Bauen". Zudem sollte dieser Bereich mit Ruhe und Zeit genossen werden, um später keine Zeit in der "Fehlersuche-Phase" zu verschwenden. Wer also weniger Debuggen möchte sollte sich erst einmal die Basics der Modellierung beibringen, die glücklicherweise in diesem Skript Zusammenfasst sind.

Ein weiter wichtiger Punkt ist es zu Wissen, dass durch die richtige Anwendung der "Modellierung" trotzdem zu Zeitverschwendungen kommen kann, falls die Kommunikation zwischen Kunden, Manager, Projektleiter und Entwickler nicht ernst genommen wird. Es kommen dadurch zu Missverständnissen, wie es in der Abbildung "CycleOfDemand" vorgezeigt wird.

## 2.1 Probleme bei Software-Entwicklung

- Verständigungsschwierigkeiten zwischen Entwicklern und Anwendern werden
  - nicht systematisch erfasst,
  - sind selbst Benutzern unbekannt,
  - und ändern sich regelmässig
- Benutzeranforderungen werden häufig
  - I can't tell you what I want, but I'll know it when I see it
- Folge
  - Systeme erfüllen Anforderungen nicht
  - werden zu spät fertig gestellt,
  - sind zu teuer
- Natürlichsprachliche Beschreibung oft mit geringer Qualität
  - Widerspruchsvoll
  - Unvollständig

## 2.2 Definitionen System

- Definition
  - Der Begriff System umschreibt eine Realität mit allen für den Untersuchungszweck relevanten Wechselwirkungen zwischen ihren Bestandteilen
- Klassifikation von Systemen
  - statisch / dynamisch
  - ideell / real
  - deterministisch / stochastisch

## 2.3 Beispiele: Modellierung im Alltag

- Definition "Modell" nach Helmut Balzert

- Modell als idealisierte, vereinfachte, eines Gegenstands, Systems oder sonstigen Weltausschnitts
- Ziel: bestimmte Eigenschaften des Vorbilds zu studieren

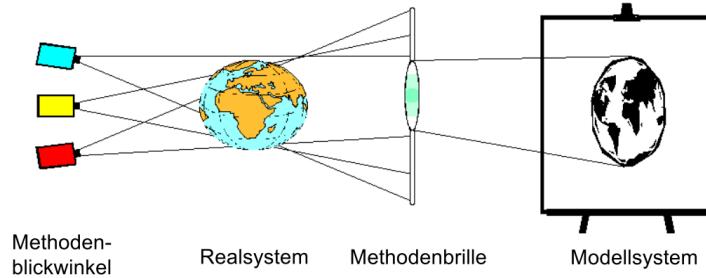


Abbildung 3: Projektion Model

Dazu gibt es auch noch die Abstraktion in der Modellierung, wie zum Beispiel bei einem Busplan. Hier kommt es lediglich darauf an zu wissen wie die Linien miteinander verbunden sind. Jegliche Gebäude oder Straßen sind hier überflüssig, es soll ja nur dem Zweck bestmöglich dienen. Zu diesem Beispiel () kann man zur Veranschaulichung zusätzlich eine Tabelle erstellen, um extra die Fahrzeiten zu verdeutlichen. Also geht es hierbei nur um ein Teilespekt eines Models.

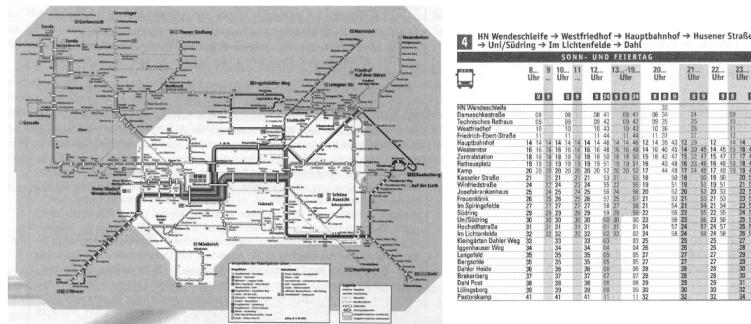


Abbildung 4: Abstraktion in der Modellierung

Das gleiche Prinzip lässt sich auf vieles anwenden. Eines davon wäre die Konstruktion einer Gebäude.



Abbildung 5: Abstraktion von Gebäuden

## 2.4 Aspekte eines Modells

- Verwendungszweck eines Modells bestimmt die Art des Modells
- Beispiel: Mögliche Verwendungszwecke eines Hausbaus
  - Gebäudemodell: optischer Eindruck
  - Grundriss: Einteilung des Grundstücks und Räume
  - Kostenplan: Finanzierung
  - Bauplan: Bauabwicklung
- Verwendete Methoden und Kalküle in der Modellierung
  - Struktur: Wertebereiche, Entity-Relationship, Klassifikation, Typen
  - Eigenschaften: Logik, Relationen
  - Beziehungen: Graphen, Relationen, Logik, Entity-Relationship
  - Verhalten: endliche Automaten, Petrinetze, Algebren, Graphen

Die oberen vier Aspekte sind besonders wichtig, da sie die Grundstruktur eines Modells vorgeben. Es ist nur eine Vorgabe, also sprechen wir hier nur von Schablonen, die man verwenden kann aber nicht unbedingt gezwungen wird.

Hierzu kann man das Einführungs-Beispiel - endlicher Automat mit Übergängen auf der Seite 18 (1. Grundlagen) sich anschauen. (\*\*)

### 3 Algebraische Spezifikation

In diesem Kapitel geht es um eindeutige Aussagen über Datenstrukturen ohne diese Schriftlich beschreiben zu müssen. Um diese Kenntnisse erlangen zu können werden wir die Mengenlehre und Wertebereiche, deren Operationen sowie Potenzmengen, das Kartesische Produkt, Endliche Folgen, Relationen, Funktionen und zum Schluss die Modellierung mit Wertebereichen kennenlernen. Außerdem sollte man im Klaren sein, dass diese Dokumentation einer Zusammenfassung ähneln soll und deshalb in diesem Kapitel vereinfachte Definitionen genannt und dargestellt wird. Um das Thema zu Strukturen werden die oben genannte Themen in 3 Kategorien unterteilt: Modellierung von Datenstrukturen, Relationen und Funktionen und Algebraische Spezifikation von Datenstrukturen. Wir fangen mit dem Thema Modellierung von Datenstrukturen an und wenden uns der Abstraktion von Datenstrukturen.

### 3.1 Modellierung von Datenstrukturen

Die Einführung für dieses Thema finden Sie im Anhang "2. Algebraische Spezifikation 2"().

- **Wertebereich**

- Ein Wertebereich ist eine Menge von Werten, die im Sinne eines Modells als gleichartig angesehen werden
- Wo ein Wert eines Wertebereichs  $W$  gefordert wird, kann prinzipiell jedes Element aus  $W$  diese Rolle übernehmen

- **Menge**

- Mengen zur Modellierung von Wertebereichen.
- Eine Menge  $M$  ist eine Zusammenfassung von verschiedenen Objekten, den Elementen der Menge.
- $a$  ist ein Element der Menge  $M$ .
- Notation:  $a \in M$
- Die Elemente in einer Menge sind nicht geordnet.

### Definition von Typen:

- Intensionale Sicht
  - Definition durch Angabe einer Bedingung, die alle Elemente erfüllen
  - Beispiel:  $a \in \text{IN}$ ,  $a$  ist Quadratzahl und  $a < 30$
- Extensionale Sicht
  - Definition als Auflistung von Elementen
  - Beispiel: 1, 4, 9, 16, 25

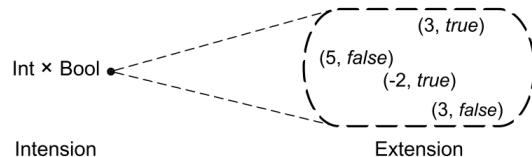


Abbildung 6: links Intension, rechts Extension

### • Mengenoperationen

Bezeichnung	Notation	Bedeutung
Ist Teilmenge	$M \subseteq N$	Aus $a \in M$ folgt $a \in N$
Ist echte Teilmenge	$M \subset N$	$M \subseteq N$ und $M \neq N$
Vereinigung	$M \cup N$	$\{x \mid x \in M \text{ oder } x \in N\}$
Durchschnitt	$M \cap N$	$\{x \mid x \in M \text{ und } x \in N\}$
Differenz	$M \setminus N$	$\{x \mid x \in M \text{ und } x \notin N\}$

Abbildung 7: Mengenoperationen

### • Definitionen

- Zwei Mengen  $M$  und  $N$  sind disjunkt, wenn gilt:  $M \cap N = \emptyset$
- Die Anzahl der Elemente einer Menge  $M$  heißt ihre Kardinalität, notiert als  $|M|$

- **Definition Potenzmenge**

- Die Potenzmenge einer Grundmenge  $U$  ist die Menge aller Teilmenge von  $U$ , geschrieben  $\text{Pow}(U)$  oder  $\text{IP}(U)$
- Als Formel:  $\text{IP}(U) = M \mid M \subseteq U$

- **Beispiel**

- BeigabenArten = {Milch, Zucker}
- Beigaben =  $\text{IP}(\text{BeigabenArten})$

### Kartesisches Produkt // Beispiel unter 2. Algebraische Spezifikation 8

- Definition kartesisches Produkt oder geordnetes Paar oder Kreuzprodukt

- Ein geordnetes Paar  $(x, y)$  besteht aus zwei Werten  $x$  und  $y$ , wobei  $x$  die erste und  $y$  die zweite Komponente ist.
- Das kartesische Produkt  $M \times N$  zweier Mengen  $M$  und  $N$  ist die Menge aller geordneten Paare mit erster Komponente aus  $M$  und zweiter Komponente aus  $N$
- In Formeln:  $M \times N = (x, y) \mid x \in M \text{ und } y \in N$

### Verallgemeinerung kartesisches Produkt

- Definition kartesisches Produkt mit  $n > 1$  Mengen, als Menge von geordneten  $n$ -Tupeln

- $M_1 \times M_2 \times \dots \times M_n = \{ (a_1, a_2, \dots, a_n) \mid a_i \in M_i \text{ und } i \in I \}$  mit Indexmenge  $I = 1, \dots, n$  und nichtleeren  $M_i$ .

- Wenn alle Komponenten aus dem selben Wertebereich kommen:

- $M^n = M \times M \times \dots \times M$

- Modellieren Sie ein Lottoergebnis. Dieses besteht aus sechs verschiedenen Zahlem zwischen 1 und 49

- Lottoergebnis =  $\{x \mid x \in \text{IP}(\{1, 2, \dots, 49\}), |x| = 6\}$

Weitere Beispiele sind auf der Folie 2. Algebraische Spezifikation 10 zu finden ()

### Vereinigung

- Verwendung

- Bilder = {Bube, Dame, König, Ass} Zahlwerte = {7, 8, 9, 10} KartenSymbole = Bilder  $\cup$  ZahlWerte
- Kunde = {Siemens, VW, Bosch} Lieferant = {Siemens, Metabo} Geschäftspartner = Kunde  $\cup$  Lieferant

### Endliche Folgen

- Folge von Elementen können eine unterschiedliche Länge haben
- Definition endliche Folge
  - Ein n-Tupel aus  $A^n$  mit  $n > 1$  Komponenten aus der Menge A heißt Folge der Länge n über A
  - (a), mit  $a \in A$  ist eine Folge der Länge 1 über A
  - oder  $e$  steht für die leere Folge
  - Definition Wertebereich der endlichen nicht-leeren Folgen über A als  $A^+ = \{(a) \mid a \in A\} \cup \{x \mid x \in A^i \text{ und } i > 1\}$
  - Definition Wertebereich der endlichen Folgen über A:  $A^* = \{e\} \cup A^+$

## 3.2 Relationen und Funktionen

### Relationen – Einführung

- Relationen setzen Elemente aus unterschiedlichen Wertebereichen zueinander in Beziehung
- Beispiel 1
  - Personen = {Peter, Gaby, Jens, Inge} Vorlesung = {Mathe1, Mathe2, SOMO, Prog1}
  - Wie lässt sich die Beziehung modellieren, wer welche Vorlesung besucht?
- Beispiel 2
  - Peter ist älter als Gaby, Gaby ist älter als Jens, Jens ist älter als Inge.
  - Wie kann dieser Sachverhalt dargestellt werden?
- Definition
  - Eine n-stellige Relation R ist eine Menge von n-Tupel, wobei jedes davon aus dem Wertebereich  $M_1 \times M_2 \times \dots \times M_n$  mit  $n > 1$  stammt
  - d.h.  $R \subseteq M_1 \times M_2 \times \dots \times M_n$
  - R stammt aus dem Wertebereich  $IP(M_1 \times M_2 \times \dots \times M_n)$
  - Eine einstellige Relation über der Menge M ist eine Teilmenge von M
- Beispiel
  - $A = \{a, b\}$ ,  $B = \{1, 2\}$
  - Wie ist  $A \times B$  definiert, wie ist  $IP$  von  $A \times B$  definiert, wie viele Elemente hat diese Potenzmenge?
- Beispielrelation
  - $A = \{1, 2, 3\}$   $B = \{a, b, c\}$   $R \subseteq A \times B$
  - $R = \{(1, a), (2, b), (2, c), (3, a)\}$
- Darstellung als Matrix

	a	b	c
1	x		
2		x	x
3	x		

- Darstellung als Graph

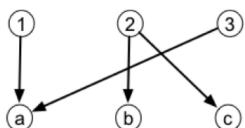


Abbildung 8: 3 Möglichkeiten

Weitere Beispiele sind auf der Folie 2. Algebraische Spezifikation 17 zu finden()

## Zweistellige Relationen - Eigenschaften

- Eigenschaften

reflexiv	Wenn für alle $x \in M$ gilt: $x R x$
irreflexiv	Wenn für alle $x \in M$ gilt: $x R x$ gilt nicht
symmetrisch	Wenn für alle $x, y \in M$ gilt: aus $x R y$ folgt $y R x$
antisymmetrisch	Wenn für alle $x, y \in M$ gilt: aus $x R y$ und $y R x$ folgt $x = y$
asymmetrisch	Wenn für alle $x, y \in M$ gilt: aus $x R y$ folgt $y R x$ gilt nicht
transitiv	Wenn für alle $x, y, z \in M$ gilt: aus $x R y$ und $y R z$ folgt $x R z$

Abbildung 9: Zweistellige Relationen - Eigenschaften

## Äquivalenz- und Ordnungsrelationen

- Definition Äquivalenzrelation
  - Eine zweistellige Relation  $R \subseteq M \times M$  ist eine Äquivalenzrelation, wenn sie reflexiv, symmetrisch und transitiv ist
- Definition Ordnungsrelation
  - Eine zweistellige Relation  $R \subseteq M \times M$  ist eine partielle Ordnung oder Halbordnung, wenn  $R$  reflexiv, antisymmetrisch und transitiv ist
  - eine strenge Ordnung oder strenge Halbordnung, wenn  $R$  irreflexiv und transitiv ist

## Funktionen

- Definition Funktion
  - Eine Funktion  $f$  ist eine 2-stellige Relation  $f \in IP(D \times B)$  für die gilt: aus  $(x, y) \in f$  und  $(x, z) \in f$  folgt  $y = z$
  - Einem Wert aus  $D$  ist also höchstens ein Wert aus  $B$  zugeordnet
  - Die Menge  $D$  heißt Definitionsbereich, die Menge  $B$  Bildbereich oder Wertebereich der Funktion  $f$
  - Schreibweisen:  $(x, y) \in f$  oder  $y = f(x)$  – Signatur einer Funktion:  $f: D \rightarrow B$
  - Signatur einer Funktion:  $f: D \rightarrow B$
- Beispiel 1
  - Funktion, die Nationen auf Ihre Einwohnerzahlen in Millionen abbildet:  $EinwohnerMio = \{(Deutschland, 82), (Frankreich, 58), (\ddot{O}sterreich, 8), (Spanien, 39)\}$
  - $EinwohnerMio : L\ddot{a}nder \rightarrow N^+, EinwohnerMio(Deutschland) = 82$

Weitere Beispiele sind auf der Folie 2. Algebraische Spezifikation 22 zu finden ()

