Universität Stuttgart
Fakultät Informatik

Prüfer:	Prof. Dr.-Ing. U. G. Baitinger
Betreuer:	Dr. rer. nat. Michael Ryba

Beginn am:	1. Mai 1996
Beendet am:	4. November1996

CR-Klassifikation	D.1.5, D.2.2, D.2.7, I.3.5, J.6

Diplomarbeit Nr. 1409

Entwicklung objektorientierter Mechanismen zur Visualisierung numerischer Wertemengen

Sven Van Steenkiste

Institut für Parallele und Verteilte Höchstleistungsrechner

Breitwiesenstraße 20-22

D-70565 Stuttgart

Inhaltsverzeichnis

Abbildungsverzeichnis

Formelverzeichnis

Beispielverzeichnis

Kurzbeschreibung

Die vorliegende Arbeit beschreibt die Konzeption und Implementierung von objektorientierten Mechanismen zur Visualisierung und statistischen Bewertung von n-dimensionalen Wertemengen. Das Konzept setzt sich aus drei entkoppelbaren Bestandteilen zusammen.

Ein Teil der Arbeit befaßt sich mit der Repräsentation von n-dimensionalen Wertemengen und stellt Methoden zu deren Speicherung und Bearbeitung bereit. Die Repräsentation berücksichtigt Mehrfacheinträge gleicher Wertepaare und verwaltet diese. Die Arbeit zeigt, daß aufgrund der Gewichtung von Wertepaaren keine optimale Repräsentation existiert. Optimal bezieht sich in diesem Zusammenhang auf den minimalen Zeitbedarf für Modifikationen der Menge oder iterative Auswertungen über einen Intervallbereich der Wertepaare. Daher werden zwei alternative Darstellungen angeboten. Sie beseitigen jeweils die Stärken und Schwächen des korrespondierenden Pendants.

Ein weiterer Teil der Arbeit befaßt sich mit den mathematischen Verfahren, dabei handelt es sich um Methoden zur statistischen Auswertung von Wertemengen. Die Mathematik bietet für die Interpretation über n-dimensionale Wertemengen, bei denen die relative und/oder absolute Häufigkeit von Wertepaaren zu berücksichtigen sind, die Methoden der Regressions- und Korrelationsanalyse. Die Regressionsanalyse untersucht einen funktionalen Zusammenhang und ermöglicht so, das Verhalten der Wertemenge auch in solchen Bereichen zu approximieren zu denen nur wenige Wertepaare existieren. Die Auswahl geeigneter Funktionen wird durch ein Maß über die Güte, mit der die Funktion das Intervall der Wertemenge repräsentiert, erleichtert. Dieses Maß wird durch die Korrelationsanalyse bereitgestellt. In der Ausarbeitung wird auf Probleme eingegangen, die bei der Verwendung von mehrdimensionaler Regression entstehen, und warum bei Abschätzung unter Verwendung von nichtlinearen Funktionen Transformationsschritte erforderlich sind.

Den Abschluß der Arbeit bildet die Visualisierung der darstellbaren Wertemenge unter Verwendung des Beurteilungsverfahren. Große Bedeutung erhalten bei der Visualisierung die jeweilige Interpretation der Daten. Zum Beispiel werden bei Streudiagrammen die Wertepaare disjunkten Intervallen zugeordnet und über statistische Funktionen ausgewertet. Die Ergebnisse werden anschließend in die Darstellung in Form von Streubalken überführt. Die Wertemengen sind in diesem Fall naturgemäß auf maximal drei Dimensionen beschränkt. Zur Reduzierung der Dimensionen werden einfache Projektionen bereitgestellt.

Bei diesem Vorgang können Mehrdeutigkeiten entstehen, die es zu berücksichtigen gilt. Die Visualisierung ist so konzipiert, daß eine Erweiterung um weitere Darstellungsformen, wie zum Beispiel B-Spline oder einfache Kuchendiagramme, ermöglicht wird.

1 Einleitung

Bei der Planung und Durchführung größerer Projekte, wie beispielsweise der Entwicklung einer integrierten Schaltung, fallen durch Beobachtung des Projektverlaufes verstärkt Wertemengen an, die es auszuwerten gilt. Da diese Wertemengen sich einer intuitiven Interpretation entziehen, ist es häufig sinnvoll, diese Wertemengen mit geeigneten mathematischen (statistischen) Verfahren zu analysieren und zu visualisieren.

Diese Verfahren zur Analyse bzw. Visualisierung sind nicht neu, stehen aber nur in Verbindung mit speziellen Softwaresystemen zur Verfügung. Um einen Einsatz im Bereich der CAD - Softwareentwicklung zu ermöglichen, war es die Aufgabe der Diplomarbeit eine Bibliothek mit entsprechenden Klassen zur Visualisierung und Analyse allgemeiner Wertemengen zu entwickeln.

1.1 Durchführung der Arbeit

Ziel der Arbeit ist es, geeignete Strukturen zu finden, mit denen sich die Wertemengen verwalten und visualisieren lassen. Der Aufgabe zugrundegelegt werden Betrachtungen für existierende Visualisierungen über Wertemengen. Dabei werden bestehende Gemeinsamkeiten und Unterschiede herauskristallisiert. Es werden verschiedene Arten von Interpretationen über Wertemengen und deren unterschiedliche Visualisierungsmöglichkeiten analysiert. Diese Analyse liefert die Anfangsbasis um zwischen der Verallgemeinerung in der allgemeinen Bibliothek und der Spezialisierung, zum Beispiel im Fall der Regressionsanalyse, geeignet zu trennen. Diese Trennung ist erforderlich, um dem Gesichtspunkt der Wiederverwertung von Komponenten gerecht zu werden.

Zur Sicherstellung der Wiederverwertung von Teilen der Arbeit, sind die Klassen sehr feingranular aufgespalten und der erste Ansatz einer Klasse ist soweit wie nur möglich verallgemeinert. Zusätzlich ist natürlich eine strikte Trennung zwischen verallgemeinerten Ansatz und der Spezialisierung erforderlich. Auf dem allgemeinen Ansatz der Visualisierung von Wertemengen setzt die Spezialisierung der Visualisierung auf Basis der Regressionsanalyse auf.

Voraussetzung für die Auswertung durch die Regressionsanalyse ist ein funktionaler Zusammenhang zwischen Attributen des Definitionsbereiches und dem Attribut, das als Wertebereich ausgewählt worden ist.

Die Abbildungen der Wertemengen wurden so entworfen, daß wiederholtes Aufteten gleicher Tupel berücksichtigt wird. Bei dem allgemeinen Ansatz der Visualisierung wird mit geschachtelten Verwaltungseinheiten gearbeitet.

1.2 Aufbau der Arbeit

Kapitel 2 beschreibt die grundlegenden Möglichkeiten der Visualisierung von Wertemengen. Hier werden die Gemeinsamkeiten und Unterschiede herausgearbeitet, die bei der Trennung in allgemeinen Ansatz und Spezialisierung anschließend eine Rolle spielen.

In Kapitel 3 werden Notationen für OOA/OOD vorgestellt und die in der Ausarbeitung verwendete Notation genauer beschrieben.

Kapitel 4 stellt die verwendete Entwicklungsumgebung und die Programmiersprache EIFFEL kurz vor.

Kapitel 5 beschreibt die mathematischen Grundlagen für die Auswertung der zu analysierenden Wertemengen. Dabei wird insbesondere auf die Korrelations- und Regressionsanalyse eingegangen.

Kapitel 6 beschreibt den Grobentwurf der zu implementierenden Bibliotheken.

Kapitel 7 beschäftigt sich mit der Verwaltung von Wertemengen, wobei insbesondere der Repräsentation von n-dimensionalen Wertemengen Bedeutung zukommt. Die Ausarbeitung beschreibt die resultierenden zwei Repräsentationen, optimiert auf Modifikation bzw. Auswertung von Wertemengen.

Kapitel 8 beschreibt die Klassenbibliothek für die eigentliche Analyse und Bewertung der Wertemengen basierend auf der Regression.

Kapitel 9 beschäftigt sich mit allgemeinen Mechanismen zur Visualisierung von Diagrammen. Ausgehend vom bekannten Modell-View-Controller Ansatz, wird dargestellt wie sich die Wertemengen über geeignete Interpretationsvorschriften in visuelle Repräsentationen überführen lassen.

Kapitel 10 gibt die Vorgehensweise an wie aus dem allgemeinen Modell konkrete Diagramme erzeugt werden können.