## Multimengen

- Modellierung von Münzen in einem Geldbeutel
- Problem: In einem Geldbeutel können Münzen mehrfach vorkommen
- Lösung 1: Modellierung der Anzahl wie oft Münze vorkommt
  - EuroMünzen = {1, 2, 5, 10, 20, 50, 100, 200}
  - Funktion EuroMünzen –  $\rightarrow IN_0$
  - MeinGeldBeutel = {(1,3), (2,0), (5,0), (10,2), (20,0), (50,1), (100, 4), (200,2)}
- Modellierung mit Multimengen (engl. bags)
  - {1,1,1,10,10,50,100,100,100,200,200}
  - Verwendung in vielen Datenstrukturen von Programmiersprachen (bag)

## Indexmenge

- Problem: manchmal möchte man Elementen einer Menge eine Position zuordnen
- Beispiel
  - F := {w,e,l,l,e}
  - Indexmenge Positionen = {1,...,5}
- Definition einer Funktion F positionen –  $\rightarrow F$ 
  - F auftreten := {(1,w),(2,e),(3,l),(4,l),(5,e)}
  - Verwendung als Array in Programmiersprachen
  - In vielen Programmiersprachen wird in der Indexmenge mit 0 begonnen zu zählen

## Funktionen – Eigenschaften

- Sei Funktion  $f \in D -> B$

total	Quasi eine Zuweisung	Wenn es für jedes $x \in D$ ein Paar $(x, y) \in f$ gibt	
surjektiv		Wenn es zu jedem $y \in B$ ein Paar $(x, y) \in f$ gibt	
injektiv		Wenn für zu jedem $y \in B$ höchstens ein Paar $(x, y) \in f$ gibt	
bijektiv		Wenn $f$ zugleich surjektiv und injektiv ist	

Abbildung 10: Funktionen - Eigenschaften

## Modellierung mit Wertebereichen

- Hinweise
  - Erst Grundmengen festlegen, dann Strukturen darüber bilden
  - Typische Elemente eines Wertebereichs angeben
  - Wertebereiche ausdruckskräftige Namen geben
  - Zusammengesetzte Wertebereiche schrittweise aufbauen
  - Nur gleichartige Elemente in einem Wertebereich
  - Mengen, Tupel und Folgen beliebiger Länge nicht verwechseln
  - Alle Klammern haben Bedeutung - zusätzliche verändern das Modell
- Häufige Funktionen für Mengen
  - Zählen –  $> IN_0$
  - Messen –  $> IR$
  - Entscheiden –  $> \{true, false\}$

Weitere Beispiele sind auf der Folie 2. Algebraische Spezifikation 27 zu finden ()

### 3.3 Algebraische Spezifikation von Datenstrukturen

#### Terme, Sorten und Signaturen

- Term
  - Ausdruck in der Mathematik, der Variablen, Zahlen, Verknüpfungen, Klammern beinhaltet kann
- Operatorbeschreibungen
  - $+: \text{Zahl} \times \text{Zahl} \rightarrow \text{Zahl}$
  - $<: \text{Zahl} \times \text{Zahl} \rightarrow \text{Bool}$
  - $\wedge: \text{Bool} \times \text{Bool} \rightarrow \text{Bool}$
  - $\text{true}: \rightarrow \text{Bool}$
  - $1: \rightarrow \text{Zahl}$
- Sorte
  - Disjunkte Teilmenge der Terme, die durch Operatoren gebildet wird
- Definition Stelligkeit
  - Eine Operator ist  $n$ -stellig mit  $n \geq 0$ , wenn er  $n$  Operanden hat. – 0-stellige Operatoren sind Konstanten

#### Algebren

- Definition Signatur
  - Eine Signatur  $\Sigma = (S, F)$  beschreibt eine Menge von Sorten und eine Menge von Strukturbeschreibungen  $F$
  - $\text{op}: s_1 \times \dots \times s_n \rightarrow s_o$  mit  $s_i \in S$
  - Beschreibung einer Funktion durch die Signatur  $D \rightarrow B$
  - Oder kurz:  $f: D \rightarrow B$
- Abstrakte Algebra
  - Eine abstrakte Algebra  $A = (\tau, \Sigma, Q)$  ist definiert durch die Signatur  $\Sigma$ , die Menge der korrekten Terme  $\tau$  zu  $\Sigma$  und eine Menge von Axiomen (Gesetze)  $Q$ .
  - Ein Axiom hat die Form  $t_1 \rightarrow t_2$ , wobei  $t_1$  und  $t_2$  korrekte Terme gleicher Sorte sind und Variablen enthalten können.
- Konkrete Algebra
  - Zuordnung konkreter Wertebereiche

## Abstrakte boolsche Algebra

- Signatur
  - Signatur  $\Sigma = (S, F)$ ,  $S = \{\text{BOOL}\}$  Operationen  $F$ :
  - $\text{true}: - \rightarrow \text{BOOL}$
  - $\text{false}: - \rightarrow \text{BOOL}$
  - $\neg: \text{BOOL} - \rightarrow \text{BOOL}$
  - $\wedge: \text{BOOL} \times \text{BOOL} - \rightarrow \text{BOOL}$
  - $\vee: \text{BOOL} \times \text{BOOL} - \rightarrow \text{BOOL}$
  - Konstante true und false als „Grenzfall“ von Funktionen
- Axiome Q: für alle  $x, y$  der Sorte BOOL gilt:
  - Q1:  $\neg\text{true} = \text{false}$
  - Q2:  $\text{false} = \text{true}$
  - Q3:  $\text{true} \wedge x = x$
  - Q4:  $\text{false} \wedge x = \text{false}$
  - Q5:  $x \wedge y = \neg(\neg x \vee \neg y)$

## Algebraische Spezifikation von Datenstrukturen

- Datenstruktur Stack (Keller)
  - Datenstruktur zum Einfügen und Entfernen von Elementen
  - Last-In-First-Out Prinzip
  - Anwendung z.B. bei Implementierung von Programmiersprachen
- Operationen
  - `createStack`: liefert einen leeren Stack
  - `push`: fügt ein Element in den Stack ein  
gibt an, ob Stack leer ist
  - `pop`: entfernt das zuletzt eingefügte Element
  - `top`: liefert das zuletzt eingefügte, nicht gelöschte Element
  - `empty`: gibt an, ob Stack leer ist
- Beispiele
  - `push( push( push( createStack, 1), 2), 3)`
  - `push( pop( push( push( createStack, 1), 2)), 3)`

## Spezifikation Stack

- Definition

- Signatur  $\Sigma = (S, F)$
- Sorten S: {Stack, Element, BOOL}
- createStack:  $- \rightarrow \text{Stack}$
- push:  $\text{Stack} \times \text{Element} \rightarrow \text{Stack}$
- pop:  $\text{Stack} \rightarrow \text{Stack}$
- top:  $\text{Stack} \rightarrow \text{Element}$
- empty:  $\text{Stack} \rightarrow \text{BOOL}$

- Axiome

- K1: empty(createStack) = true
- K2: empty(push(k, t)) = false
- K3: pop(push(k, t)) = k
- K4: top(push(k, t)) = t

Das Prinzip lässt sich auch auf Natürliche Zahlen und Mengen anwenden. Es hängt lediglich davon ab welche Signatur die Spezifikation trägt und auf was sie angewendet wird. Die vorhin genannten zwei Anwendungsmöglichkeiten werden in der Folie 2. Algebraische Spezifikation 35 nochmal gezeigt. In die Zusammenfassung kommt sie jedoch nicht mit rein.

## Spezifikation Getränkeautomat

- Beschreibung Getränkeautomat
  - Zwei Knöpfe sweet und white, zur Auswahl von Zucker oder Milch oder beides
  - Eine bereits getroffene Auswahl kann durch nochmaliges drücken zurückgenommen werden
  - Die Reihenfolge der Knöpfe ist gleichgültig

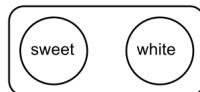


Abbildung 11: Sweet and white button

- Algebraische Spezifikation Getränkeautomat
  - Sorten  $s = \{\text{Add}, \text{Choice}\}$
  - Operationen:
    - sweet:  $- \rightarrow \text{Add}$
    - white:  $- \rightarrow \text{Add}$
    - noChoice:  $- \rightarrow \text{Choice}$
    - press:  $\text{Add} \times \text{Choice} \rightarrow \text{Choice}$
    - Ein Term beschreibt eine Folge von Tastendrücken  $\text{press}(\text{sweet}, \text{press}(\text{white}, \text{press}(\text{sweet}, \text{noChoice})))$
  - Axiome:
    - Q1:  $\text{press}(\text{a}, \text{press}(\text{a}, \text{c})) = \text{c}$
    - Q2:  $\text{press}(\text{sweet}, \text{press}(\text{white}, \text{c})) = \text{press}(\text{white}, \text{press}(\text{sweet}, \text{c}))$
- Terme in Normalform
  - noChoice
  - $\text{press}(\text{white}, \text{noChoice})$
  - $\text{press}(\text{sweet}, \text{noChoice})$
  - $\text{press}(\text{white}, \text{press}(\text{sweet}, \text{noChoice}))$
  - Alle anderen Terme können in diese vier Terme umgeformt werden

## 4 Logik

In diesem Kapitel befassen wir uns mit zwei klassischen Gebieten der Logik, und zwar der Aussagenlogik und der Prädikatenlogik erster Stufe. Beide Gebiete gehören zur mathematischen Logik und gehen von zwei Wahrheitswerten, dem Wert (wahr) und dem Wert (falsch), aus.

Lange Zeit waren die Grundlagen der Mathematik das Hauptanwendungsgebiet der Logik. Durch die Informatik mit ihrem Bedarf an formalen Werkzeugen sind neue und interessante Fragen aufgeworfen worden. Neben der Modellbildung finden sich Anwendungen der Logik zum Beispiel in der Künstlichen Intelligenz, im Gebiet der Datenbanken oder bei der Verifikation von Softwaresystemen.

Wir haben uns hier auf die für Anfänger wichtigsten Grundlagen beschränkt. So fehlen Kalküle für das logische Schließen und Sätze über die Grenzen der Prädikatenlogik.

1. die Syntax und die Semantik sowohl der Aussagenlogik wie auch der Prädikatenlogik zu vermitteln
2. die Grundlagen für den sicheren Umgang mit den logischen Operatoren, wie zum Beispiel der  $\wedge$  - und  $\vee$ - Verknüpfung, und den Quantoren  $\exists$  und  $\forall$  zu vermitteln.
3. Normalformen, wie zum Beispiel die konjunktive Normalform, und Transformationsverfahren von Formeln in die verschiedenen Normalformen vorzustellen.
4. ein Grundverständnis für die Modellierung mit Hilfe der Aussagenlogik und der Prädikatenlogik erster Stufe zu vermitteln.

## 4.1 Relationen und Funktionen

### Aussagenlogik in der Spezifikation

- Aussagenlogik
  - Sätze, die prinzipiell als wahr oder falsch angesehen werden können
  - Beispiel: „Es regnet“, „Die Strasse ist naß“
- Junktoren verknüpfen Aussagen
  - Es regnet nicht, oder die Straße ist nass
- Belegung mit Wahrheitswerten
  - Regen  $\rightarrow$  straßeNass
- Grundsymbole
  - Variablen
  - Funktionssymbole. 0-stelliges Funktionssymbol wird Konstante genannt
  - Junktoren  $\wedge$  (und),  $\vee$  (oder),  $\rightarrow$  (Implikation, wenn-dann),  $\leftrightarrow$  Äquivalenz (genau dann, wenn)
- Präzedenzregeln
  - $\neg$  bindet stärker als  $\wedge$
  - $\wedge$  bindet stärker als  $\vee$
  - $\vee$  bindet stärker als  $\rightarrow$  und  $\leftrightarrow$
- Weitere Definitionen
  - Elementaraussagen bezeichnet man auch als Atome
  - Ein Literal ist ein Atom A oder ein negiertes Atom  $\neg A$
- Aussagenlogische Formeln: Terme mit Variablen zur Signatur der boolschen Algebra
  - false:  $\rightarrow$  BOOL
  - true:  $\rightarrow$  BOOL
  - $\wedge$ : BOOL  $\times$  BOOL  $\rightarrow$  BOOL
  - $\vee$ : BOOL  $\times$  BOOL  $\rightarrow$  BOOL
  - $\neg$ : BOOL  $\rightarrow$  BOOL
- Erweiterung
  - $\rightarrow$ : BOOL  $\times$  BOOL  $\rightarrow$  BOOL
  - $\leftrightarrow$ : BOOL  $\times$  BOOL  $\rightarrow$  BOOL

### Bewertung eines Terms

- $\Im$  sei Bewertung von Formeln

- $\Im(\neg\alpha) = w$  genau dann, wenn  $\Im(\alpha)=f$
- $\Im(\alpha \wedge \beta) = w$  genau dann, wenn  $\Im(\alpha)=w$  und  $\Im(\beta)=w$
- $\Im(\alpha \vee \beta) = w$  genau dann, wenn  $\Im(\alpha)=w$  oder  $\Im(\beta)=w$
- $\Im(\alpha \rightarrow \beta) = w$  genau dann, wenn  $\Im(\alpha)=f$  oder  $\Im(\beta)=w$
- $\Im(\alpha \leftrightarrow \beta) = w$  genau dann, wenn  $\Im(\alpha)=\Im(\beta)=w$  oder  $\Im(\alpha)=\Im(\beta)=f$
- $\Im(\text{true}) = w$  und  $\Im(\text{false})=f$

Abbildung 12: Bewertung von Formeln

- Wahrheitstafel

$\alpha$	$\beta$	$\neg\alpha$	$\alpha \wedge \beta$	$\alpha \vee \beta$	$\alpha \rightarrow \beta$	$\alpha \leftrightarrow \beta$	$\alpha \vee \beta$
W	W	F	W	W	W	W	F
W	F	F	F	W	F	F	W
F	W	W	F	W	W	F	W
F	F	W	F	F	W	W	F

Abbildung 13: Wahrheitstafel

## Umformungsregeln

Negation	$\neg\neg\alpha \approx \alpha$	
Idempotenz	$\alpha \vee \alpha \approx \alpha$ $\alpha \wedge \alpha \approx \alpha$	
Kommutativität	$\alpha \vee \beta \approx \beta \vee \alpha$ $\alpha \wedge \beta \approx \beta \wedge \alpha$	
Assoziativität	$(\alpha \vee \beta) \vee \gamma \approx \alpha \vee (\beta \vee \gamma)$ $(\alpha \wedge \beta) \wedge \gamma \approx \alpha \wedge (\beta \wedge \gamma)$	
Distributivität	$(\alpha \wedge \beta) \vee \gamma \approx (\alpha \vee \gamma) \wedge (\beta \vee \gamma)$ $(\alpha \vee \beta) \wedge \gamma \approx (\alpha \wedge \gamma) \vee (\beta \wedge \gamma)$	
De Morgan	$\neg(\alpha \wedge \beta) \approx \neg\alpha \vee \neg\beta$ $\neg(\alpha \vee \beta) \approx \neg\alpha \wedge \neg\beta$	
Komplement	$\alpha \vee \neg\alpha \approx \text{true}$	$\alpha \wedge \neg\alpha \approx \text{false}$
Neutrale Elemente	$\alpha \wedge \text{true} \approx \alpha$ $\alpha \vee \text{true} \approx \text{true}$	$\alpha \wedge \text{false} \approx \text{false}$ $\alpha \vee \text{false} \approx \alpha$

Abbildung 14: Umformungsregeln

Implikation	$\alpha \rightarrow \beta$ $\approx \neg\alpha \vee \beta$ $\approx \beta \vee \neg\alpha$ $\approx \neg\beta \rightarrow \neg\alpha$
Äquivalenz	$\alpha \leftrightarrow \beta \approx (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
Exklusives Oder	$\alpha \underline{\vee} \beta \approx (\alpha \wedge \neg\beta) \vee (\neg\alpha \wedge \beta)$ $\alpha \underline{\vee} \beta \approx (\alpha \vee \beta) \wedge (\neg\alpha \vee \neg\beta)$

Abbildung 15: Weitere Umformungsregeln

## Wahrheitstafel

- Wahrheitstafel für komplexere Formeln
  - Definieren Sie die Wahrheitstafel für  $(A \vee \neg B) \wedge (A \vee B -> C)$

$A$	$B$	$C$	$(A \vee \neg B)$	$(A \vee B) \rightarrow C$	$(A \vee \neg B) \wedge (A \vee B \rightarrow C)$
f	f	f	w	w	w
f	f	w	w	w	w
f	w	f	f	f	f
f	w	w	f	w	f
w	f	f	w	f	f
w	f	w	w	w	w
w	w	f	w	f	f
w	w	w	w	w	w

Abbildung 16: Wahrheitstafel

## Normalformen

- Konjunktive Normalform KNF
  - Eine Formel ist in konjunktiver Normalform, wenn sie eine Konjunktion von Disjunktionen von Literalen ist
  - Beispiel:  $(A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee \neg B)$
- Disjunktive Normalform DNF
  - Eine Formel ist in disjunktiver Normalform, wenn sie eine Disjunktion von Konjunktionen von Literalen ist
  - Beispiel:  $(A \wedge B) \vee (A \wedge \neg B) \vee (\neg A \wedge \neg B)$
- Zu jeder aussagenlogischen Formel gibt es
  - eine äquivalente Formel in KNF
  - eine äquivalente Formel in DNF

## Erfüllbar, tautologisch, falsifizierbar

- Definitionen

- Eine Formel ! ist erfüllbar genau dann, wenn es eine Belegung gibt, bei der sie true ist
- Eine Formel ! ist tautologisch (eine Tautologie, allgemeingültig) genau dann, wenn sie für jede Belegung true ist
- Eine Formel ! ist widerspruchsvoll (unerfüllbar) genau dann, wenn sie für jede Belegung false ist
- Eine Formel ! ist falsifizierbar genau dann, wenn es eine Belegung gibt, bei der sie false ist

## Aussagenlogische Modellbildung

- Aussagenlogik zur Modellierung von statischem Wissen

- Dynamische Systeme werden normalerweise nicht mit Aussagenlogik modelliert
- Die Aussage „Es regnet“ wird dem Atom R und „Die Straße ist naß“ dem Atom S zugeordnet

Es regnet nicht	$\neg R$
Es regnet oder die Straße ist nass	$R \vee S$
Es regnet und die Straße ist nass	$R \wedge S$
Wenn es regnet, ist die Straße nass	$R \rightarrow S$
Genau dann, wenn es regnet, ist die Straße nass	$R \leftrightarrow S$

Abbildung 17: Es regnet Beispiel

Die Straße ist nass genau dann, wenn es regnet	$R \leftrightarrow S$
Die Straße ist nass dann und nur dann, wenn es regnet	$R \leftrightarrow S$
Entweder ist die Straße nass oder es regnet	$R \vee S$ $(R \vee S) \wedge (\neg R \vee \neg S)$ $(R \wedge \neg S) \vee (\neg R \wedge S)$

Abbildung 18: Es regnet Beispiel

## Formalisierung umgangssprachlicher Ausdrücke

- Vorsicht bei Implikationen; mit Belegungen prüfen, was gemeint ist

Wenn es regnet, benutze ich den Schirm	$\text{regnet} \rightarrow \text{schirm}$
Ich benutze den Schirm, wenn es regnet	$\text{regnet} \rightarrow \text{schirm}$
Ich benutze den Schirm nur wenn es regnet	$\text{schirm} \rightarrow \text{regnet}$

- „Oder“ kann fast immer in das  $\vee$  (xor) übersetzt werden

Hast Du einen Euro <b>oder</b> zwei Fünfziger?	$\text{euro} \vee \text{zweiFünfziger}$
Morgen fahr ich mit dem Auto <b>oder</b> dem Zug nach Berlin.	$\text{zug} \vee \text{auto}$
x ist kleiner y <b>oder</b> y ist kleiner x	$x < y \vee y < x$
Der Händler gibt Rabatt <b>oder</b> ein kostenloses Radio	$\text{rabatt} \vee \text{radio}$

Abbildung 19: Beispiel

- Aussagen sind häufig kontext-abhängig

Weil das Wetter schön ist, gehe ich nicht in die Vorlesung	$\text{Wetter\_schön} \wedge \neg \text{Vorlesung\_besuchen}$
Weil ich die Matheklausur nicht bestanden habe, nehme ich am zweiten Versuch teil	$\neg \text{Mathe-bestanden} \wedge \text{Mathe-Versuch2}$

- Klammern sind meist nur aus dem Kontext erkennbar

Sie wollten <b>nicht</b> verlieren <b>oder</b> unentschieden spielen	$\neg(\text{verlieren} \vee \text{unentschieden})$ oder $\neg \text{verlieren} \vee \text{unentschieden}$
--	---

→ Vorsicht beim Übertragen von Umgangssprache in Formeln, insbesondere bei Klammern und Implikationen

Abbildung 20: Weitere Beispiele

Weitere Beispiele sind auf der Folie FMSM– 3. Logik 18 zu finden ()

## Prädikatenlogik

- Erweiterung um parametrisierte Elementaraussagen
- n-stellige Prädikate sind Operationen  $P: T^n \rightarrow \text{BOOL}$  In Konkretisierung entsprechen ihnen n-stellige Relationen
  - „x ist eine Katze“ als Formel:  $\text{istKatze}(x)$
  - „a teilt b“ als Formel:  $\text{teilt}(a, b)$
- 0-stellige Prädikate als Konstante
  - $\text{istKatze}(\text{molly})$
  - $\text{istKatze}(\text{peterle})$
- Informale Definition
  - Individuen als Grundobjekte von Strukturen
  - Ihre Menge als Individuenbereich oder „Universum“
  - Konstante zur Bezeichnung ausgezeichneter Individuen
- Quantoren
  - „für alle x gilt  $\alpha$ “, in Symbolen:  $\forall x \alpha$  (Allquantor)
  - „es gibt ein x, so dass  $\alpha$  gilt“, in Symbolen:  $\exists x \alpha$  (Existenzquantor)
  - Wenn  $\alpha$  eine Formel darstellt, dann sind auch  $\exists x \alpha$  und  $\forall x \alpha$  Formeln
  - Variable sind nur als Operanden in Terme erlaubt, nicht für Funktionen oder Prädikate (Prädikatenlogik erster Stufe)
- Wirkungsbereich von Quantoren
  - $\forall x (P(x) \wedge Q(x)) \vee \exists y (P(y) \wedge \exists x R(y, x))$

Abbildung 21: Wirkungsbereich von Quantoren

- Freie und gebundene Variablen

- Ein Vorkommen von x in einer Formel  $\alpha$  heißt frei, wenn es nicht im Wirkungsbereich für x eines Quantors liegt
- Ein Quantor  $\forall x \alpha$  bzw.  $\exists x \alpha$  binden alle x, die frei sind in  $\alpha$ . Die Variable x heißt dann gebunden
- Beispiel: Formel  $\alpha = (R(y) \wedge \exists y (P(y, x) \vee Q(y, z)))$   $\alpha$  hat drei freie Variable x, y, z. y kommt frei und gebunden vor
- Eine Formel ist eine Aussage, genau dann wenn sie keine freie Variablen enthält
- Gebundene Namen können umbenannt werden

- Beispiele
  - Hans liebt Gaby wird zu:  $\text{liebt}(\text{Hans}, \text{Gaby})$
  - Alle lieben Gaby wird zu:  $\forall x (\text{liebt}(x, \text{Gaby}))$

Weitere Beispiele sind auf der Folie FMSM– 3. Logik 24 zu finden ()

- Negation
  - $\neg \forall x \alpha \approx \exists x \neg \alpha$  Oder:  $\neg \exists x \alpha \approx \forall x \neg \alpha$
  - Beispiel: Alle haben den Schuss gehört Negiert: Es gibt einen, der den Schuss nicht gehört  
hat Falsch negiert: Keiner hat den Schuss gehört
- Quantoren vertauschen
  - $\forall x \forall y \alpha \approx \forall y \forall x \alpha$   $\exists x \exists y \alpha \approx \exists y \exists x \alpha$
  - Vorsicht, nicht äquivalent:  $\forall s \exists v \text{ bestanden}(s, v) \not\approx \exists v \forall s \text{ bestanden}(s, v)$
  - Alle haben eine Prüfung bestanden ist nicht das gleiche wie Es gibt eine Prüfung die alle bestanden haben
- Quantoren zusammenfassen
  - $\forall x \alpha \wedge \forall x \beta \approx \forall x \alpha \wedge \beta$  Oder:  $\exists x \alpha \vee \exists x \beta \approx \exists x \alpha \vee \beta$

Weitere Beispiele sind auf der Folie FMSM– 3. Logik 27 zu finden ()

## 5 Anhang

Alle Abbildungen (mit Ausnahme des offiziellen HGS-Singen Logos) bis hierher sind von uns gemacht worden und verletzen somit keine Urheberrechte.

Nachfolgend die Regeln zu den einzelnen Spielen. Quellen:

# 1 Grundlagen

- 1 Grundlagen
  - 1 Motivation
  - 2 Einführung Softwaremodellierung
  - 3 Beispiel
- 2 Algebraische Spezifikation
- 3 Logik
- 4 Zeichenfolgen

# Literaturempfehlung

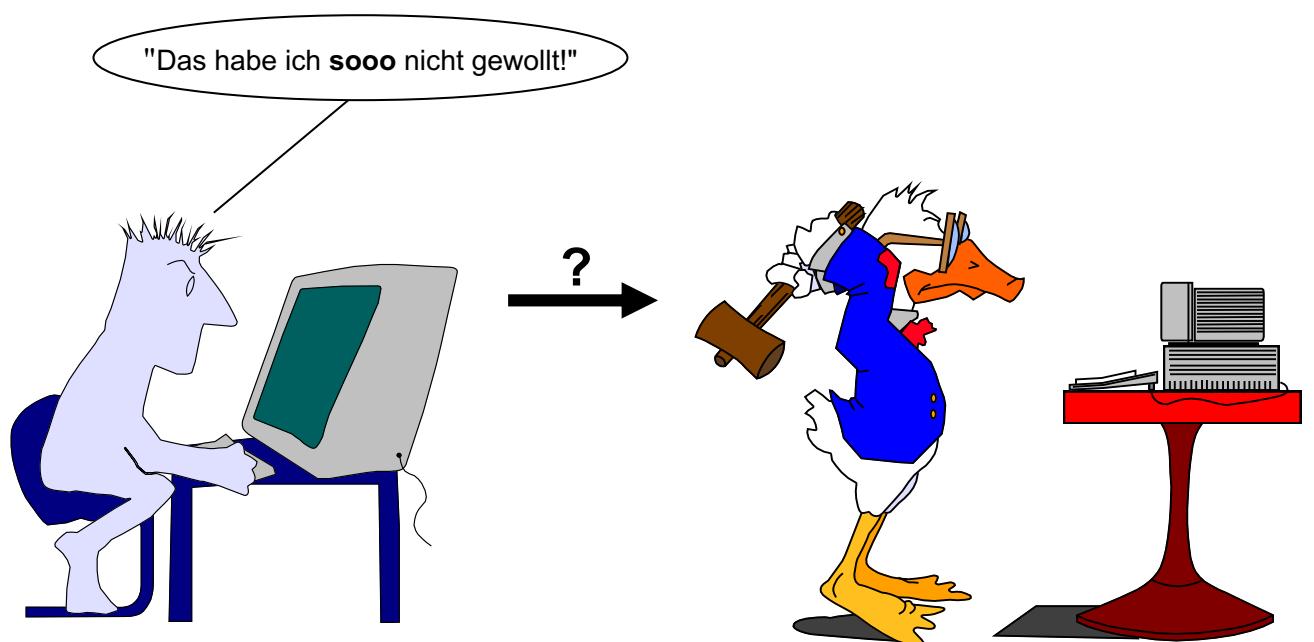
Kastens, U., Kleine Büning, Hans: Modellierung – Grundlagen und formale Methoden, Carl Hanser Verlag München, 4. Auflage, 2018



# Software

- Software bestimmt fast alle unsere Lebensbereiche
- Software „kann“ immer mehr, z.B. intelligente Systeme
- Hohe Abhängigkeit von Software
- Häufige Qualitätsmängel bei Software
  - Systeme sind selten fehlerfrei
  - Fehler in Software können Katastrophen als Auswirkung haben, z.B. Flugzeugabsturz
- Entwicklung von Software als Schlüsseldisziplin

# Was wissen Sie über die Softwarekrise?



# Software-Krise damals und heute

„Als es noch keine Rechner gab, war auch das Programmieren noch kein Problem, als es dann ein paar leistungsschwache Rechner gab, war das Programmieren ein kleines Problem und nun, wo wir gigantische Rechner haben, ist auch das Programmieren zu einem gigantischen Problem geworden. In diesem Sinne hat die elektronische Industrie kein einziges Problem gelöst, sondern nur neue geschaffen. Sie hat das Problem geschaffen, ihre Produkte zu nutzen...“

E. W. Dijkstra: The Humble Programmer, 1972

"Um in E-Commerce-Umgebungen zu testen, ist ein erheblicher Zusatzaufwand erforderlich ... im letzten Jahr mussten die Tester mehrere neue Versionen von Browsern, Java-Klassenbibliotheken und –Development Kits berücksichtigen. Daraus ergaben sich rund 240 verschiedene Kombinationen für Testsituationen"

Aus: Fraunhofer Gesellschaft, Institut für Experimentelles Softwareengineering, Informationweek 1/1999

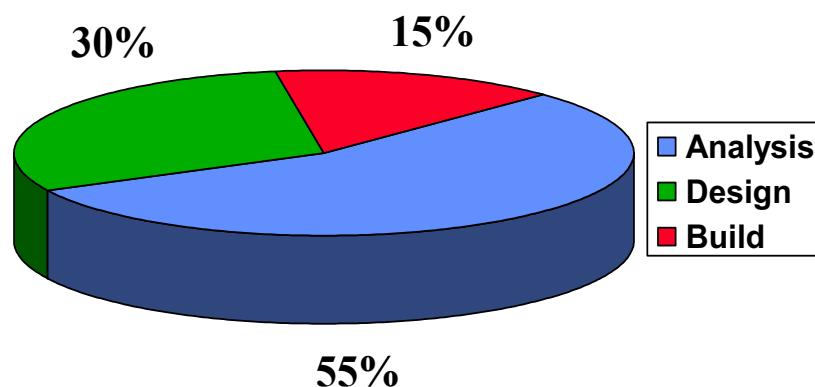
# Untersuchung der Fehlerquellen

**amdaHL**

*Making Sense of AD Trends*

## The Need for Change

### Error incidence in implemented applications





How the customer explained it



How the Project Leader understood it



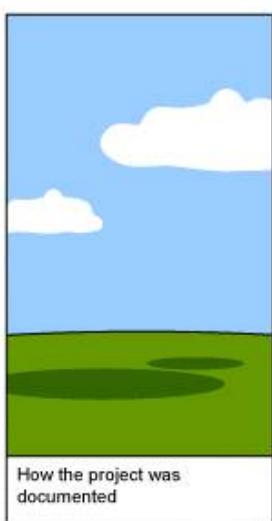
How the Analyst designed it



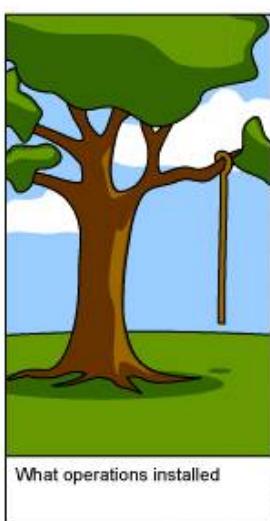
How the Programmer wrote it



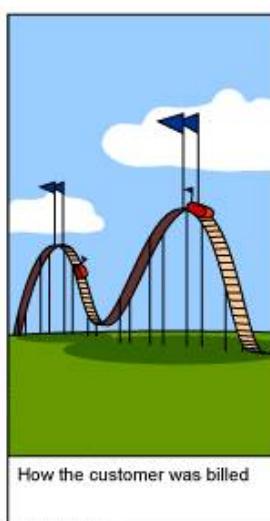
How the Business Consultant described it



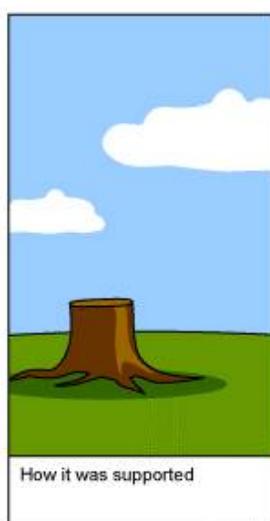
How the project was documented



What operations installed



How the customer was billed



How it was supported



What the customer really needed

# Probleme bei Software-Entwicklung

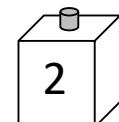
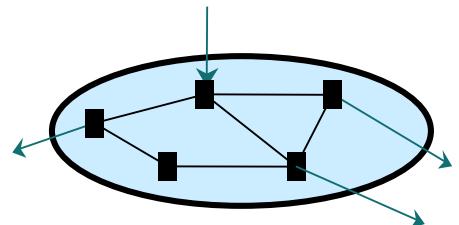
- Verständigungsschwierigkeiten zw. Entwicklern und Anwendern
- Benutzeranforderungen werden häufig
  - nicht systematisch erfasst,
  - sind selbst Benutzern unbekannt,
  - ändern sich regelmässig
- Der typische Anwender
  - I can't tell you what I want, but I'll know it when I see it
- Folge
  - Systeme erfüllen Anforderungen nicht
  - werden zu spät fertig gestellt,
  - sind zu teuer

# Probleme bei Software-Entwicklung

- Natürlichsprachliche Beschreibung oft mit geringer Qualität
  - Widerspruchsvoll
  - Unvollständig
- Beispiel einer einfachen Beschreibung
  - „Copy kopiert markierten Text in Zwischenablage, Paste kopiert Zwischenablage in aktuelle Textposition.“
  - Was ist an dieser Beschreibung nicht definiert?
    - Was passiert bei Copy, wenn kein Text markiert?
    - Was passiert bei Paste, wenn kein Copy zuvor erfolgte?
    - Was passiert bei Paste, wenn Text markiert ist?
    - Was passiert bei Paste, wenn beides zutrifft?
    - Was passiert bei Copy bzw. Paste, wenn zweimal hintereinander ausgeführt?
    - Was passiert bei Paste, wenn keine Textposition ausgewählt?