Kapitel 11 faßt schließlich die Ziele und erreichten Ergebnisse zusammen und gibt einen Ausblick auf weiterführende Arbeiten.

2
Möglichkeiten der Visualisierung

In diesem Kapitel sollen unterschiedliche Möglichkeiten der Visualisierung von Wertemengen beschrieben werden, um die anschließende Gliederung der gemeinsamen Struktur nachvollziehbar zu gestalten. Es gilt die Unterschiede im Aufbau und der Funktionsweise von Diagrammen aufzuzeigen und herauszukristallisieren. Die wesentliche Fragestellung hierbei ist, wie sich gleiche Interpretationsansätze in unterschiedlicher Art und Weise strukturieren und gliedern lassen. Für die Beschreibung wird zunächst eine getrennte Betrachtung von diskreten und kontinuierlichen Wertemengen durchgeführt. Hierbei wird unter kontinuierlich verstanden, daß die Wertepaaren über Funktionen ermittelt werden und so eine inkrementelle Berechnung von Wertepaare durchgeführt werden kann. Die Trennung in der Betrachtungsweise nach kontinuierlich und diskret erfolgte aufgrund der Aufgabenstellung, deren Schwerpunkt auf der Visualisierung diskreter Wertemengen liegt.

Der in der Ausarbeitung verwendete Begriff der Dimension weicht in Teilen von dem direkten Eindruck aufgrund der Visualisierung ab. In Visualisierungen wird häufig eine Dimension in Form der Anzahl oder des Index des zugrundegelegten Tupels einer Wertemenge gebildet. Zum Beispiel werden, bei der Generierung eines Diagramms aus einer Tabellendarstellung heraus, in einigen Visualisierungsarten Spalten- und Zeilenindizes als zusätzliche Dimension verwendet.

Der hier zugrundegelegte Dimensionsbegriff ist verankert in der Beschreibung eines einzelnen Tupels, das heißt in der Frage: Welche Angaben sind erforderlich um das Tupel geeignet zu beschreiben? Werden Tupel in zeitlicher Reihenfolge betrachtet, so ist ein Attribut zur Speicherung des Zeitwertes fester Bestandteil des Tupels. Das heißt, die Wertemenge legt den Dimensionsgrad fest, nicht deren Visualisierung.

2.1 Darstellung diskreter Wertemengen

Bei diskreten Wertemengen ist Anzahl der Dimensionen über die Anzahl der Attribute enthaltenen Tupel definiert. Tupel gleichen Wertes dürfen in den Wertemengen auch mehrfach auftreten. Soweit die Interpretationsvorschrift es nicht anders festlegt, sind die Tupel einzeln zu betrachten und als voneinander unabhängig anzusehen.

Bis auf den Fall von 1-dimensionalen Wertemengen wird ein Attribut eines Tupels als Teil der Zielmenge betrachtet und alle anderen Attribute als Bestandteile der Definitionsmengen. Daher wird im Fall von diskreten Wertemengen auch zwischen einem relationalem oder funktionalem Zusammenhang zwischen Definitions- und Wertemengen unterschieden. Diese explizite Unterscheidung fällt bei der Betrachtung von kontinuierlichen Wertemengen weg. In diesem Fall ist die Einteilung implizit in der Berechnungsvorschrift festgelegt.

Für den Dimensionsbegriff über diskrete Wertemengen noch ein paar Beispiele: Ein Segment eines Kreisdiagrammes repräsentiert ein Tupel durch die Angabe eines Attributes, es wird somit eine 1-dimensionale Wertemenge zugrundegelegt. Ein Punkt in einem 2-dimensionalen kontinuierlichen Koordinatenfeld muß Bestandteil einer 2-dimensionalen Wertemenge sein. Vereinfacht ist der Dimensionsbegriff an die Fähigkeit der Darstellungsprimitive gekoppelt, das heißt ob dieses ein Tupel mit n Attributen repräsentieren kann oder nicht. Beispielsweise kann ein Netzdiagramm mit sechs Achsen, ein Tupel mit sechs Attributen repräsentieren.

Eine weitere Trennung erfolgt abhängig von der Anzahl untersuchter Tupel einer Wertemenge. Zuerst werden die Wertemengen, bei denen sich die Darstellung der einzelnen Tupel aufgrund einer kleineren Anzahl aussagekräftig visualisieren lassen, betrachtet. Dem gegenübergestellt sind Wertemengen, bei denen die Anzahl von Tupeln so groß ist, daß eine vorgeschaltete Interpretationsvorschrift benötigt wird, um die Wertemenge samt Tupel in ihrem Aussagewert auf ein beurteilbares Maß zu reduzieren.

In der Visualisierung werden dann nur noch die Ergebnisse der Berechnung der Interpretationsvorschrift visualisiert. Zum Beispiel besitzt die „reine“ Punktewolke einer Wertemenge mit über 1000 Tupeln nur eine geringen Aussagewert, erst durch geeignete Vorschriften der Interpretation wird eine sinnvolle Aussage daraus gewonnen. Hierbei kann es sein, daß die endgültigen Ergebnisse unterschiedlicher Interpretationsvorschriften in der gleichen visuellen Repräsentation resultieren.

2.1.1 Darstellung 1-dimensionaler Wertemengen

Unter den Begriff der 1-dimensionalen Wertemenge fallen alle als Geschäftsgraphiken bezeichneten Diagrammdarstellungen. Dieser Bereich wird wegen der Untersuchung der Gemeinsamkeiten von Diagrammen hinzugezogen. Eine Bearbeitung mit Methoden der Regressionsanalyse entfällt, da keine Trennung zwischen Definitions- und Wertemengen erfolgen kann.

Unter 1-dimensionalen Wertemenge werden alle Wertemengen eingeordnet, die frei wählbare Werte in einer Dimension annehmen können. Darunter werden auch Diagramme verstanden, die Einzelwerte unterschiedlicher Quartale miteinander vergleichen. In diesem Fall wird entweder eine Ansammlung von unterschiedlichen 1-dimen-

sionalen Wertemengen zugrundegelegt oder aber eine implizite Erhöhung der Dimensionsanzahl der Wertemenge durch die Berücksichtigung des Index der Tupel durchgeführt. Die Ausgangsmenge bleibt aber 1-dimensional, da die Vertauschung der Indizes von Tupeln an der Menge nichts ändert. Dem Index von Tupeln kommt in Wertemengen keine Bedeutung zu.

Kuchendiagramme

Ein verbreitetes Diagramm zur Visualisierung 1-dimensionaler Wertemengen ist das Kuchendiagramm. In einem Kuchendiagramme werden ein oder mehrere Wertemengen betrachtet. Bei der Interpretation wird die absolute und die relative Wertbetrachtung unterschieden. Im Unterschied zu anderen Diagrammen bekommen in diesem Diagramm einzelne Tupelausprägungen und nicht die Wertemenge an sich unterschiedliche Farbwerte zugeordnet.

Abbildung 2-1: Kreisdiagramme

Bei der Betrachtung mehrerer Wertemengen wird unterschieden, ob diese gemeinsamen Ursprungs sind, daß heißt die gleiche Legende verwenden können oder nicht. Eine besondere Art ist die Darstellung von zwei Kuchendiagrammen, wobei ein einzelnes Segment des einen Diagrammes im Zweiten verfeinert betrachtet wird. Eine Ansammlung von Wertemengen läßt sich in Form von ineinander geschachtelter Kreisringe darstellen.

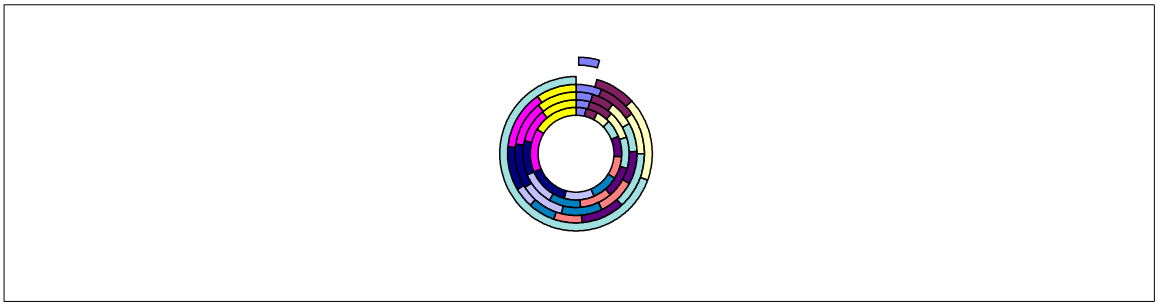

Abbildung 2-2: Diagramm aus Kreisringen

Balkendiagramme

Eine andere Art der Betrachtung von 1-dimensionalen Wertemengen ist die Darstellung als Balkendiagramm; bei der Visualisierung wird zwischen vertikaler und horizontaler Darstellungsart unterschieden.