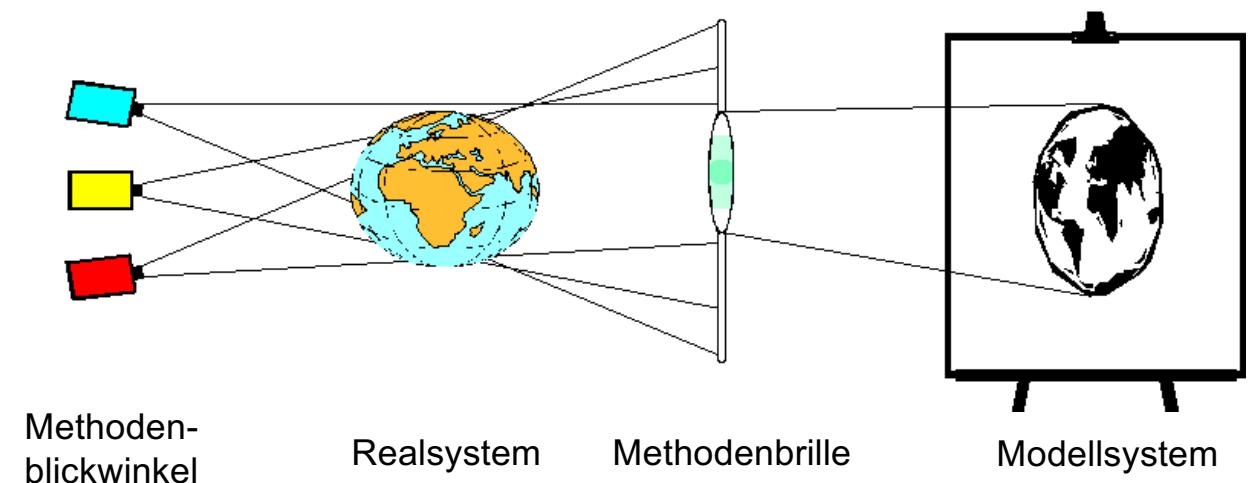
# Definitionen System

- Definition
  - Der Begriff System umschreibt eine Realität mit allen für den Untersuchungszweck relevanten Wechselwirkungen zwischen ihren Bestandteilen
- Blackbox / Whitebox – Sicht
- Klassifikation von Systemen
  - statisch – dynamisch
  - ideell – real
  - deterministisch – stochastisch

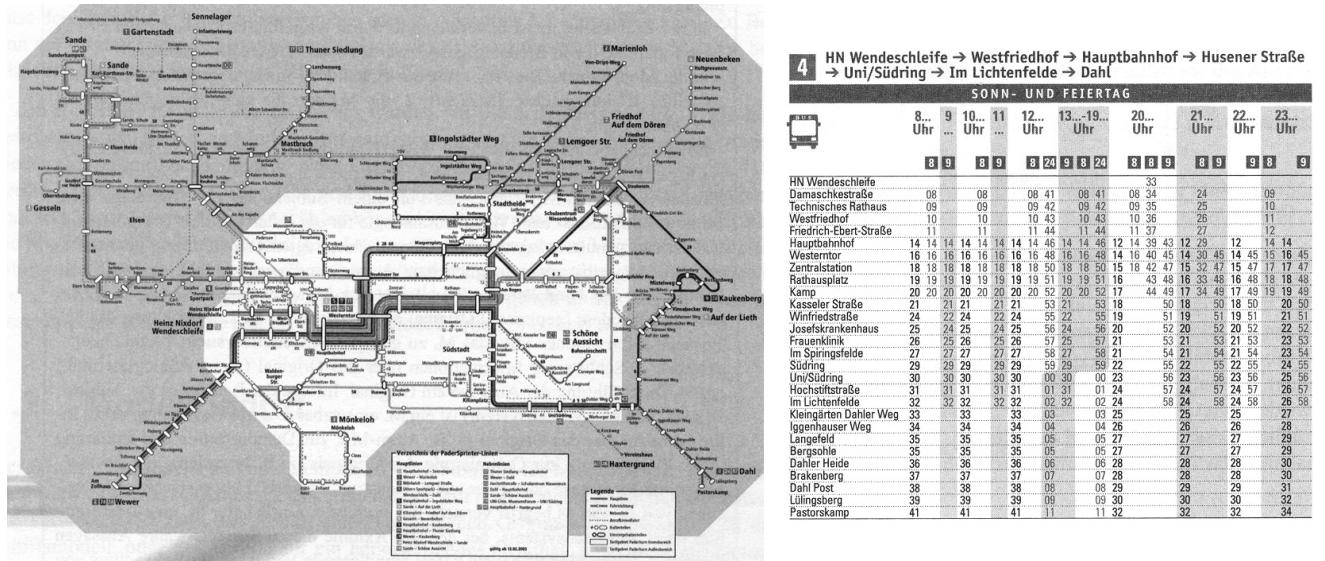


# Projektion in der Modellierung

- Definition "Modell" nach Helmut Balzert
  - Modell als idealisierte, vereinfachte, eines Gegenstands, Systems oder sonstigen Weltausschnitts
  - Ziel: bestimmte Eigenschaften des Vorbilds zu studieren



# Abstraktion in der Modellierung - Beispiel

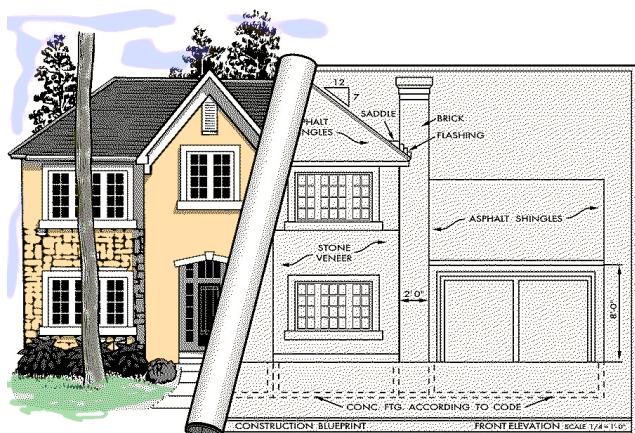


Quelle: Kastens, Kleine Büning: Modellierung, Hanser

# Motivation Bau eines Wohnhauses

Eine **Entwicklung nach ingenieurmäßigen Prinzipien** bedeutet, ein Vorhaben systematisch zu konstruieren:

- Grundrisszeichnungen,
- Berechnungen,
- Modellbau...



**Modellierungsansätze** unterstützen die

- **Visualisierung**
- **Spezifikation**
- **Konstruktion**
- **Dokumentation** der Konstruktionsbausteine

# Aspekte eines Modells

- Verwendungszweck eines Modells bestimmt die Art des Modells
- Beispiel: Mögliche Verwendungszwecke eines Hausbaus
  - Gebäudemodell: optischer Eindruck
  - Grundriss: Einteilung des Grundstücks und Räume
  - Kostenplan: Finanzierung
  - Bauplan: Bauabwicklung
- Verwendete Methoden und Kalküle in der Modellierung
  - Struktur: Wertebereiche, Entity-Relationship, Klassifikation, Typen
  - Eigenschaften: Logik, Relationen
  - Beziehungen: Graphen, Relationen, Logik, Entity-Relationship
  - Verhalten: endliche Automaten, Petrinetze, Algebren, Graphen

## Einführendes Beispiel

Ein Mann steht mit einem Wolf, einer Ziege und einem Kohlkopf am linken Ufer eines Flusses, den er überqueren will. Er hat ein Boot, das groß genug ist, ihn und ein weiteres Objekt zu transportieren, so dass er immer nur eins der drei mit sich hinübernehmen kann. Falls der Mann allerdings den Wolf und die Ziege oder die Ziege und den Kohlkopf unbewacht an einem Ufer zurücklässt, so wird einer gefressen werden. Ist es möglich, den Fluss zu überqueren, ohne dass die Ziege oder der Kohlkopf gefressen werden?

Quelle: Hopcroft, Ullman: Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie, S. 14, 15

# Einführendes Beispiel – erste Analyse

Ein **Mann** steht mit einem **Wolf**, einer **Ziege** und einem **Kohlkopf** am **linken Ufer** eines **Flusses**, den er überqueren will. Er hat ein **Boot**, das groß genug ist, **ihn und ein weiteres Objekt** zu transportieren, so dass er immer nur eins der drei mit sich hinübernehmen kann. Falls der Mann allerdings den **Wolf** und die **Ziege** oder die **Ziege** und den **Kohlkopf** unbewacht an **einem Ufer** zurücklässt, so wird einer **gefressen** werden. Ist es möglich, den Fluss zu überqueren, ohne dass die Ziege oder der Kohlkopf gefressen werden?

- Objekte
  - **Mann, Wolf, Ziege, Kohlkopf, Ufer (links u. rechts), Boot**
- Eigenschaften, Beziehungen
  - unbewacht an einem Ufer, Wolf frisst Ziege, Ziege frisst Kohl, Boot trägt Mann + 1 Objekt
- Tätigkeiten
  - überqueren

Quelle: Kastens, Kleine Büning: Modellierung, Hanser

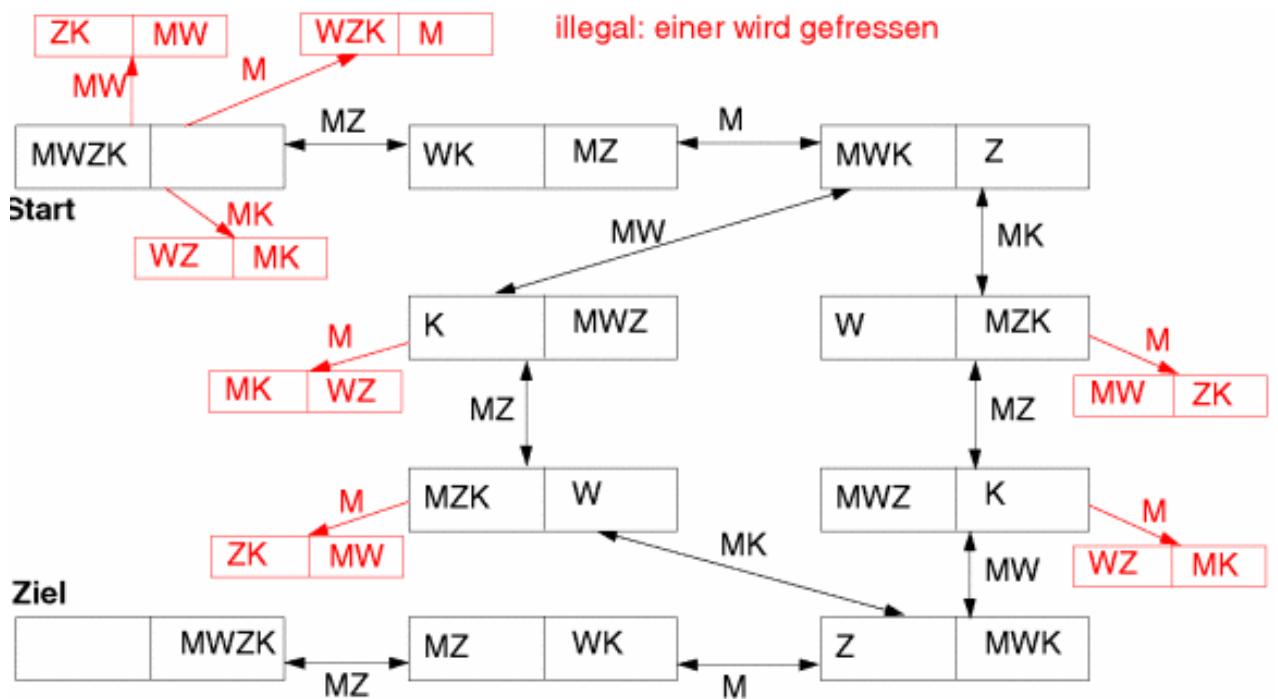
# Einführendes Beispiel – Diskussion

- Modellierung von Abläufen: Kalkül endlicher Automat
- Abstraktion: nur Zustände und Übergänge interessant
- Relevante Objekte: M, W, Z, K
- Jeder Zustand charakterisiert durch ein Paar von Objektmengen
  - Linkes und rechtes Ufer
  - Jedes Objekt kommt nur einmal vor
- Nicht modelliert werden
  - Eigenschaften des Bootes, z.B. Länge
  - Breite des Flusses
- Modellierung von
  - Zustandsübergängen
  - Test ob Zustand illegal

M	W	Z	K

© Prof. Dr. Uwe Kastens

# Einführendes Beispiel – endlicher Automat mit Übergängen



© Prof. Dr. Uwe Kastens

# Gliederung der Vorlesung

- 1 Grundlagen
- 2 Algebraische Spezifikation
  - 1 Modellierung von Datenstrukturen
  - 2 Relationen und Funktionen
  - 3 Algebraische Spezifikation von Datenstrukturen
- 3 Logik
- 4 Zeichenfolgen

# Einführung

- Aufgabe
  - Ziehe drei Karten aus einem Kartenspiel. Bestimme die höchste der drei Karten
- Fragen
  - Was sind die Wertebereiche für Karten?
  - Welche Karte ist „höher“ als eine andere?
- Lösung
  - Definition des Wertebereichs eines Kartenspiels
  - Definition einer Relation „höher“ für das Kartenspiel

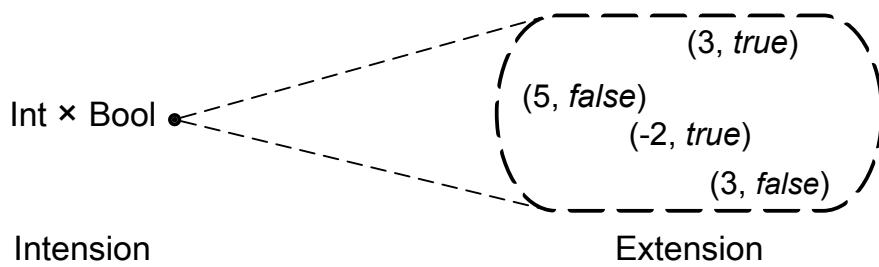


# Definition Menge und Wertebereich

- Wertebereich
  - Ein Wertebereich ist eine Menge von Werten, die im Sinne eines Modells als gleichartig angesehen werden
  - Wo ein Wert eines Wertebereichs  $W$  gefordert wird, kann prinzipiell jedes Element aus  $W$  diese Rolle übernehmen
- Menge
  - Mengen zur Modellierung von Wertebereichen
  - Eine Menge  $M$  ist eine Zusammenfassung von verschiedenen Objekten, den Elementen der Menge.
  - $a$  ist ein Element der Menge  $M$
  - Notation:  $a \in M$
  - Die Elemente in einer Menge sind nicht geordnet

# Definition von Typen

- Intensionale Sicht
  - Definition durch Angabe einer Bedingung, die alle Elemente erfüllen
  - Beispiel
    - $\{a \in \mathbb{N}, a \text{ ist Quadratzahl und } a < 30\}$
- Extensionale Sicht
  - Definition als Auflistung von Elementen
  - Beispiel
    - $\{1, 4, 9, 16, 25\}$



Quelle: P. Pepper: Funktionale Programmierung

# Mengenoperationen

- Mengenoperationen

Bezeichnung	Notation	Bedeutung
Ist Teilmenge	$M \subseteq N$	Aus $a \in M$ folgt $a \in N$
Ist echte Teilmenge	$M \subset N$	$M \subseteq N$ und $M \neq N$
Vereinigung	$M \cup N$	$\{x \mid x \in M \text{ oder } x \in N\}$
Durchschnitt	$M \cap N$	$\{x \mid x \in M \text{ und } x \in N\}$
Differenz	$M \setminus N$	$\{x \mid x \in M \text{ und } x \notin N\}$

- Definitionen
  - Zwei Mengen  $M$  und  $N$  sind **disjunkt**, wenn gilt:  $M \cap N = \emptyset$
  - Die Anzahl der Elemente einer Menge  $M$  heißt ihre **Kardinalität**, notiert als  $|M|$

# Potenzmenge

- Definition Potenzmenge
  - Die Potenzmenge einer Grundmenge  $U$  ist die Menge aller Teilmenge von  $U$ , geschrieben  $\text{Pow}(U)$  oder  $\mathbb{P}(U)$
  - Als Formel:  $\mathbb{P}(U) = \{M \mid M \subseteq U\}$
- Beispiel
  - BeigabenArten = {Milch, Zucker}  
Beigaben =  $\mathbb{P}(\text{BeigabenArten})$
  - Sprachen = {Deutsch, Französisch, Spanisch, Englisch}  
Von welchem Typ ist die Muttersprache einer Person?  
Von welchem Typ ist gesprochene Sprache einer Person?

# Kartesisches Produkt

- Definition kartesisches Produkt oder geordnetes Paar oder Kreuzprodukt
  - Ein geordnetes Paar  $(x, y)$  besteht aus zwei Werten  $x$  und  $y$ , wobei  $x$  die erste und  $y$  die zweite Komponente ist.
  - Das kartesische Produkt  $M \times N$  zweier Mengen  $M$  und  $N$  ist die Menge aller geordneten Paare mit erster Komponente aus  $M$  und zweiter Komponente aus  $N$
  - In Formeln:  $M \times N = \{(x, y) \mid x \in M \text{ und } y \in N\}$
- Wie lassen sich die Spielkarten definieren???

# Kartesisches Produkt am Beispiel Spielkarten

- Definition Spielkarten
  - KartenArten = {Kreuz, Pik, Herz, Karo}
  - KartenSymbole = {7, 8, 9, 10, Bube, Dame, König, Ass}
  - Karte = KartenArten × KartenSymbole
- Was ist der Vorteil an dieser Darstellung gegenüber einer einfachen Aufzählung aller Karten?
- Wie viele Karten gibt es?
- Wie lassen sich rote und schwarze Karten unterscheiden?

# Verallgemeinerung kartesisches Produkt

- Definition kartesisches Produkt mit  $n > 1$  Mengen, als Menge von geordneten n-Tupeln
  - $M_1 \times M_2 \times \dots \times M_n = \{(a_1, a_2, \dots, a_n) \mid a_i \in M_i \text{ und } i \in I\}$   
mit Indexmenge  $I = \{1, \dots, n\}$  und nichtleeren  $M_i$ .
- Wenn alle Komponenten aus dem selben Wertebereich kommen:
  - $M^n = M \times M \times \dots \times M$
- Modellieren Sie ein Lottoergebnis. Dieses besteht aus sechs verschiedenen Zahlen zwischen 1 und 49
  - $\text{Lottoergebnis} = \{x \mid x \in \mathbb{P}(\{1, 2, \dots, 49\}), |x| = 6\}$

# Beispiele kartesisches Produkt

- Beispiele und Aufgaben
  - KalenderDaten = Tag × Monat × Jahr  
Wie sind Tag, Monat, Jahr definiert???
  - Name = Vorname × Zuname  
Strasse = Strassenname × Hausnummer  
Ort = Postleitzahl × Ortsname  
Adresse = Name × Strasse × Ort
  - Wie sieht ein Element von Adresse aus?
  - Wurf = {1, 2, 3, 4, 5, 6}  
Wie sieht das Ergebnis eines dreimaligen Würfels aus?
  - Wie lassen sich die Maße eines Schranks definieren?

# Vereinigung

- Verwendung
  - Bilder = {Bube, Dame, König, Ass}  
Zahlwerte = {7, 8, 9, 10}  
KartenSymbole = Bilder  $\cup$  ZahlWerte
  - Kunde = {Siemens, VW, Bosch}  
Lieferant = {Siemens, Metabo}  
Geschäftspartner = Kunde  $\cup$  Lieferant

# Endliche Folgen

- Folge von Elementen können eine unterschiedliche Länge haben
- Definition endliche Folge
  - Ein  $n$ -Tupel aus  $A^n$  mit  $n > 1$  Komponenten aus der Menge  $A$  heißt Folge der Länge  $n$  über  $A$
  - $(a)$ , mit  $a \in A$  ist eine Folge der Länge 1 über  $A$
  - $()$  oder  $\varepsilon$  steht für die leere Folge
  - Definition Wertebereich der endlichen nicht-leeren Folgen über  $A$  als  $A^+ = \{(a) \mid a \in A\} \cup \{x \mid x \in A^i \text{ und } i > 1\}$
  - Definition Wertebereich der endlichen Folgen über  $A$ :  $A^* = \{\varepsilon\} \cup A^+$

# Endliche Folge

- Beispiel
  - WürfelWerte = {1, 2, 3, 4, 5, 6}  
WürfelFolge = WürfelWerte\*
  - Welche Werte kann der Wertebereich WürfelFolge haben?

## 2 Algebraische Spezifikation

- 2.1 Modellierung von Datenstrukturen
- 2.2 Relationen und Funktionen
- 2.3 Algebraische Spezifikation von Datenstrukturen

# Relationen – Einführung

- Relationen setzen Elemente aus unterschiedlichen Wertebereichen zueinander in Beziehung
- Beispiel 1
  - Personen = {Peter, Gaby, Jens, Inge}
  - Vorlesung= {Mathe1, Mathe2, SOMO, Prog1}
  - Wie lässt sich die Beziehung modellieren, wer welche Vorlesung besucht?
  - Wie kann diese Beziehung mathematisch in der Mengennotation dargestellt werden?
- Beispiel 2
  - Peter ist älter als Gaby, Gaby ist älter als Jens, Jens ist älter als Inge.
  - Wie kann dieser Sachverhalt dargestellt werden?

# Relationen – Definition

- Definition
  - Eine  $n$ -stellige Relation  $R$  ist eine Menge von  $n$ -Tupel, wobei jedes davon aus dem Wertebereich  $M_1 \times M_2 \times \dots \times M_n$  mit  $n > 1$  stammt
  - d.h.  $R \subseteq M_1 \times M_2 \times \dots \times M_n$
  - $R$  stammt aus dem Wertebereich  $\mathbb{P}(M_1 \times M_2 \times \dots \times M_n)$
  - Eine einstellige Relation über der Menge  $M$  ist eine Teilmenge von  $M$
- Beispiel
  - $A = \{a, b\}$ ,  $B = \{1, 2\}$
  - Wie ist  $A \times B$  definiert, wie ist  $\mathbb{P}$  von  $A \times B$  definiert, wie viele Elemente hat diese Potenzmenge?

# Relationen – Beispiel

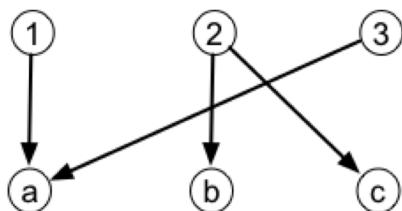
- Beispiel
  - Personen = {Peter, Gaby, Jens}
  - Sprachen = {Deutsch, Englisch, Spanisch}
  - spricht = {(Peter, Englisch), (Gaby, Spanisch), (Gaby, Deutsch), (Jens, Deutsch)}
  - Wie sieht die Potenzmenge von Personen x Sprachen aus?
  - Wie ist die Kardinalität der Potenzmenge?
  - Was sagt diese Zahl überhaupt aus?
- Andere Notation für zweistellige Relationen
  - $x R y$  statt  $(x, y) \in R$
  - Hier: Peter spricht Englisch, Gaby spricht Spanisch, etc.
  - Beispiel mit Operationszeichen:  $a > 1$ ,  $b < c$

# Relationen – Darstellungsmöglichkeiten

- Beispielrelation
  - $A = \{1, 2, 3\}$   $B = \{a, b, c\}$   $R \subseteq A \times B$
  - $R = \{(1, a), (2, b), (2, c), (3, a)\}$
- Darstellung als Matrix

	a	b	c
1	x		
2		x	x
3	x		

- Darstellung als Graph



# Zweistellige Relationen - Eigenschaften

- Eigenschaften

reflexiv	Wenn für alle $x \in M$ gilt: $x R x$
irreflexiv	Wenn für alle $x \in M$ gilt: $x R x$ gilt nicht
symmetrisch	Wenn für alle $x, y \in M$ gilt: aus $x R y$ folgt $y R x$
antisymmetrisch	Wenn für alle $x, y \in M$ gilt: aus $x R y$ und $y R x$ folgt $x = y$
asymmetrisch	Wenn für alle $x, y \in M$ gilt: aus $x R y$ folgt $y R x$ gilt nicht
transitiv	Wenn für alle $x, y, z \in M$ gilt: aus $x R y$ und $y R z$ folgt $x R z$

# Äquivalenz- und Ordnungsrelationen

- Definition Äquivalenzrelation
  - Eine zweistellige Relation  $R \subseteq M \times M$  ist eine **Äquivalenzrelation**, wenn sie reflexiv, symmetrisch und transitiv ist
- Definition Ordnungsrelation
  - Eine zweistellige Relation  $R \subseteq M \times M$  ist eine **partielle Ordnung** oder **Halbordnung**, wenn R reflexiv, antisymmetrisch und transitiv ist
  - eine **strenge Ordnung** oder **strenge Halbordnung**, wenn R irreflexiv und transitiv ist

# Funktionen

- Definition Funktion
  - Eine **Funktion**  $f$  ist eine 2-stellige Relation  $f \in P(D \times B)$  für die gilt:  
aus  $(x, y) \in f$  und  $(x, z) \in f$  folgt  $y = z$
  - Einem Wert aus  $D$  ist also höchstens ein Wert aus  $B$  zugeordnet
  - Die Menge  $D$  heißt **Definitionsbereich**, die Menge  $B$  **Bildbereich** oder **Wertebereich** der Funktion  $f$
  - Schreibweisen:  $(x, y) \in f$  oder  $y = f(x)$
  - Signatur einer Funktion:  $f: D \rightarrow B$
- Beispiel 1
  - Funktion, die Nationen auf Ihre Einwohnerzahlen in Millionen abbildet:  
 $\text{EinwohnerMio} = \{(Deutschland, 82), (Frankreich, 58), (\ddot{O}sterreich, 8), (\text{Spanien}, 39)\}$

# Funktionen

- Beispiel 2
  - Wie ist Definition- und Wertebereich für Multiplikation definiert?
- Beispiel 3
  - Personen = {Peter, Gaby, Jens}
  - Sprachen = {Deutsch, Englisch}
  - Wie müssen Definitions- und Wertebereich für die Funktion „spricht“ gewählt werden?

# Multimengen

- Modellierung von Münzen in einem Geldbeutel
- Problem: In einem Geldbeutel können Münzen mehrfach vorkommen
- Lösung 1: Modellierung der Anzahl wie oft Münze vorkommt
  - EuroMünzen = {1, 2, 5, 10, 20, 50, 100, 200}
  - Funktion EuroMünzen →  $\mathbb{N}_0$
  - MeinGeldBeutel = {(1,3), (2,0), (5,0), (10,2), (20,0), (50,1), (100, 4), (200,2)}
- Modellierung mit **Multimengen** (engl. bags)
  - {1,1,1,10,10,50,100,100,100,200,200}
  - Verwendung in vielen Datenstrukturen von Programmiersprachen (bag)

# Indexmenge

- Problem: manchmal möchte man Elementen einer Menge eine Position zuordnen
- Beispiel
  - $F := \{w,e,l,l,e\}$
  - Indexmenge Fpositionen = {1,...,5}
- Definition einer Funktion Fpositionen → F
  - Auftreten := {(1,w),(2,e),(3,l),(4,l),(5,e)}
  - Verwendung als Array in Programmiersprachen
  - In vielen Programmiersprachen wird in der Indexmenge mit 0 begonnen zu zählen

# Funktionen – Eigenschaften

- Sei Funktion  $f \in D \rightarrow B$

total	Wenn es für jedes $x \in D$ ein Paar $(x, y) \in f$ gibt
surjektiv	Wenn es zu jedem $y \in B$ ein Paar $(x, y) \in f$ gibt
injektiv	Wenn für zu jedem $y \in B$ höchstens ein Paar $(x, y) \in f$ gibt
bijektiv	Wenn $f$ zugleich surjektiv und injektiv ist

- Beispiele

- Welche Eigenschaften haben die Funktionen „not“, „EinwohnerMio“
- Welche Funktion ist nicht surjektiv?
- Welche Funktion ist nicht injektiv?

# Modellierung mit Wertebereichen

- Hinweise
  - Erst Grundmengen festlegen, dann Strukturen darüber bilden
  - Typische Elemente eines Wertebereichs angeben
  - Wertebereiche ausdruckskräftige Namen geben
  - Zusammengesetzte Wertebereiche schrittweise aufbauen
  - Nur gleichartige Elemente in einem Wertebereich
  - Mengen, Tupel und Folgen beliebiger Länge nicht verwechseln
  - Alle Klammern haben Bedeutung – zusätzliche verändern das Modell
  - Häufige Funktionen für Mengen
    - Zählen →  $\mathbb{N}_0$
    - Messen →  $\mathbb{R}$
    - Entscheiden → {true, false}

# Modellierung mit Wertebereichen – Beispiele

- Beispiel Delegierte
  - Nationen = {Deutschland, Frankreich, Österreich, Schweiz}
  - Sprache = {Deutsch, Französisch, Spanisch}
  - Jedes Land hat drei Delegierte:  
 $\text{DelegiertenIndex} = \{1, 2, 3\}$   
 $\text{Delegierte} = \text{Nationen} \times \text{DelegiertenIndex}$
  - Es gibt drei Arbeitskreise:  
 $\text{Arbeitskreise} = \{A1, A2, A3\}$
  - Definition welcher Delegierte welche Arbeitskreise besucht:  
 $\text{Besucht} \subseteq \text{Arbeitskreis} \times \text{DelegiertenIndex}$
  - Oder Definition als Funktion:  
 $\text{AkBesetzungen}: \text{Arbeitskreise} \rightarrow \mathbb{P}(\text{Delegierte})$   
 $\text{AkTeilnahme}: \text{Delegierte} \rightarrow \mathbb{P}(\text{Arbeitskreis})$

# Modellierung mit Wertebereichen – Beispiele

- Beispiel Getränkeautomat
  - BeigabeArten = {Milch, Zucker}
  - Größe = {groß, klein}
  - GetränkeArten = {Kaffee, Tee, Kakao}
  - Beigaben =  $\wp(\text{BeigabenArten})$
  - Getränke = GetränkeArten  $\times$  Beigaben  $\times$  Größe
  - Definition der Funktion Preis?