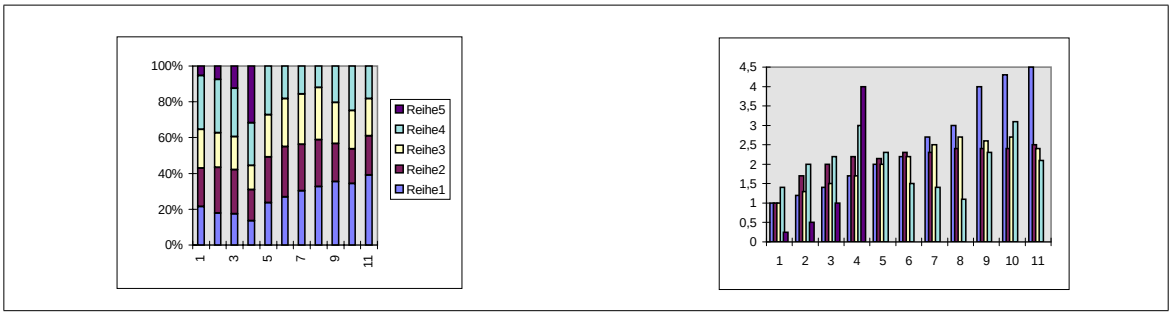

Abbildung 2-3: Balkendiagramme

Die Betrachtungsweisen von relativen und absoluten Werten sind genauso gebräuchlich wie die Betrachtung von summierten Einzelwerten als Ganzes. In diesem Fall wird eine Ansammlung aus Wertemengen betrachtet.

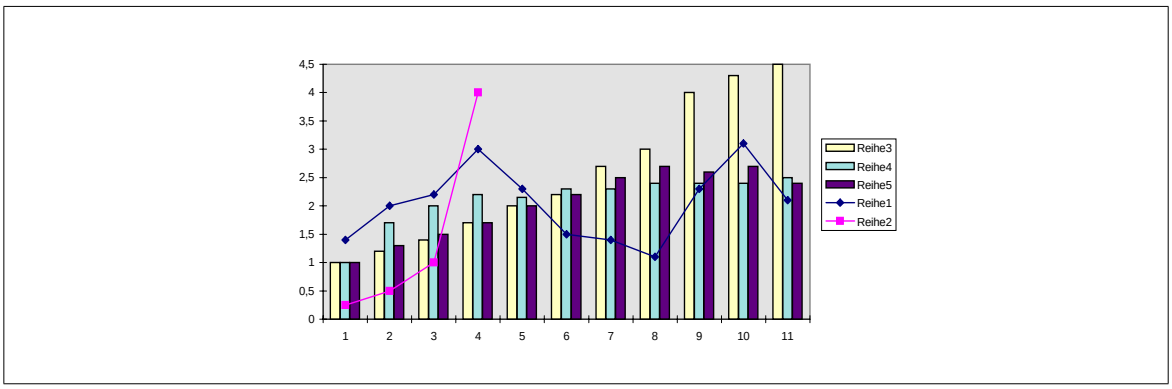

Abbildung 2-4: Gemischte Darstellung im Balkendiagramm

Als Beschriftung der Achsen dienen in diesem Fall oft festgelegte Zeiträume, wie zum Beispiel Quartale oder Jahre. Die Darstellungen von Balkensdiagrammen werden natürlich wahlweise auch in vertikaler Ausrichtung dargestellt.

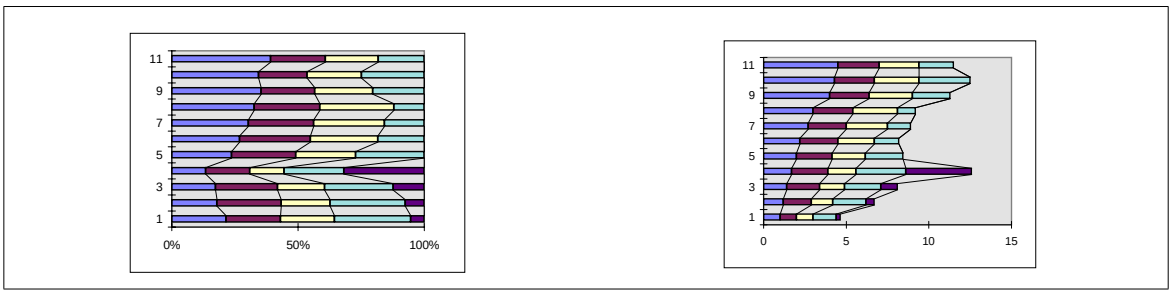

Abbildung 2-5: Vertikale Balkendiagramme

Punktediagramme

Im Punktediagramm von Wertemengen werden die Tupel „direkt“ in einem Koordinatenpunkt der Darstellung visualisiert. Die Betrachtungsweise ähnelt derjenigen des Balkendiagrammes, zusätzlich werden aber noch Symbole der Legenden als Marker verwendet und die Punkte über geeignete Interpolationen oder Approximation verbunden.

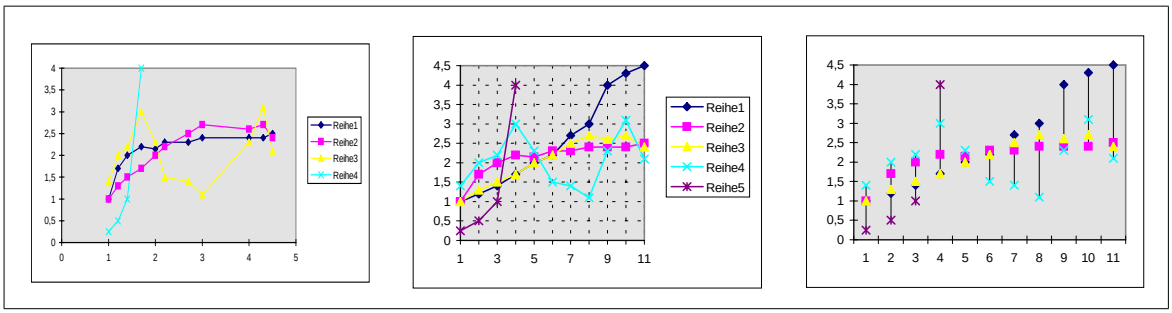

Abbildung 2-6: Punktediagramme

Die Darstellung läßt sich auf 3-Dimensionen erweitern, aber jede Wertemenge für sich ist eindimensional, da die zweite Dimension durch den Index der Tupel erzeugt wird.

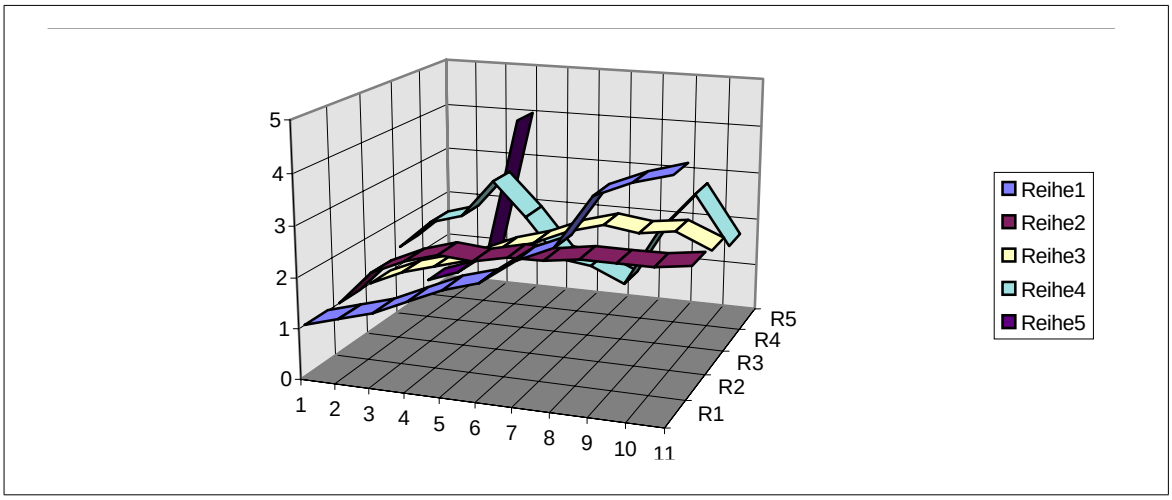

Abbildung 2-7: Liniendiagramm mit 3-dimensionalem Effekt

Intervallbetrachtung

Die Intervallbetrachtung generiert über eine zu definierende Intervallvorschrift Werte, die eine Aussage über das Verhalten in dem Intervall liefern. Zum Beispiel die Vorschrift: Suche für jedes Intervall der Wertmenge die Attributintervalle in denen alle Tupel enthalten sind. Es entsteht somit eine „reduzierte Wertemenge“ bestehend aus dem Intervall von Attributwerten aller Tupel, die anschließend geeignet zu visualisieren ist.

Dies kann in Form der bereits vorgestellten Diagrammarten geschehen, naheliegender sind aber Betrachtungen, die dem Intervall mehr Informationen über den Charakter der Wertemenge bereitstellen. Hierfür werden Streudiagramme oder Netzdiagramme verwendet.

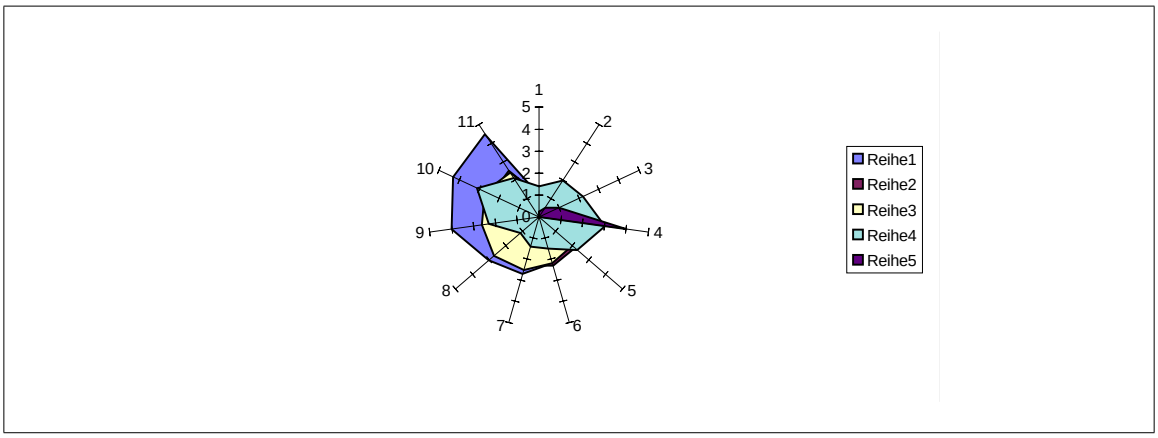

Abbildung 2-8: Netzdiagramm über reduzierte Wertemenge

Streudiagramme

Streudiagramme sind Diagramme, die aus Wertemengen über die Definition von Intervallen und darüber operierenden Interpretationsvorschriften „reduzierte“ Wertemengen berechnen, die es anschließend zu visualisieren gilt. Bekannte Streudiagramme sind solche, die Streubalken erzeugen, die den minimale, den maximale und den Durchschnittswert angeben.

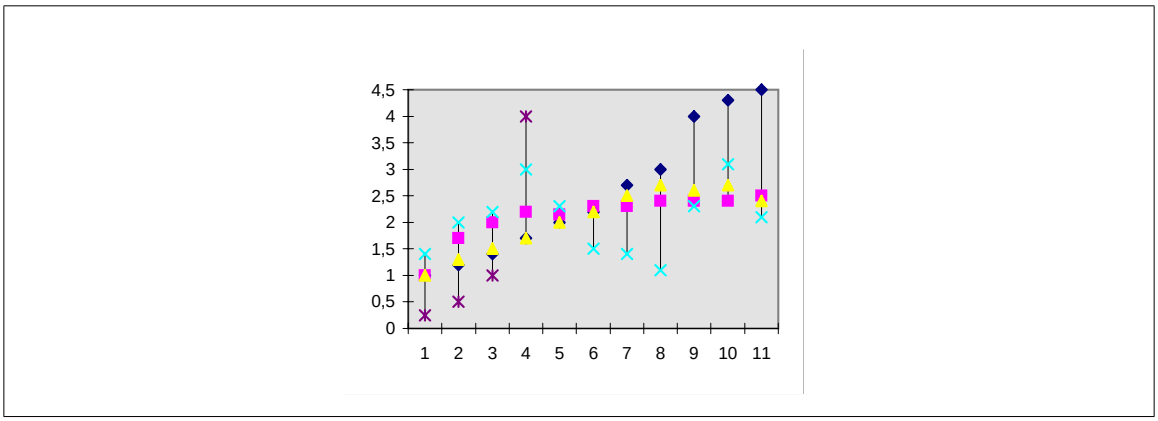

Abbildung 2-9: Streudiagramm

Die Vorschrift der zugrundegelegten Interpretation ist ein Modellansatz für diese Art von Darstellung. Genauso lassen sich andere Darstellungsformen basierend auf statistischen Auswertungsmethoden berechnen und anschließend visualisieren.

Eine andere Möglichkeit der Visualisierung ist das Bilden von Bereichen, zu denen Werte existierten.

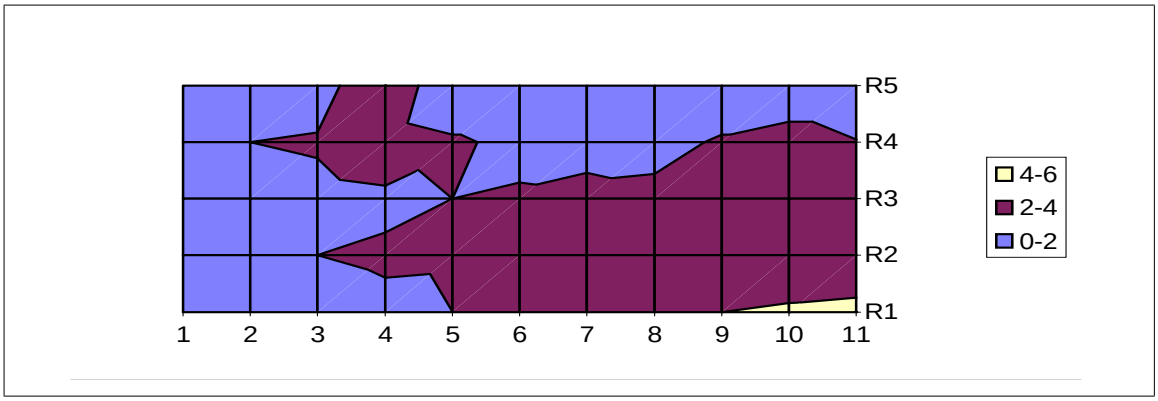

Abbildung 2-10: Bereichsdiagramm

Flächendiagramme

Aus Gründen der Vollständigkeit wird noch ein Beispiel für die Darstellung in Form von Flächendiagrammen gezeigt.

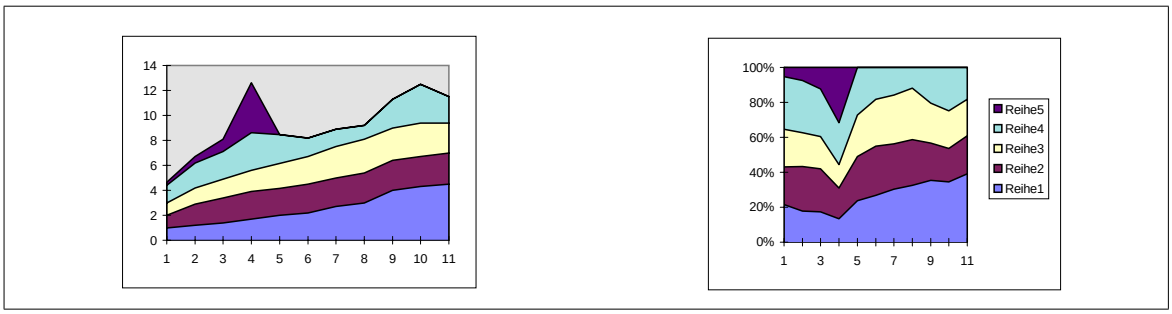

Abbildung 2-11: Flächendiagramme

Darstellungen mit 3-D Effekt

Eine weitere Möglichkeit bei der Visualisierung ist die Verwendung von 3-D Effekten.