## 2 Algebraische Spezifikation

- 2.1 Modellierung von Datenstrukturen
- 2.2 Relationen und Funktionen
- 2.3 Algebraische Spezifikation von Datenstrukturen

# Terme, Sorten und Signaturen

- Term
  - Ausdruck in der Mathematik, der Variablen, Zahlen, Verknüpfungen, Klammern beinhalten kann
- Operatorbeschreibungen
  - $+$ : Zahl  $\times$  Zahl  $\rightarrow$  Zahl
  - $<$ : Zahl  $\times$  Zahl  $\rightarrow$  Bool
  - $\wedge$ : Bool  $\times$  Bool  $\rightarrow$  Bool
  - true:  $\rightarrow$  Bool
  - 1:  $\rightarrow$  Zahl
- Sorte
  - Disjunkte Teilmenge der Terme, die durch Operatoren gebildet wird
- Definition Stelligkeit
  - Eine Operator ist  $n$ -stellig mit  $n \geq 0$ , wenn er  $n$  Operanden hat.
  - 0-stellige Operatoren sind Konstanten

# Algebren

- Definition Signatur
  - Eine Signatur  $\Sigma = (S, F)$  beschreibt eine Menge von Sorten und eine Menge von Strukturbeschreibungen F
  - $op: s_1 \times \dots \times s_n \rightarrow s_o$  mit  $s_i \in S$
  - Beschreibung einer Funktion durch die Signatur  $D \rightarrow B$
  - Oder kurz:  $f: D \rightarrow B$
- Abstrakte Algebra
  - Eine abstrakte Algebra  $A = (\tau, \Sigma, Q)$  ist definiert durch die Signatur  $\Sigma$ , die Menge der korrekten Terme  $\tau$  zu  $\Sigma$  und eine Menge von Axiomen (Gesetze) Q.
  - Ein Axiom hat die Form  $t_1 \rightarrow t_2$ , wobei  $t_1$  und  $t_2$  korrekte Terme gleicher Sorte sind und Variablen enthalten können.
- Konkrete Algebra
  - Zuordnung konkreter Wertebereiche

# Abstrakte boolsche Algebra

- Signatur
  - Signatur  $\Sigma = (S, F)$ ,  $S = \{\text{BOOL}\}$  Operationen  $F$ :  
true:  $\rightarrow \text{BOOL}$   
false:  $\rightarrow \text{BOOL}$   
 $\neg$ :  $\text{BOOL} \rightarrow \text{BOOL}$   
 $\wedge$ :  $\text{BOOL} \times \text{BOOL} \rightarrow \text{BOOL}$   
 $\vee$ :  $\text{BOOL} \times \text{BOOL} \rightarrow \text{BOOL}$
  - Konstante true und false als „Grenzfall“ von Funktionen
- Axiome Q: für alle  $x, y$  der Sorte BOOL gilt:
  - Q1:  $\neg \text{true} = \text{false}$
  - Q2:  $\neg \text{false} = \text{true}$
  - Q3:  $\text{true} \wedge x = x$
  - Q4:  $\text{false} \wedge x = \text{false}$
  - Q5:  $x \wedge y = \neg(\neg x \vee \neg y)$

# Algebraische Spezifikation von Datenstrukturen

- Datenstruktur Stack (Keller)
  - Datenstruktur zum Einfügen und Entfernen von Elementen
  - Last-In-First-Out Prinzip
  - Anwendung z.B. bei Implementierung von Programmiersprachen
- Operationen
  - createStack: liefert einen leeren Stack
  - push: fügt ein Element in den Stack ein
  - pop: entfernt das zuletzt eingefügte Element
  - top: liefert das zuletzt eingefügte, nicht gelöschte Element
  - empty: gibt an, ob Stack leer ist
- Beispiele
  - push( push( push (createStack, 1), 2), 3)
  - push( pop( push( push( createStack, 1), 2)), 3)

# Spezifikation Stack

- Definition
  - Signatur  $\Sigma = (S, F)$
  - Sorten  $S: \{\text{Stack}, \text{Element}, \text{BOOL}\}$
  - Welche Strukturbeschreibung haben die Operationen  $F$ ?
  - Welche Axiome gelten?
  - |              |                                      |               |         |
|--------------|--------------------------------------|---------------|---------|
| createStack: | $\rightarrow$                        | Stack         |         |
| push:        | $\text{Stack} \times \text{Element}$ | $\rightarrow$ | Stack   |
| pop:         | Stack                                | $\rightarrow$ | Stack   |
| top:         | Stack                                | $\rightarrow$ | Element |
| empty:       | Stack                                | $\rightarrow$ | BOOL    |
  - K1: empty (createStack) = true  
K2: empty (push(k, t)) = false  
K3: pop(push(k, t)) = k  
K4: top( push(k, t)) = t

# Spezifikation Natürliche Zahlen

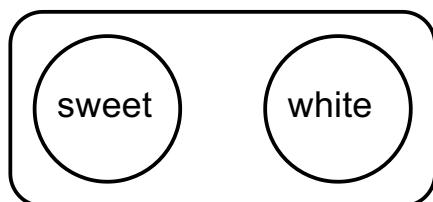
- Definition
  - Signatur  $\Sigma = (S, F)$
  - Sorten  $S: \{\text{Nat}\}$
  - zero:  $\rightarrow \text{Nat}$
  - succ:  $\text{Nat} \rightarrow \text{Nat}$
  - add:  $\text{Nat} \times \text{Nat} \rightarrow \text{Nat}$
  - mult:  $\text{Nat} \times \text{Nat} \rightarrow \text{Nat}$
  - K1:  $\text{add}(\text{zero}, n) = n$
  - K2:  $\text{add}(\text{succ}(n), m) = \text{succ}(\text{add}(n, m))$
  - K3:  $\text{mult}(\text{zero}, n) = \text{zero}$
  - K4:  $\text{mult}(\text{succ}(n), m) = \text{add}(m, \text{mult}(n, m))$

# Spezifikation Mengen

- Strukturbeschreibung von Mengen von Objekten
  - Sorten S: {Set, Object, Bool, Integer}
  - newSet:                                   → Set
  - empty:   Set                               → Bool
  - size:      Set                               → Integer
  - contains: Set × Object                   → Bool
  - add:       Set × Object                   → Set
  - remove: Set × Object                   → Set
  - equal:     Set × Set                       → Bool
  - subset:   Set × Set                       → Bool

# Spezifikation Getränkeautomat

- Beschreibung Getränkeautomat
  - Zwei Knöpfe *sweet* und *white*, zur Auswahl von Zucker oder Milch oder beides
  - Eine bereits getroffene Auswahl kann durch nochmaliges drücken zurückgenommen werden
  - Die Reihenfolge der Knöpfe ist gleichgültig



# Spezifikation Getränkeautomat

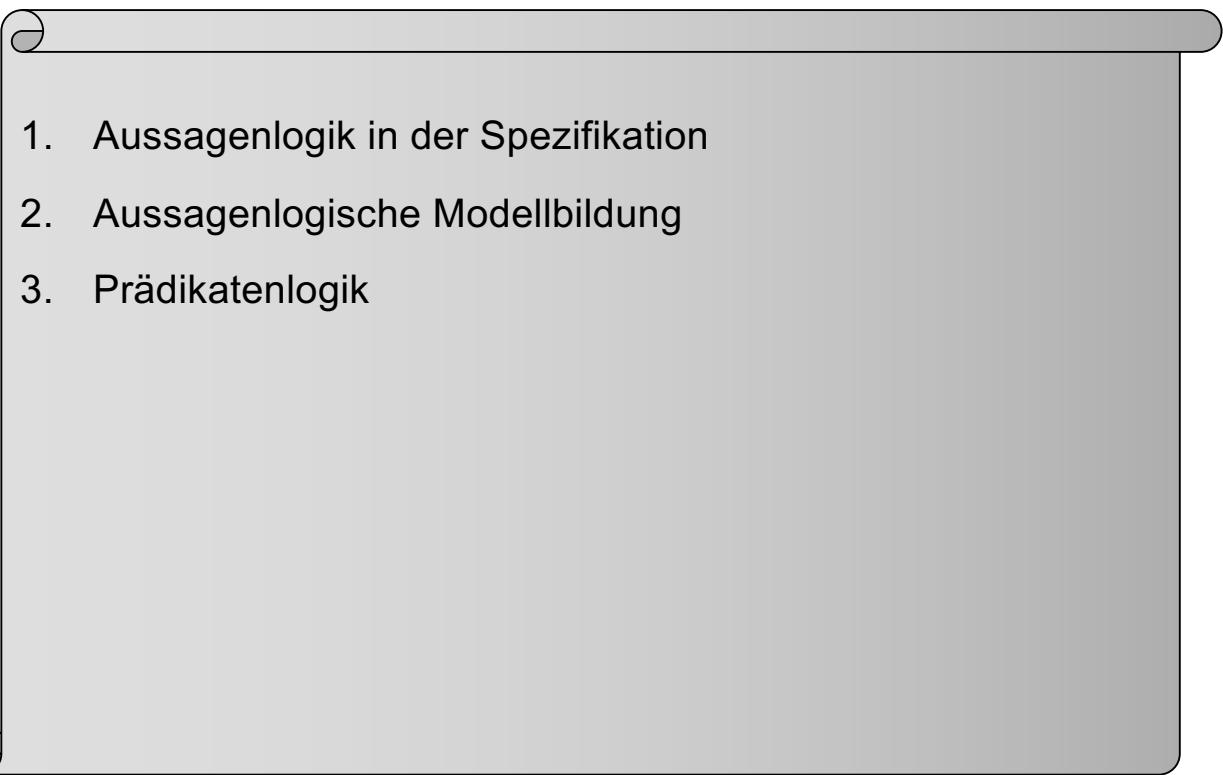
- Algebraische Spezifikation Getränkeautomat
  - Sorten  $s = \{\text{Add}, \text{Choice}\}$
  - Operationen:

sweet:	$\rightarrow$	Add
white:	$\rightarrow$	Add
noChoice:	$\rightarrow$	Choice
press:      Add $\times$ Choice	$\rightarrow$	Choice
  - Ein Term beschreibt eine Folge von Tastendrücken  
 $\text{press}(\text{sweet}, \text{press}(\text{white}, \text{press}(\text{sweet}, \text{noChoice})))$
  - Axiome:
    - Q1:  $\text{press}(a, \text{press}(a, c)) = c$
    - Q2:  $\text{press}(\text{sweet}, \text{press}(\text{white}, c)) = \text{press}(\text{white}, \text{press}(\text{sweet}, c))$

# Spezifikation Getränkeautomat

- Terme in Normalform
  - noChoice
  - press (white, noChoice)
  - press (sweet, noChoice)
  - press (white, press (sweet, noChoice))
  - Alle anderen Terme können in diese vier Terme umgeformt werden

## 3 Logik

- 
1. Aussagenlogik in der Spezifikation
  2. Aussagenlogische Modellbildung
  3. Prädikatenlogik

# Aussagenlogik in der Spezifikation

## Beispiel

- Unfall durch fehlerhafte Spezifikation
  - Airbus A320, Warschau (1993): Der zuständige Rechner blockiert bei der Landung die Aktivierung der Schubumkehr und Störklappen, wodurch das Flugzeug über das Landebahnende hinausschießt. Es herrschen starker Wind von schräg hinten und Aquaplaning auf der Landebahn.
- Beabsichtigte Spezifikation der Störfreigabe
  - Die Störklappen dürfen benutzt werden
    - im Reise- und Sinkflug (Bremswirkung)
    - nach der Landung (Vernichtung des Auftriebs und Bremswirkung)
  - Sie dürfen nicht benutzt werden
    - im Endanflug (gefährlicher Auftriebsverlust)

Quelle: Kastens, Kleine Büning: Modellierung

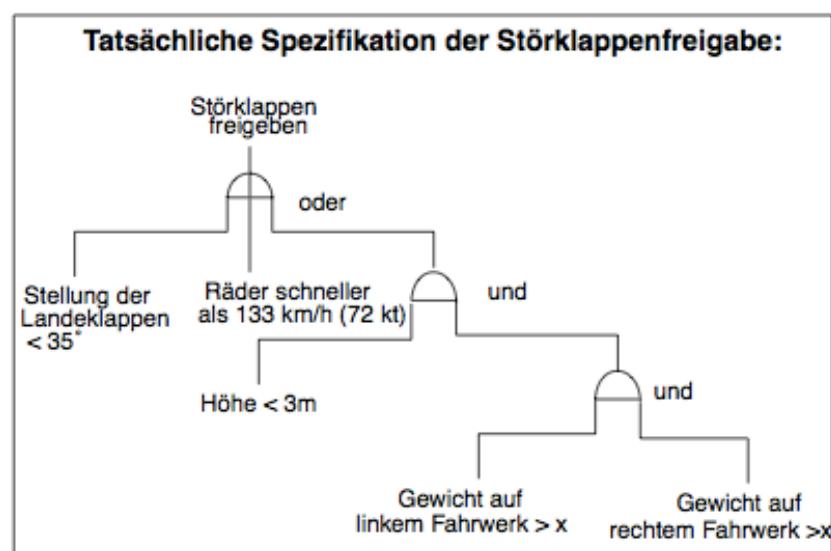


Quelle: wikipedia.de

# Aussagenlogik in der Spezifikation

## Beispiel

- Unvollständige Spezifikation in deutscher Sprache
  - Die Störklappen werden freigegeben, wenn die Stellung der Landeklappen  $< 35^\circ$  oder Räder schneller als 133 km/h sind oder die Höhe  $< 3\text{m}$  und Gewicht auf den Fahrwerken  $> x$  ist



Quelle: Kastens, Kleine Büning: Modellierung

# Aussagenlogik in der natürlichen Sprache

- Ausschnitt aus der AGB eines Internet-Portals
  - Der Kunde hat die Kosten der Rücksendung zu tragen, wenn die gelieferte Ware der bestellten entspricht und wenn der Preis der zurückzusendenden Sache einen Betrag von 40,00 EUR nicht übersteigt oder wenn bei einem höheren Preis dieser Sache zum Zeitpunkt des Widerrufs die Gegenleistung oder eine vertraglich vereinbarte Teilzahlung noch nicht erbracht ist.  
Andernfalls ist die Rücksendung durch den Kunden kostenfrei.
- Wie ist die Klammerung in dieser Beschreibung?

# Aussagenlogik

- Aussagen
  - Sätze, die prinzipiell als wahr oder falsch angesehen werden können
  - Beispiel: „Es regnet“, „Die Strasse ist naß“
- Junktoren verknüpfen Aussagen
  - Es regnet nicht, **oder** die Straße ist nass
- Belegung mit Wahrheitswerten
  - Regen → straßeNass

W                    W  
                      W

# Aussagenlogik

- Grundsymbole
  - Variablen
  - Funktionssymbole.  
0-stelliges Funktionssymbol wird Konstante genannt
  - Junktoren  $\wedge$  (und),  $\vee$  (oder),  $\rightarrow$  (Implikation, wenn-dann),  
 $\leftrightarrow$  Äquivalenz (genau dann, wenn)
- Präzedenzregeln
  - $\neg$  bindet stärker als  $\wedge$
  - $\wedge$  bindet stärker als  $\vee$
  - $\vee$  bindet stärker als  $\rightarrow$  und  $\leftrightarrow$
- Weitere Definitionen
  - Elementaraussagen bezeichnet man auch als **Atome**
  - Ein **Literal** ist ein Atom  $A$  oder ein negiertes Atom  $\neg A$

# Aussagenlogik

- Aussagenlogische Formeln: Terme mit Variablen zur Signatur der boolschen Algebra
  - false:  $\rightarrow \text{BOOL}$
  - true:  $\rightarrow \text{BOOL}$
  - $\wedge: \text{BOOL} \times \text{BOOL} \rightarrow \text{BOOL}$
  - $\vee: \text{BOOL} \times \text{BOOL} \rightarrow \text{BOOL}$
  - $\neg: \text{BOOL} \rightarrow \text{BOOL}$
- Erweiterung
  - $\rightarrow: \text{BOOL} \times \text{BOOL} \rightarrow \text{BOOL}$       Implikation
  - $\leftrightarrow: \text{BOOL} \times \text{BOOL} \rightarrow \text{BOOL}$       Äquivalenz

# Bewertung eines Terms

- $\Im$  sei Bewertung von Formeln
  - $\Im(\neg\alpha) = w$  genau dann, wenn  $\Im(\alpha)=f$
  - $\Im(\alpha \wedge \beta) = w$  genau dann, wenn  $\Im(\alpha)=w$  und  $\Im(\beta)=w$
  - $\Im(\alpha \vee \beta) = w$  genau dann, wenn  $\Im(\alpha)=w$  oder  $\Im(\beta)=w$
  - $\Im(\alpha \rightarrow \beta) = w$  genau dann, wenn  $\Im(\alpha)=f$  oder  $\Im(\beta)=w$
  - $\Im(\alpha \leftrightarrow \beta) = w$  genau dann, wenn  $\Im(\alpha)=\Im(\beta)=w$  oder  $\Im(\alpha)=\Im(\beta)=f$
  - $\Im(\text{true}) = w$  und  $\Im(\text{false})=f$
- Wahrheitstafel

$\alpha$	$\beta$	$\neg\alpha$	$\alpha \wedge \beta$	$\alpha \vee \beta$	$\alpha \rightarrow \beta$	$\alpha \leftrightarrow \beta$	$\alpha \underline{\vee} \beta$
W	W	F	W	W	W	W	F
W	F	F	F	W	F	F	W
F	W	W	F	W	W	F	W
F	F	W	F	F	W	W	F

# Umformungsregeln

Negation	$\neg\neg\alpha \approx \alpha$
Idempotenz	$\alpha v \alpha \approx \alpha$ $\alpha \wedge \alpha \approx \alpha$
Kommutativität	$\alpha v \beta \approx \beta v \alpha$ $\alpha \wedge \beta \approx \beta \wedge \alpha$
Assoziativität	$(\alpha v \beta) v \gamma \approx \alpha v (\beta v \gamma)$ $(\alpha \wedge \beta) \wedge \gamma \approx \alpha \wedge (\beta \wedge \gamma)$
Distributivität	$(\alpha \wedge \beta) v \gamma \approx (\alpha v \gamma) \wedge (\beta v \gamma)$ $(\alpha v \beta) \wedge \gamma \approx (\alpha \wedge \gamma) v (\beta \wedge \gamma)$
De Morgan	$\neg(\alpha \wedge \beta) \approx \neg \alpha \vee \neg \beta$ $\neg(\alpha \vee \beta) \approx \neg \alpha \wedge \neg \beta$
Komplement	$\alpha v \neg \alpha \approx \text{true}$ $\alpha \wedge \neg \alpha \approx \text{false}$
Neutrale Elemente	$\alpha \wedge \text{true} \approx \alpha$ $\alpha \wedge \text{false} \approx \text{false}$ $\alpha v \text{true} \approx \text{true}$ $\alpha v \text{false} \approx \alpha$

# Weitere Umformungsregeln

Implikation	$\alpha \rightarrow \beta$ $\approx \neg \alpha \vee \beta$ $\approx \beta \vee \neg \alpha$ $\approx \neg \beta \rightarrow \neg \alpha$
Äquivalenz	$\alpha \leftrightarrow \beta \approx (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
Exklusives Oder	$\alpha \underline{\vee} \beta \approx (\alpha \wedge \neg \beta) \vee (\neg \alpha \wedge \beta)$ $\alpha \underline{\vee} \beta \approx (\alpha \vee \beta) \wedge (\neg \alpha \vee \neg \beta)$

# Wahrheitstafel

- Wahrheitstafel für komplexere Formeln
  - Definieren Sie die Wahrheitstafel für  $(A \vee \neg B) \wedge (A \vee B \rightarrow C)$

A	B	C	$(A \vee \neg B)$	$(A \vee B) \rightarrow C$	$(A \vee \neg B) \wedge (A \vee B \rightarrow C)$
f	f	f	w	w	w
f	f	w	w	w	w
f	w	f	f	f	f
f	w	w	f	w	f
w	f	f	w	f	f
w	f	w	w	w	w
w	w	f	w	f	f
w	w	w	w	w	w

# Normalformen

- Konjunktive Normalform KNF
  - Eine Formel ist in **konjunktiver Normalform**, wenn sie eine Konjunktion von Disjunktionen von Literalen ist
  - Beispiel:  $(A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee \neg B)$
- Disjunktive Normalform DNF
  - Eine Formel ist in **disjunktiver Normalform**, wenn sie eine Disjunktion von Konjunktionen von Literalen ist
  - Beispiel:  $(A \wedge B) \vee (A \wedge \neg B) \vee (\neg A \wedge \neg B)$
- Zu jeder aussagenlogischen Formel gibt es
  - eine äquivalente Formel in KNF
  - eine äquivalente Formel in DNF
- Beispiel
  - Überführen Sie  $A \leftrightarrow B$  in die DNF und KNF

# Erfüllbar, tautologisch, falsifizierbar

- Definitionen
  - Eine Formel  $\alpha$  ist **erfüllbar** genau dann, wenn es eine Belegung gibt, bei der sie true ist
  - Eine Formel  $\alpha$  ist **tautologisch** (**eine Tautologie, allgemeingültig**) genau dann, wenn sie für jede Belegung true ist
  - Eine Formel  $\alpha$  ist **widerspruchsvoll (unerfüllbar)** genau dann, wenn sie für jede Belegung false ist
  - Eine Formel  $\alpha$  ist **falsifizierbar** genau dann, wenn es eine Belegung gibt, bei der sie false ist
- Aufgaben
  - Geben Sie jeweils ein Beispiel an
  - Was gilt für die Formel aus dem vorherigen Beispiel Folie 11?
  - Ist folgender Satz eine Tautologie?  
Wenn der Hahn kräht auf dem Mist, ändert sich das Wetter oder es bleibt wie es ist

# Aussagenlogische Modellbildung

- Aussagenlogik zur Modellierung von statischem Wissen
  - Dynamische Systeme werden normalerweise nicht mit Aussagenlogik modelliert
  - Die Aussage „Es regnet“ wird dem Atom  $R$  und „Die Straße ist naß“ dem Atom  $S$  zugeordnet

Es regnet nicht	$\neg R$
Es regnet oder die Straße ist nass	$R \vee S$
Es regnet und die Straße ist nass	$R \wedge S$
Wenn es regnet, ist die Straße nass	$R \rightarrow S$
Genau dann, wenn es regnet, ist die Straße nass	$R \leftrightarrow S$

# Aussagenlogische Modellbildung

Die Straße ist nass genau dann, wenn es regnet	$R \leftrightarrow S$
Die Straße ist nass dann und nur dann, wenn es regnet	$R \leftrightarrow S$
Entweder ist die Straße nass oder es regnet	$R \vee S$ $(R \vee S) \wedge (\neg R \vee \neg S)$ $(R \wedge \neg S) \vee (\neg R \wedge S)$

- Gegeben ist die Aussage „Wenn es regnet, ist die Strasse naß“
  - Formalisieren Sie diese Aussage
  - Was sagt diese Aussage aus, wenn die Strasse naß ist?
  - Was sagt diese Aussage aus, wenn die Strasse nicht naß ist?
- Formalisieren Sie die Aussage „Bei Nacht sind alle Katzen grau“

# Formalisierung umgangssprachlicher Ausdrücke

- Vorsicht bei Implikationen; mit Belegungen prüfen, was gemeint ist

<b>Wenn</b> es regnet, benutze ich den Schirm	regnet → schirm
Ich benutze den Schirm, <b>wenn</b> es regnet	regnet → schirm
Ich benutze den Schirm <b>nur wenn</b> es regnet	schirm → regnet

- „Oder“ kann fast immer in das  $\vee$  (xor) übersetzt werden

Hast Du einen Euro <b>oder</b> zwei Fünfziger?	euro $\vee$ zweiFünfziger
Morgen fahr ich mit dem Auto <b>oder</b> dem Zug nach Berlin.	zug $\vee$ auto
x ist kleiner y <b>oder</b> y ist kleiner x	$x < y \vee y < x$
Der Händler gibt Rabatt <b>oder</b> ein kostenloses Radio	rabatt $\vee$ radio

# Formalisierung umgangssprachlicher Ausdrücke

- Aussagen sind häufig kontext-abhängig

Weil das Wetter schön ist, gehe ich nicht in die Vorlesung	$\text{Wetter\_schön} \wedge \neg\text{Vorlesung\_besuchen}$
Weil ich die Matheklausur nicht bestanden habe, nehme ich am zweiten Versuch teil	$\neg\text{Mathe-bestanden} \wedge \text{Mathe-Versuch2}$

- Klammern sind meist nur aus dem Kontext erkennbar

Sie wollten <b>nicht</b> verlieren <b>oder</b> unentschieden spielen	$\neg(\text{verlieren} \vee \text{unentschieden})$ <i>oder</i> $\neg\text{verlieren} \vee \text{unentschieden}$
--	---

→ Vorsicht beim Übertragen von Umgangssprache in Formeln, insbesondere bei Klammern und Implikationen

# Aufgabe

“Wie hast du es geschafft, eine Eins in der Modellierungsklausur zu schreiben?”

“Das war gar nicht so schwer. Ich habe mich einfach an folgende Grundsätze gehalten:”

- Wenn ich mal nicht in der Übung war, bin ich in die Vorlesung gegangen.
  - Immer wenn ich in der Übung und in der Vorlesung war, habe ich nicht ins Modellierungsbuch geschaut.
  - Wenn ich ins Modellierungsbuch geschaut habe oder nicht in der Übung war, war ich nicht in der Vorlesung.
1. Formalisieren Sie die drei Aussagen. Verwenden Sie dazu die angegebenen Variablenamen:
    - ue      Ich war in der Übung
    - vo      Ich war in der Vorlesung
    - bu      Ich habe ins Modellierungsbuch geschaut
  2. Interpretieren Sie die Formeln für alle Belegungen von *ue*, *vo*, und *bu*. Verwenden Sie dazu eine Wahrheitstafel.
  3. Finden Sie mit Hilfe der Wahrheitstafel eine einfache Formulierung für den doch recht konfusen Ratschlag des Modellierungsstudierenden.

Quelle: <http://ag-kastens.upb.de>

# Lösung

- Formalisierung der Aussagen
  - $\neg ue \rightarrow vo$
  - $ue \wedge vo \rightarrow \neg bu$
  - $bu \vee \neg ue \rightarrow \neg vo$
- Wahrheitstafel

$ue$	$vo$	$bu$	$\neg ue$	$\neg ue \rightarrow vo$	$ue \wedge vo$	$\neg bu$	$ue \wedge vo \rightarrow \neg bu$	$bu \vee \neg ue$	$\neg vo$	$bu \vee \neg ue \rightarrow \neg vo$
f	f	f	w	f	f	w	w	w	w	w
f	f	w	w	f	f	f	w	w	w	w
f	w	f	w	w	f	w	w	w	f	f
f	w	w	w	w	f	f	w	w	f	f
w	f	f	f	w	f	w	w	f	w	w
w	f	w	f	w	f	f	w	w	w	w
w	w	f	f	w	w	w	w	f	f	w
w	w	w	f	w	w	f	f	w	f	f

# Prädikatenlogik

- Erweiterung um parametrisierte Elementaraussagen
- n-stellige Prädikate sind Operationen  $P: T^n \rightarrow \text{BOOL}$   
In Konkretisierung entsprechen ihnen n-stellige Relationen
  - „x ist eine Katze“ als Formel: istKatze(x)
  - „a teilt b“ als Formel: teilt(a, b)
- 0-stellige Prädikate als Konstante
  - istKatze(molly)
  - istKatze(peterle)
- Informale Definition
  - Individuen als Grundobjekte von Strukturen
  - Ihre Menge als Individuenbereich oder „Universum“
  - Konstante zur Bezeichnung ausgezeichneter Individuen

# Prädikatenlogik

- Quantoren
  - „für alle  $x$  gilt  $\alpha$ “, in Symbolen:  $\forall x \alpha$  (Allquantor)
  - „es gibt ein  $x$ , so daß  $\alpha$  gilt“, in Symbolen:  $\exists x \alpha$  (Existenzquantor)
  - Wenn  $\alpha$  eine Formel darstellt, dann sind auch  $\exists x \alpha$  und  $\forall x \alpha$  Formeln
  - Variable sind nur als Operanden in Terme erlaubt, nicht für Funktionen oder Prädikate (Prädikatenlogik erster Stufe)

# Prädikatenlogik

- Wirkungsbereich von Quantoren
  - $\forall x (P(x) \wedge Q(x)) \vee \exists y (P(y) \wedge \exists x R(y, x))$   

- Freie und gebundene Variablen
  - Ein Vorkommen von  $x$  in einer Formel  $\alpha$  heißt **frei**, wenn es nicht im Wirkungsbereich für  $x$  eines Quantors liegt
  - Ein Quantor  $\forall x \alpha$  bzw.  $\exists x \alpha$  binden alle  $x$ , die frei sind in  $\alpha$ . Die Variable  $x$  heißt dann **gebunden**
  - Beispiel: Formel  $\alpha = (R(y) \wedge \exists y (P(y, x) \vee Q(y, z))$   
 $\alpha$  hat drei freie Variable  $x, y, z$ .  $y$  kommt frei und gebunden vor
  - Eine Formel ist eine **Aussage**, genau dann wenn sie keine freie Variablen enthält
  - Gebundene Namen können umbenannt werden

# Prädikatenlogik

- Für Formeln gilt
    - $\mathfrak{I}(\exists x \alpha)=w$  genau dann, wenn es ein  $x_u \in U$  gibt mit  $\mathfrak{I}[x/x_u](\alpha)=w$
    - $\mathfrak{I}(\forall x \alpha)=w$  genau dann, wenn für jedes  $x_u \in U$  gilt  $\mathfrak{I}[x/x_u](\alpha)=w$
  - Anwendung in der Modellierung
    - Schon für einfache Aufgaben benötigt man Prädikatenlogik
  - Beispiele
    - Hans liebt Gaby
    - liebt(Hans, Gaby)
- Alle lieben Gaby
- $\forall x (\text{liebt}(x, \text{Gaby}))$

# Prädikatenlogik

- Allquantifizierte Aussagen
  - Aussage: Alle Hunde bellen
  - Formulierung: wenn etwas ein Hund ist, bellt es.  
 $\forall x \text{ Hund}(x) \rightarrow \text{bellt}(x)$
  - Andere Formulierung: es gibt keinen Hund der nicht bellt  
 $\neg \exists x \text{ Hund}(x) \wedge \neg \text{bellt}(x)$
  - Wie lassen sich die beiden Formeln umformen?
- Weitere Beispiele
  - Alle Tiere im Zoo sind Raubtiere
  - Schalke-Fans hassen Dortmund-Fans

# Modellierung von Geschäftsregeln

- Beispiel 1
  - „An alle Kunden, deren Mailadresse verfügbar ist, soll eine Mail geschickt werden“
  - $\forall x \text{ Kunde}(x) \wedge \text{MailadresseVerfügbar}(x) \rightarrow \text{MailSchickenAn}(x)$
- Beispiel 2
  - „Jeder Kunde ist per E-Mail oder Telefon erreichbar“
  - $\forall x \text{ Kunde}(x) \rightarrow \text{Erreichbar}(x, \text{telefon}) \vee \text{Erreichbar}(x, \text{email})$

Quelle: U. Hettstück: Complex Event Processing

# Umformungen prädikatenlogischer Formeln

- Negation

- $\neg \forall x \alpha \approx \exists x \neg \alpha$
- $\neg \exists x \alpha \approx \forall x \neg \alpha$
- Beispiel: Alle haben den Schuss gehört  
Negiert: Es gibt einen, der den Schuss nicht gehört hat  
Falsch negiert: Keiner hat den Schuss gehört

- Quantoren vertauschen

- $\forall x \forall y \alpha \approx \forall y \forall x \alpha$
- $\exists x \exists y \alpha \approx \exists y \exists x \alpha$
- Vorsicht, **nicht** äquivalent:  
 $\forall s \exists v \text{ bestanden}(s, v) \not\approx \exists v \forall s \text{ bestanden}(s, v)$
- Alle haben eine Prüfung bestanden  
ist nicht das gleiche wie  
Es gibt eine Prüfung die alle bestanden haben

# Umformungen prädikatenlogischer Formeln

- Quantoren zusammenfassen

- $\forall x \alpha \wedge \forall x \beta \approx \forall x \alpha \wedge \beta$        $\exists x \alpha \vee \exists x \beta \approx \exists x \alpha \vee \beta$

- Vorsicht, **nicht** äquivalent:

- $\forall x \alpha \vee \forall x \beta \not\approx \forall x \alpha \vee \beta$        $\exists x \alpha \wedge \exists x \beta \not\approx \exists x \alpha \wedge \beta$

- Beispiel

- $\forall x \text{gerade}(x) \vee \text{ungerade}(x)$   
 $\not\approx \forall x \text{gerade}(x) \vee \forall x \text{ungerade}(x)$

- $\exists s \text{bestanden}(s, \text{Mathe}) \wedge \exists s \text{bestanden}(s, \text{SYMO})$   
 $\not\approx \exists s \text{bestanden}(s, \text{Mathe}) \wedge \text{bestanden}(s, \text{SYMO})$

# Modellierung in Prädikatenlogik

- Einheiten der Prädikatenlogik
  - Terme repräsentieren Objekte
  - Prädikatsymbole repräsentieren Eigenschaften oder Relationen
  - Formeln repräsentieren Aussagen
- Beispiel
  - Atomare Terme: M steht München dar, HH Hamburg, d für Deutschland
  - Funktionssymbole: ewz steht für Einwohnerzahl, hauptstadt für Hauptstadt
  - Zweistelliges Prädikatsymbole: Nördlich von steht für „nördlich von“, Komplexere Terme: ewz(hauptstadt(d)) > ewz(m)