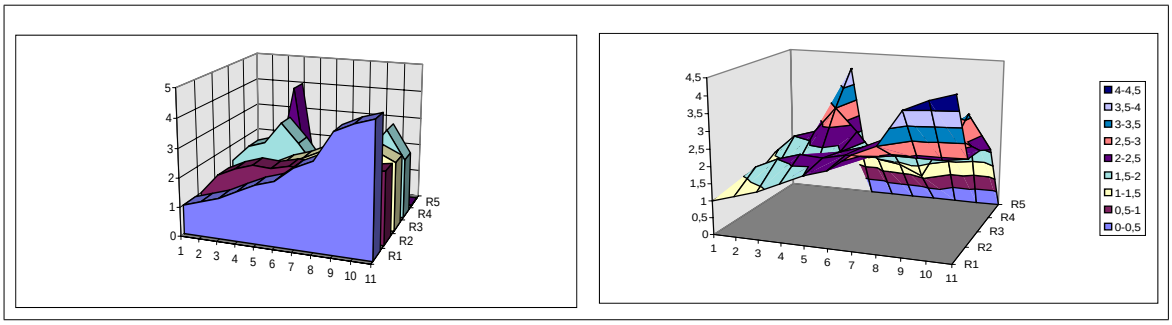

Abbildung 2-12: Diagramme mit 3-dimensionalem Effekt

Die Darstellung wirkt in dieser Form plastischer und anspruchsvoller. Hierbei wird in den meisten Darstellungsarten zur Erreichung des 3-dimensionalen Effektes nur ein fester Wert für die Tiefe gesetzt und dargestellt.

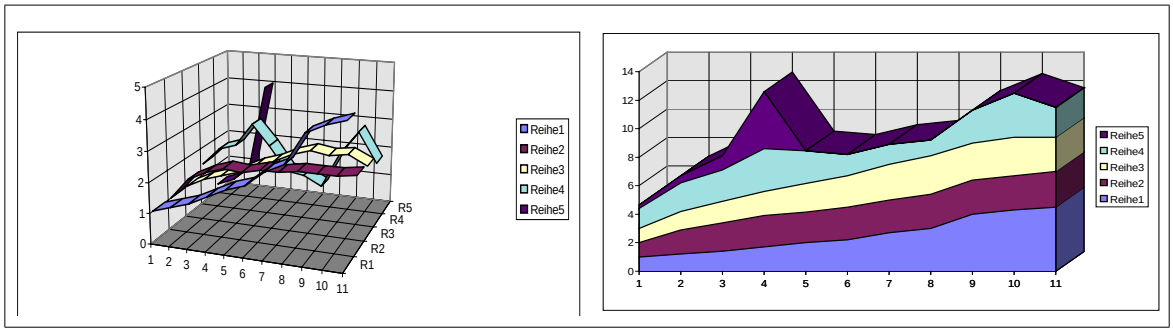

Abbildung 2-13: Weitere Diagramme mit 3-dimensionalen Effekten

2.1.2 Darstellung 2-dimensionaler Wertemengen

Bei 2-dimensionalen Wertemengen, vom Dimensionsgrad nicht zu verwechseln mit der Darstellung einer Ansammlung von 1-dimensionalen Wertemengen, werden die Werte in beiden Dimensionen frei gewählt.

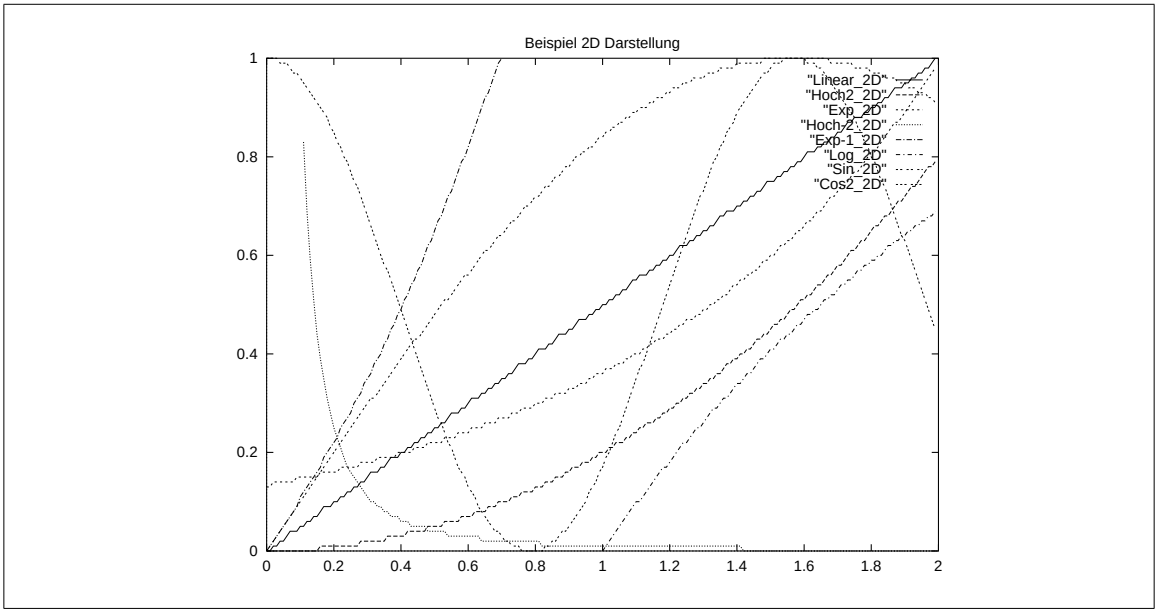

Abbildung 2-14: Liniendiagramm mit 2-Dimensionen

In diesem Fall reduzieren sich die Möglichkeiten auf die Verwendung von Koordinatensystemen mit zwei Dimensionen, in denen beliebige Punktewolken eingetragen sein können. Die wichtigsten Vertreter sind Polar- und Kartesischekoordinaten.

Punktewolke

Die Punktewolke als Repräsentant der 2-dimensionalen Wertemenge ist nur aussagekräftig, wenn die Anzahl der beteiligten Tupel oder die Anzahl der unterschiedlichen Tupel, die aus ein und demselben Darstellungspunkt resultieren, nicht zu groß wird. Ansonsten wird der Aussagewert dieser Darstellung zu gering.

Wenn nur eine Wertemenge in der Darstellung eingetragen ist, dann kann ein Mehrfachauftreten von gleichen Tupeln farbig unterschiedlich visualisiert werden. Sind mehrere Wertemengen in die Darstellung eingebunden, wird die Analyse durch überdeckende Punkte verfälscht. Um den Aussagewert zu erhöhen und um auch eine Visualisierung einer großen Anzahl von Punkten zu ermöglichen, sind Vorschriften für die Interpretation auszuführen. So wird eine „reduzierte“ Wertemenge mit Attributen der interpretierten Bedeutung erhalten.

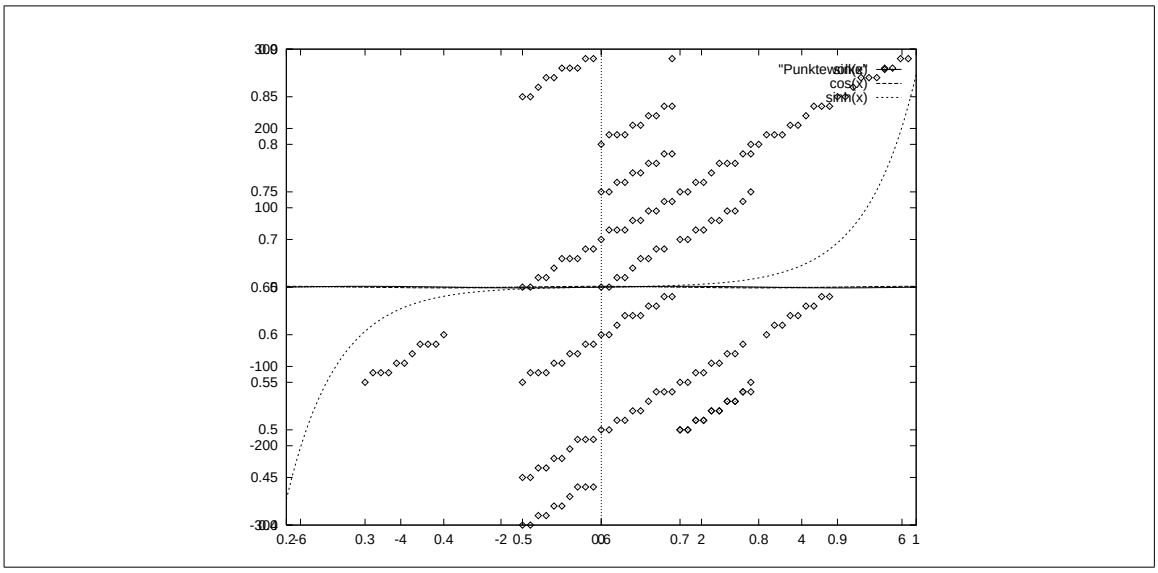

Abbildung 2-15: Punktewolke für 2-dimensionale Werte

Diese Vorschriften führen zu erweiterten Formen von Streudiagrammen. Eine vereinfachte Darstellung ist die Bildung von dynamischen Intervallbereichen, für die jeweils statistische Aussagen getroffen werden. Diese Aussagen gilt es anschließend zu visualisieren.

Abbildung auf funktionales Verhalten

Da diese Wertemengen mehr als eine Dimension aufweisen, läßt sich eine Dimension als Definitionsmenge und die andere als Zielmenge betrachten. Der im allgemeinen relationale Charakter der Abbildung zwischen Definitions- und Wertebereich wird über

eine Schätzfunktion approximiert. Diese Funktion bietet die Möglichkeit, das Verhalten der Wertemenge abzuschätzen und dadurch eine genauere und bessere Interpretation der Wertemenge zu erhalten.

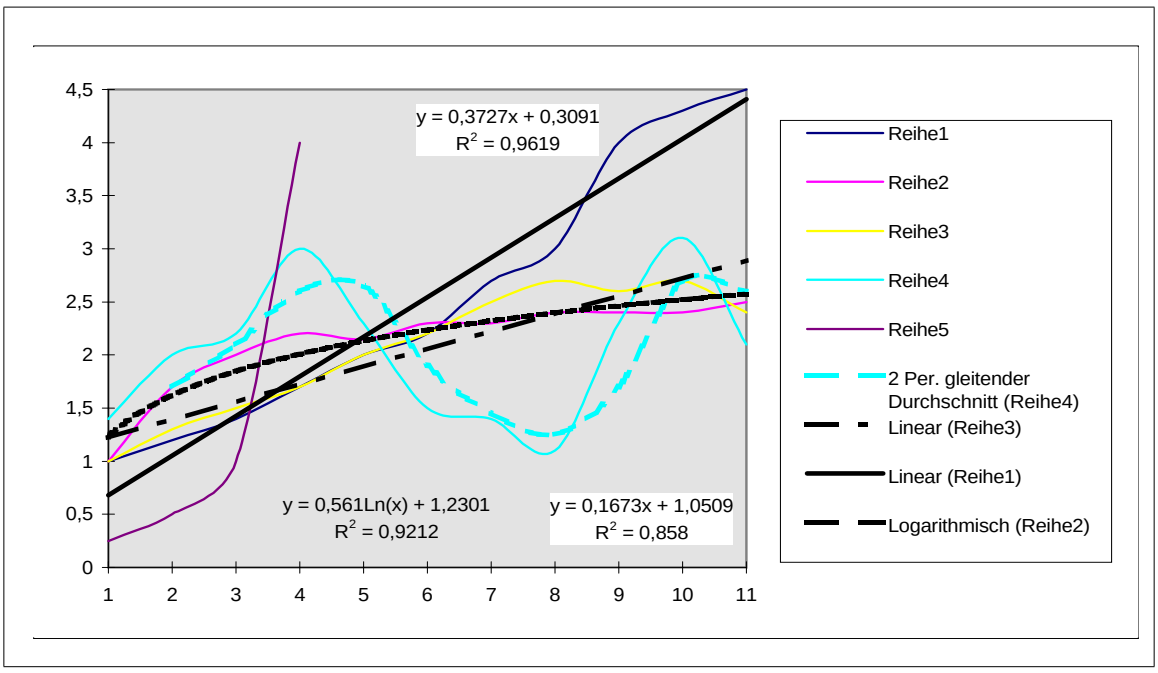

Abbildung 2-16: Regressionsdiagramm (2-dimensional)

Ein weiterer Vorteil ist, daß durch die Reduzierung der Wertemenge auf eine Ansammlung von Funktionsabschätzungen, der Vergleich zwischen Wertemengen überhaupt erst möglich gemacht wird.

2.1.3 Darstellung von 3-dimensionalen Wertemengen

Eine Darstellung von 3-dimensionalen Wertemengen ist eine Erweiterung der Darstellung von 2-dimensionalen Wertemengen. Gebräuchliche Koordinatensysteme hierfür sind Polarkoordinaten für drei Dimensionen, Zylinderkoordinaten und kartesiche Koordinaten.

Alle Einschränkungen, die in Bezug aussagekräftiger Analysen zutreffend sind, gelten in verstärkter Form im Fall von drei Dimensionen. Bei der Darstellung von 3-dimensionalen, diskreten Wertemengen müssen zusätzlich Aspekte der Z-Ordnung je nach Beobachtungsrichtung berücksichtigt werden.

Regression von 3-dimensionalen Wertemengen

Bei der Regression von 3-dimensionalen Wertemengen wird eine funktionale Abhängigkeit von zwei Attributen auf das 3. Attribut durchgeführt. Die Wertemenge wird so auf Intervallbereiche in denen festgelegte Funktionen gelten reduziert. Diese Funktonen lassen eine inkrementelle Berechnung unter Berücksichtigung der Z-Ordnung zu.

Werden zusätzlich Vorschriften verwendet, die eine „reduzierte“ diskrete Wertemenge erzeugen, so müssen diese für jede Ansicht in der jeweiligen zu zeichnenden Z-Ordnung sortiert werden.

2.1.4 Darstellung von Wertemengen höherer Dimensionsgrades

Ein Darstellung von Wertemengen mit mehr als 3 Dimensionen ist nur durch die Betrachtung einer geringen Anzahl von Tupeln oder in Form von Bereichs- oder Intervallbetrachtung möglich.

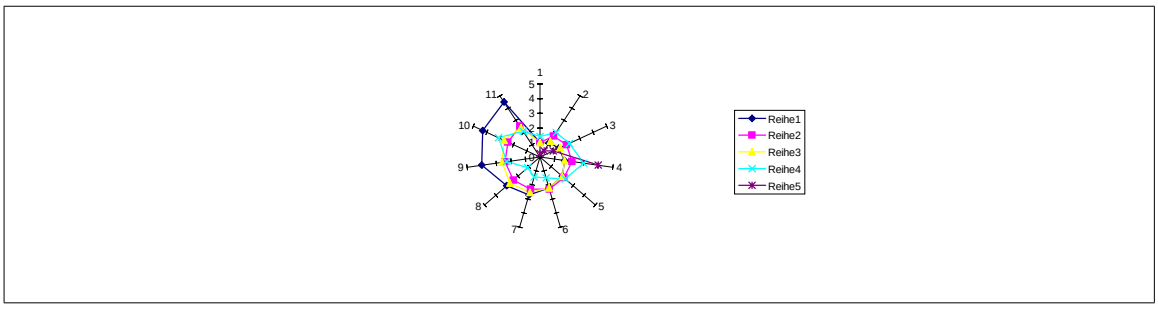

Abbildung 2-17: Netzdiagramm für einzelne Tupel

Diese Darstellung wird wegen ihres Aussehens üblicherweise als Netzdiagramm bezeichnet. Eine Intervallbetrachtung über eine geringe Anzahl von Tupeln ist auch in einem orthogonalen Koordinatensystem möglich.

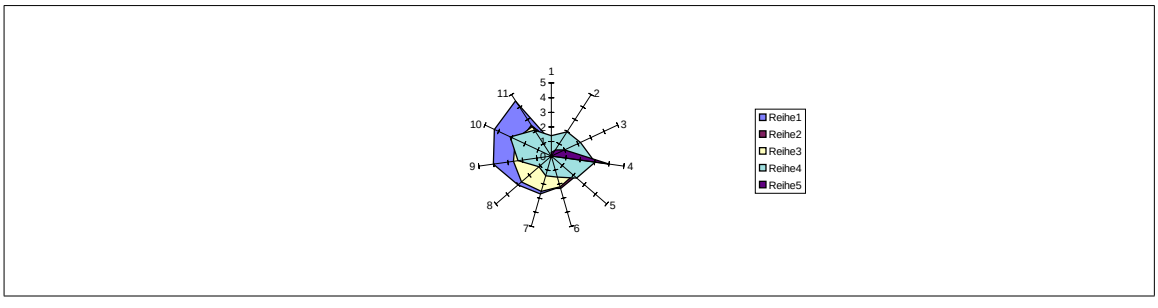

Abbildung 2-18: Netzdiagramm als Bereichsauswertung

2.2 Darstellung kontinuierlicher Wertemengen

Die Darstellung von kontinuierlichen Wertemengen wird hier nur kurz angesprochen. Unter kontinuierlichen Wertemengen werden diejenigen verstanden, die über Funktionen erzeugt werden. Dabei ist klar, daß der Dimensionsgrad der Funktion um eins kleiner ist als derjenige der resultierenden Wertemenge.