# Übersetzung Deutsch nach Prädikatenlogik

- Beispiele

- Eine Stadt liegt nördlich von München
- $\exists$  Stadt Nördlichvon münchen
- $\exists x (\text{Stadt}(x) \wedge \text{Stadt}(m) \wedge \text{Nördlichvon}(x, \text{münchen}))$
- Keine Stadt liegt nördlich von sich selbst
- $\neg \exists$  Stadt Nördlichvon
- $\neg \exists x \text{ Stadt}(x) \wedge \text{Nördlichvon}(x, x)$
- Jede Stadt ist groß
- $\forall$  Stadt Gross
- $\forall x \text{ Stadt}(x) \rightarrow \text{Gross}(x)$

# Formalisierung in Prädikatenlogik

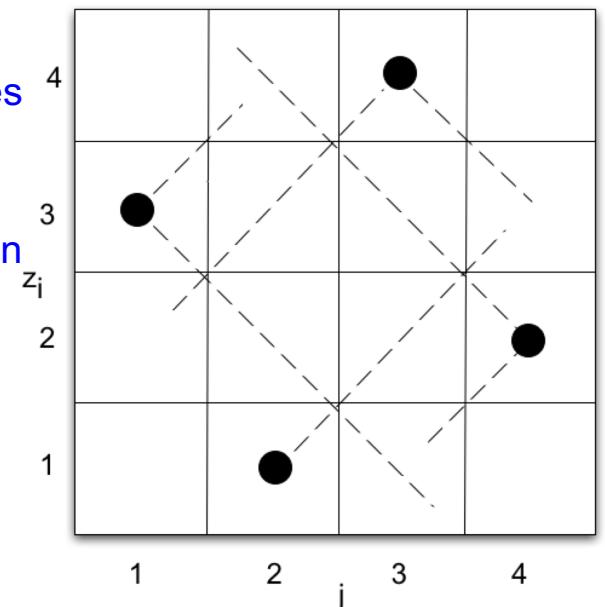
- Modellieren Sie folgende Aussagen
  - Alle Menschen sind sterblich
  - Menschen sind sterblich
  - AIN-Studierende sind schlauer
  - Jeder im Raum spricht deutsch oder englisch
  - „Die dümmsten Bauern haben die dicksten Kartoffeln“ mit den Beziehungen „dümmer“ und „erntet“ und „dicker“
- Menschen vereinfachen Sätze, so dass sie gerade noch verstanden werden
  - Vollständiger Beispielsatz: „Es gibt Gäste in der Bar, die alle Getränke trinken, die in der Bar angeboten werden.“
  - Vereinfachung mit gleicher Bedeutung: „Manche trinken alles“

# Formalisierung in Prädikatenlogik

- Weitere Beispiele
  - Definieren Sie anhand der Beziehungen Vater und Mutter die Beziehungen Geschwister
  - Definieren Sie anhand der Beziehungen Vater und Mutter die Beziehungen Großvater und Großmutter
  - Definieren Sie anhand der Beziehungen Vater und Mutter die Beziehung Enkel
  - Einfache Umsetzung in der Sprache Prolog
    - Definition größter gemeinsamer Teiler  $g$  von  $a$  und  $b$ .  $a, b \in \mathbb{N}_0$
    - $\text{teilt}(g,a) \wedge \text{teilt}(g,b) \wedge (\forall h(\text{teilt}(h,a) \wedge \text{teilt}(h,b) \rightarrow h \leq g))$

# Spezifikation des n-Damen-Problems

- Beschreibung
  - Gegeben: Kantenlänge  $n \in \mathbb{N}_0$  eines  $n \times n$  Schachbretts
  - Gesucht: Menge  $P$  der Platzierungen von jeweils  $n$  Damen auf dem Schachbrett, so dass keine Dame eine andere nach Schachregeln schlägt.
  - Wie lassen sich die Platzierungen beschreiben?
  - Wie werden die zulässigen Platzierungen definiert?
  - Index =  $\{1, \dots, n\}$   $P := \{p \mid p = (z_1, \dots, z_n) \in \text{Index}^n \wedge \text{zulässig}(p)\}$
  - $\text{zulässig}(p)$ :  $\forall i, j \in \text{Index}: i \neq j \rightarrow z_i \neq z_j \wedge |z_i - z_j| \neq |i - j|$



## 4 Zeichenfolgen

1. Reguläre Ausdrücke
2. Backus Naur Form

# Literatur, Quellen

- Friedl, J.E.F.: Reguläre Ausdrücke - 3. Aufl., Beijing ; Köln [u.a.] : O'Reilly, 2008.
- Online-Test für reguläre Ausdrücke zum einfachen Überprüfen der Übungsaufgaben
  - [www.regexr.com](http://www.regexr.com)
  - [regexpal.com](http://regexpal.com)
  - [https://regex101.com mit Erklärungen](https://regex101.com)

# Reguläre Ausdrücke

- Regeln zur Beschreibung von Zeichenketten
- Unterstützung durch zahlreiche Programmiersprachen, Tools
- Bezeichnung:
  - Englisch: regular expression
  - Abgekürzt: Regex
- Einführung
  - Ein regulärer Ausdruck beschreibt eine Menge von Worten durch Regeln
  - Typ-3 in Chomsky-Hierarchie
  - Wird durch endlichen Automat akzeptiert

# Reguläre Ausdrücke

- **Definition**
  - $a$  definiert eine Menge, die nur das Zeichen  $a$  als Wort enthält
  - $F \cdot G$  definiert Worte, die jeweils aus einem Wort aus  $f \in F$  und  $g \in G$  zusammengesetzt sind
  - $F|G$  definiert Worte, die aus  $F$  oder  $G$  stammen
  - Häufige Erweiterungen:

$Y^+$	eine nicht-leere Folge von $Y$
$Y^*$	eine evtl. leere Folge von $Y$
$Y^?$	optional $Y$
$[abc]$	eines der Zeichen $a$ , $b$ oder $c$
$Y^{\{m,n\}}$	eine Folge von $Y$ mit mind. $m$ und höchstens $n$ Elementen
$Y^{\{m,\}}$	eine Folge von $Y$ mit mind. $m$ Elementen
$Y^{\{n\}}$	eine Folge von $Y$ mit genau $n$ Elementen

# Reguläre Ausdrücke

- Beispiele

- Ausdruck besteht aus beliebig vielen a oder b
    - $(a+|b+)^*$
    - $(a|b)^*$
    - $[ab]^*$
  - Ausdruck fängt mit a an und endet mit b
    - $a(a|b)^*b$
  - Beispiel: Reguläre Ausdrücke für Zahlen
    - $(0|1|2\dots|9)$  oder  $[0-9]$
    - $[0-9]^*$
    - $[0-9]^*.[0-9]^*$
    - $[0-9]^*.(.[0-9]^*)?$

# Reguläre Ausdrücke

- Eckige Klammern
  - $[abcd]$  steht für  $a|b|c|d$
  - $[0-9]$  steht für  $0|1|2|3|4|5|6|7|8|9$
  - $[a-zA-Z]$  steht für alle Klein- und Großbuchstaben
  - Maskierung für “-“:  $[0-9\text{-}]$
- Zeichenklassen negieren
  - $[^aeiou]^+$  steht für alle Zeichenfolgen, die a,e,i,o,u nicht enthalten
  - $[^0-5]$  steht für ein Zeichen, das nicht im Bereich 0-5 ist

# Reguläre Ausdrücke in der Programmierung

- Einige vordefinierte Zeichenklassen
  - Nicht von allen Implementierungen in gleicher Weise unterstützt

.	Beliebiges Zeichen
\.	Punkt
\\	Backslash
\n	Newline
\d	Ziffer, identisch mit [0-9]
\D	Keine Ziffer, identisch mit [^0-9]
\s	Leerraumzeichen (z.B. Space, Tabulator)
\S	Kein Leerraumzeichen, identisch mit [^\s]
\w	Alphanumerisches Zeichen, identisch mit [a-zA-Z_0-9]
\W	Kein alphanumerische Zeichen, identisch mit [^\w]
^	Beginn des Textes
\$	Ende des Textes

# Reguläre Ausdrücke

- Aufgaben
  - Definieren Sie die Anrede einer Person als regulären Ausdruck
  - Definieren Sie eine Mailadresse als regulären Ausdruck
  - Definieren Sie eine Adresse inkl. Strasse Hausnummer, Postleitzahl, Stadt als regulären Ausdruck
- Verwendung regulärer Ausdrücke
  - C, Java, PHP
  - Unix-Skripte wie egrep, sed
  - Webseiten wie z.B. <http://www.regexe.de>
  - Java siehe nächste Folie

# Reguläre Ausdrücke in der Programmierung

- Verwendung regulärer Ausdrücke in Java

- Klasse String

```
public String[] split(String regex)
public boolean matches(String regex)
public String replaceAll(String regex,
                        String replacement)
```

- Klasse Pattern

```
static boolean matches(String regex,
                      CharSequence input)
```

- Beispiel

```
Pattern.matches(
    "[a-zA-Z]{2,20}\.?[0-9]{1,3}[a-zA-Z]?",
    "Brauneggerstr. 55" )
```

# Backus Naur Form (BNF)

- Beschreibung
  - Backus Naur Form (BNF) zur Beschreibung von kontextfreien Grammatiken
  - Ersetzungssysteme
  - Definition der Sprache als Menge von Sätzen, die mit Regeln erzeugt werden können
- Anwendung für
  - Programmiersprachen, wie z.B. Java, C
  - Sprachen als Schnittstelle, z.B. HTML, XML
  - Strukturen von Protokollen

## HTML-Beispiel

```
<table>
  <tr>
    <td>Mo</td>
    <td>11-13</td>
    <td>AM</td>
  </tr>
  <tr>
    <td>Fr</td>
    <td>11-13</td>
    <td>AM</td>
  </tr>
</table>
```

# Definition BNF

- Kontextfreie Grammatik  $G = (T, N, P, S)$  besteht aus
  - $T$  Menge der Terminalsymbole (Terminale)  
entspricht den Elementen der Sprache
  - $N$  Menge der Nichtterminalsymbole (Nichtterminale)  
wird in Regeln definiert
  - $S \in N$  Startsymbol
  - $P \subseteq N \times V^*$  Menge der Produktionen,  $V = T \cup N$   
 $(A, x) \in P$  mit  $A \in N$ ,  $x \in V^*$   
statt  $(A, x) \in P$  schreibt man  $A ::= x$
- Unterscheidung Terminalsymbole und Nichtterminalsymbole
  - Verwendung von spitzen Klammern für Nichtterminalsymbole
  - Oder: Anführungsstriche für Terminalsymbole

© Prof. Dr. Uwe Kastens Modellierung

# Definition BNF

- Beispiele

- <DeutscherSatz> ::= <Subjekt> <Prädikat> <Objekt>
  - <Subjekt> ::= <Artikel> <Substantiv>
  - <Objekt> ::= <Artikel> <Substantiv>
  - <Substantiv> ::= "Mann" | "Buch"
  - <Artikel> ::= "der" | "die" | "das"
  - <Prädikat> ::= "liest"
- ZifferAußerNull ::= "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
- Ziffer ::= "0" | ZifferAußerNull
- Klammerung ::= "(" Liste ")"
  - Liste ::= Klammerung Liste
  - Liste ::=

---

© Prof. Dr. Uwe Kastens Modellierung

# Häufige BNF-Definitionen

- Definition Identifier
  - `<identifier> ::= <letter> { <idchar> }`
  - `<idchar> ::= <letter> | <digit> | _`
  - `<letter> ::= A | B | C | ... | Y | Z | a | b | c | ... | y | z`
  - `<digit> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9`
- Definition Integer
  - `<integer> ::= <digit> { <digit> }`
  - `<digit> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0`
- Definition String
  - `<string> ::= " <char> { <char> } "`
- Diese Definitionen werden in der Praxis häufig als bekannt vorausgesetzt und nicht jedes mal erneut definiert

# Ableitungsregel

- Definition Ableitung
  - Sei  $G=(T, N, P, S)$  eine kontextfreie Grammatik. Dann kann man mit der Produktion  $A ::= x \in P$  das Nichtterminal  $A \in N$  durch die rechte Seite  $x$  ersetzen. Das ist eine **Ableitungsregel**.
  - Mehrere Ableitungsregeln nacheinander angewandt heißen **Ableitung**

- Beispiel

- Klammerung ::= "(" Liste ")"  
Liste ::= Klammerung Liste  
Liste ::=

Klammerung  
 $\Rightarrow$ (Liste)  
 $\Rightarrow$ (Klammerung Liste)  
 $\Rightarrow$ (Klammerung Klammerung Liste)  
 $\Rightarrow$ (Klammerung (Liste) Liste)  
 $\Rightarrow$ ((Liste) (Liste) Liste)  
 $\Rightarrow$ ((() (Liste) Liste)  
 $\Rightarrow$ ((() () Liste)  
 $\Rightarrow$ ((() ())

# Beispiele

- Boolesche Term-Bäume
  - $T = \{ \text{true}, \text{false}, (, ), \vee, \wedge, \neg, \text{Variable} \}$
  - $N = \text{BOOL}$
  - $S = \text{BOOL}$
  - $P = \{ \text{BOOL} ::= \text{Variable}$ 
    - $\text{BOOL} ::= \text{true}$
    - $\text{BOOL} ::= \text{false}$
    - $\text{BOOL} ::= (\text{BOOL} \wedge \text{BOOL})$
    - $\text{BOOL} ::= (\text{BOOL} \vee \text{BOOL})$
    - $\text{BOOL} ::= \neg \text{BOOL} \ }$

# Erweiterte BNF

- EBNF - Erweiterung der Backus Naur Form
  - Erweiterung zur besseren Lesbarkeit
- Notation
  - Alternative |
  - Gruppierung ()
  - Optionaler Inhalt []
  - Beliebige Wiederholung {}
- BNF ohne Erweiterung
  - Zahl ::= PositiveZahl | "-" Positive Zahl  
PositiveZahl ::= (Ziffer PositiveZahl) | Ziffer
- BNF mit Erweiterung
  - Zahl ::= ["-"] { Ziffer }

# Erweiterte BNF

- Wiederholung in original BNF
  - `<Ziffer> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0`
  - `<Ziffenfolge> ::= <Ziffer> | <Ziffer> <Ziffenfolge>`
- Wiederholung in Erweiterter BNF
  - `<Ziffenfolge> ::= <Ziffer> { <Ziffer> }`
- Alternative Darstellung
  - Keine spitzen Klammern
  - "`=`" statt "`::=`"
  - Wiederholungen durch `+`
  - Optionale Wiederholungen durch `*`
  - Optionen durch `?`
  - Ziffer  $= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 .$
  - Ziffenfolge  $= \text{Ziffer}^+ .$

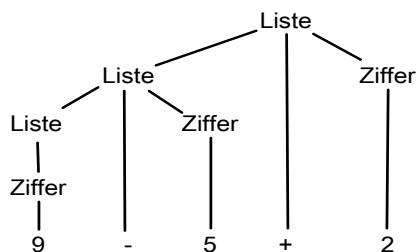
# Beispiele

- HTML-Tabellen
  - Table ::= <table> Rows </table>
  - Rows ::= { Row }
  - Row ::= <tr> Cells </tr>
  - Cells ::= { Cell }
  - Cell ::= <td> Text </td>
  - Cell ::= <td> Table </td>
- Satz der Tabellen-Grammatik
  - <table>
  - <tr>
  - <td>Mo</td>
  - <td>11-13</td>
  - <td>AM</td>
  - </tr>
  - <tr>
  - <td>Fr</td>
  - <td>11-13</td>
  - <td>AM</td>
  - </tr>
  - </table>

# Syntaxanalyse und Syntaxbäume

- Parsing: Zeichenkette → Struktur
- Beispiel-Produktionen
  - $\text{Liste} \rightarrow \text{Liste} + \text{Ziffer}$
  - $\text{Liste} \rightarrow \text{Liste} - \text{Ziffer}$
  - $\text{Liste} \rightarrow \text{Ziffer}$
  - $\text{Ziffer} \rightarrow 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9$

Parsebaum  
(konkreter Syntaxbaum)



Syntaxbaum  
(abstrakter Syntaxbaum)