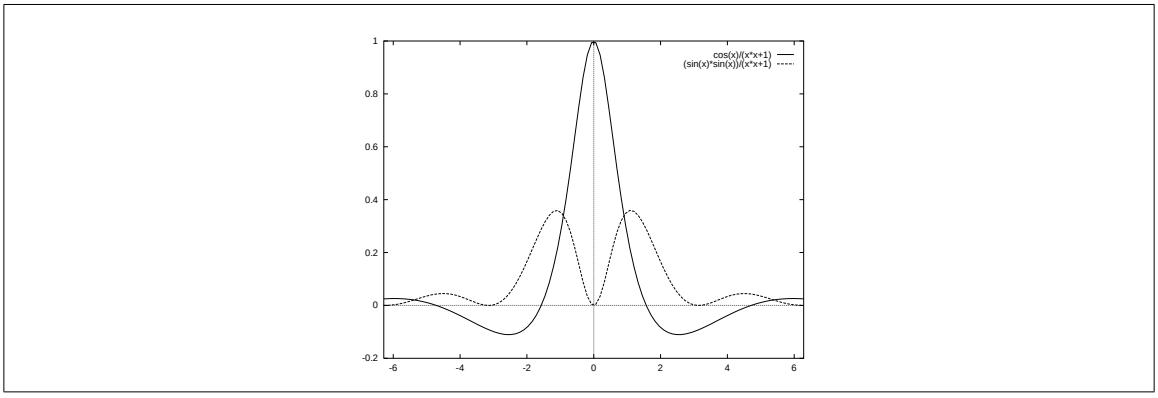

Abbildung 2-19: Funktion mit einer Variablen

Ein wesentlicher Vorteil von kontinuierlichen Werten besteht darin, daß die Z-Ordnung bei der Berechnungsreihenfolge berücksichtigt werden kann, wogegen bei diskreten Werten die gesamte resultierende Wertemenge der vorgeschalteten Interpretation entsprechend ihrer Z-Ordnung sortiert werden muß.

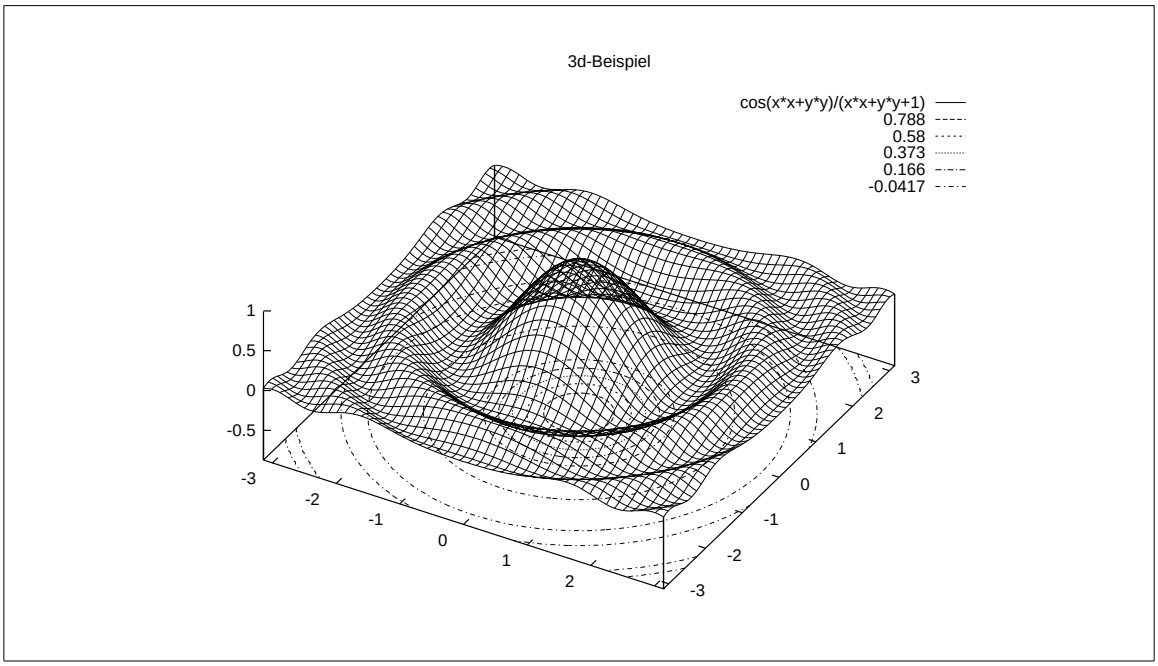

Abbildung 2-20: 3-dimensionale Funktion

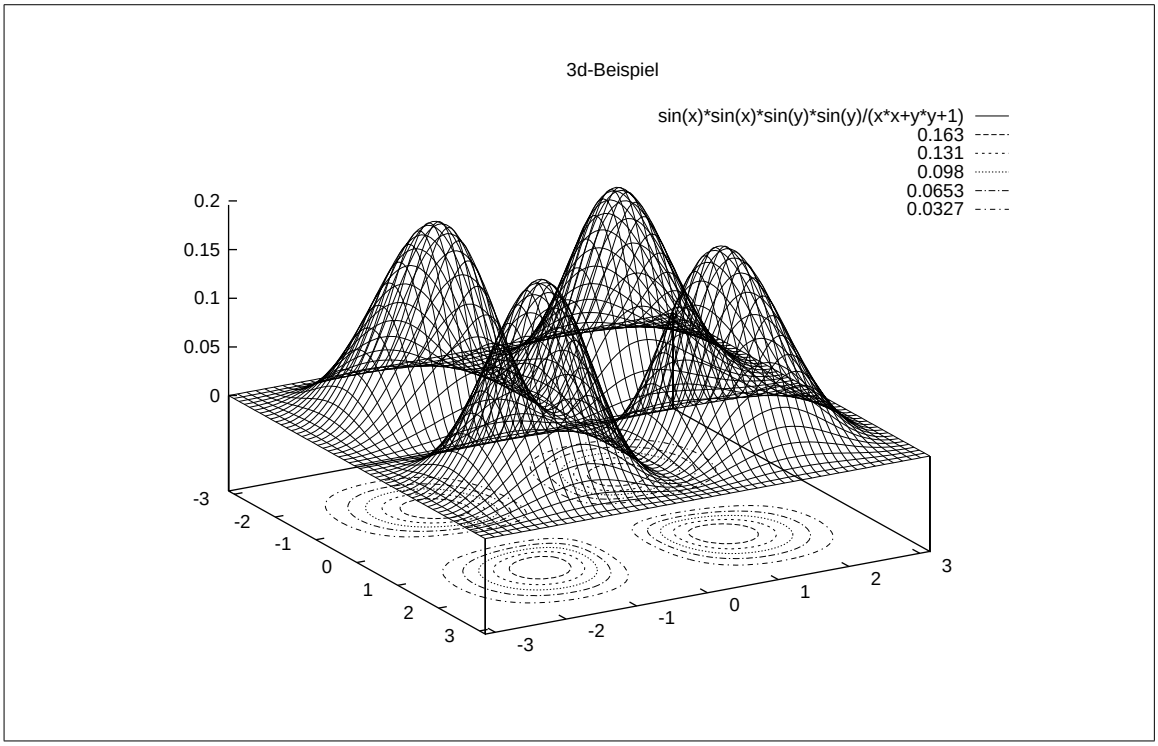

Abbildung 2-21: 3-dimensionale Funktion

2.3 Gemeinsamkeiten und Unterschiede

Aus dieser umfangreichen Sammlung von unterschiedlichen Darstellungen für Wertemenge lassen sich jetzt Gemeinsamkeiten und Unterschiede herausarbeiten. Diese dienen als Entscheidungsgrundlage zur Auftrennung in einen allgemeinen Ansatz und die aufsetzende Spezialisierung.

2.3.1 Gemeinsamkeiten

Allen Diagrammen ist gemeinsam, daß sie innerhalb eines Dokumentes einer rechtwinkligen Fläche belegen. Innerhalb dieser Bereiche werden die folgenden Darstellungsobjekte verwendet: Titel, Untertitel, Legenden, Begleittexte und interne Flächen, die abhängig von der Aussage des Diagrammes unterschiedliche Darstellungen von komplexen Objekten visualisieren. Diese internen Flächen werden im folgenden auch als Zeichenflächen bezeichnet.

Ob jetzt in der jeweiligen Zeichenfläche ein Kuchendiagramm oder eine Punktewolke visualisiert wird ist nur abhängig von der zugrundegelegten Interpretation. Zeichenflächen können auch aus Achsen, deren Beschriftung und dem eigentlichen Bereich der Darstellung bestehen.

Die Zeichenflächen wiederum können mehrere unterschiedliche Interpretationen einer Wertemenge darstellen. Eine Zeichenfläche muß in der Lage sein mehrere Interpretationsvorschriften über eine Ansammlung von Wertemengen auszuwerten.

Bei Darstellungen von 3-dimensionalen Wertemengen muß zu jedem unterschiedlichen Bezugspunkt der für eine Ansicht gewählt werden kann die Z-Ordnung berechnet werden. Diese legt fest in welcher Reihenfolge die bei der Interpretationsvorschrift ermittelten Objekte des Darstellungsraumes zu zeichnen sind.

2.3.2 Unterschiede

Die großen Unterschiede liegen in verschiedenen Kopplungen zwischen den einzelnen Komponenten. Werden zum Beispiel mehrere Kuchendiagramme in einem Diagramm dargestellt, stellt sich die Frage, ob die Legendengruppe gemeinsam genutzt wird oder nicht. Wird nur ein Darstellungsbereich in Form einer Zeichenfläche verwendet, stellt sich die Frage, ob unterschiedliche Darstellungsmodelle zugelassen werden und wenn ja welche ? Genauso lassen sich über Spezialisierung weitere Darstellungen aufbauen, bei denen die Interpretationsvorschrift fest vorgegeben wird und nur das Hinzufügen zusätzlicher Wertemengen erlaubt ist.

3
Rahmenbedingungen der Entwurfsumgebung

Unter Rahmenbedingen werden in diesem Kapitel solche Bedingungen verstanden, die Werkzeuge oder Notationen festlegen, welche der Ausarbeitung zugrunde gelegt und projektübergreifend eingesetzt werden. Diese Bedingungen umfassen die gewählte Entwurfsmethodik und die Festlegung der Umgebung, in der die Implementierung durchgeführt werden soll. Darunter werden die verwendete Sprache und die zu benützenden oder benutzbaren Werkzeuge verstanden (siehe auch Kapitel 4).

3.1 Entwurfsmethode für Analyse und Design

Bei der Entwurfsmethodik, die in der Analyse- und Designphase eingesetzt wird, war schon vor Beginn der Arbeit das objektorientierte Paradigma als Vorgabe festgeschrieben. Die Methode der objektorientierten Analyse und des objektorientierten Design, im folgenden mit OOA und OOD abgekürzt, erweist sich als leicht nachvollziehbar und verständlich. Sie bildet das Fundament für eine einheitliche Vorgehensweise innerhalb der Enwurfsphasen für Analyse und Design [Hru94].

Die Entwurfmethode ist themenbedingt naheliegend, aber nicht zwingend erforderlich. Ein weiteres Argument für die Wahl von OOA/OOD ist ihre bereits bestehende Verwendung beim verwendeten *„Application Framework“* [Grh96][Grh96]

Bei einem Entwurf nach OOA/OOD sind verschiedene Notationen verbreitet. Dabei halten viele objektorientierte Ansätze die folgende Vorgehensweise für wichtig:

- Auffinden von Objekten
- Beschreiben der statischen Zusammenhänge zwischen Objekten, oft als statisches Modell bezeichnet
- Festlegen der strukturellen Beziehungen der Objekte in Bezug auf Generalisierung und Spezialisierung
- Beschreibung der dynamische Zusammenhänge zwischen Objekten; diese basieren in den meisten Fällen auf dem Nachrichtenaustausch und werden als dynamisches Modell bezeichnet
- Festlegen der interner Aufbau der herauskristallisierten Objekte

Das Design präzisiert die Analyse durch sukzessive Verfeinerung und Erweiterung um folgenden Aspekte:

- Einbringen von Technologieaspekten
- Modularisierung in hierarchischen Verbände; diese werden auch als *Kategorien* oder *Cluster* bezeichnet.

Eine Vereinheitlichung der Notationen mit dem Ziel einer Standardisierung hat noch nicht stattgefunden [Hru94]. Die Beschreibung in den jeweiligen Notationen erfolgt textuell oder aber graphisch dominiert. Die Schwerpunkte sind hier bei den einzelnen Notationen unterschiedlich gewichtet. Manche Notationen unterstützen historisch bedingt sowohl einen textuell als auch einen graphisch dominierten Ansatz. Als Beispiel wäre die Notation nach Rumbaugh zu nennen.

Die Trennung zwischen den Projektphasen Analyse und Design verläuft in vielen Notationen fließend und wird nur durch den Grad der Granularität der Beschreibung bestimmt. Der in der Analysephase stetig verfeinerte Entwurf geht in das Design über, sobald die Beschreibung präzisiert wird und Aspekte der Implementierung berücksichtigt werden. Beim Entwurf von Systemen können Bestandteile noch in der Analysephase sein, wohingegen andere Bereiche bereits ein fundiertes Design aufweisen.

Die Beschreibungsanteile in den jeweiligen Notationen, welche das Design präzisieren oder sich mit dynamischen Aspekten der Analyse befassen, können erheblich voneinander abweichen oder sind in manchen Notationen gar nicht verfügbar [StBa94].

Viele Notationen werden durch spezielle Werkzeuge unterstützt. Diese verbinden die Designphase mit der nachfolgenden Implementierung. Dies geschieht über den als Kopplung dazwischengeschalteten Codegenerator. Durch diese Kopplung weisen etliche Notationen Vorlieben und codespezifische Erweiterungen in Bezug auf den Generator und dessen Implementierungsprache auf. Diese Vorlieben werden durch Sprach- oder Graphikanteile in den Notationen deutlich hervorgehoben.

3.2 Objektorientierte Analyse und Design

Drei Notationen werden anhand kurzer Beispiele vorgestellt. Die ausgewählte Notation wird in Kapitel 3.3 genauer beschrieben. Ausführliche Vergleiche zwischen unterschiedlichen Notationen können der einschlägigen Literatur entnommen werden.

Unter den verbreiteten Vertretern von objektorientierten Notationen werden skizziert:

- Rumbaugh
- Booch [Boo94]

- Business Object Notation

Die Notationen nach Booch und Rumbaugh sind zumindest in Bezug auf die Phase der Analyse äquivalent in den Bereichnen Basis- und Strukturkomponenten [StBa94].

Die beispielhafte Aufzählung von drei Vertretern für OOA/OOD Notationen hat keinen Anspruch auf Vollständigkeit. Vielmehr soll sie eine Auswahl möglicher Notationen und die jeweiligen Vorlieben und Schwerpunkte aufzeigen.

3.2.1 Business Object Notation

Die Business Object Notation, im folgenden mit BON abgekürzt, ist historisch bedingt, sowohl in einer textuellen als einer graphisch dominierten Beschreibung definiert. Die textuelle Beschreibung basiert auf dem Grundgedanken von Karteikästen mit den dazugehörenden Karteikarten. Die grobgranularen Strukturen der Beschreibung bestehen aus Verbänden von Klassen, Systemen und Einzelbeschreibungen für Klassen. Für jeden Aspekt der Notation existiert in der textuell dominierten Beschreibung eine andere Karteikarte.

CLASS diagram	REGIE	
TYPE OF OBJECT Repräsentation der Diagramdarstellungen		INDEXING
inherits from	SELECTABLE CONQUERED_AREA HAS_MODEL_MANAGER	
Queries	set_local_position	
Commands	move, resize	
Constrains		

Abbildung 3-1: Die textuell dominierte Darstellung nach BON

Diese überwiegend textuell basierte Notation wurde durch graphische Anteile erweitert. Dieser graphische Anteil der Notation wird in EIFFEL-3 durch das Werkzeug EiffelCase unterstützt, genaueres siehe Seite 23.

Die BON-Notation, zumindestens die Bestandteile, die durch das Werkzeug EiffelCase bereitgestellt werden, können automatisch in EIFFEL-Code übersetzt werden. Von daher weist die Notation eine ausgeprägte Vorliebe für diese Sprache auf.

Diese Notation, kombiniert mit dem Werkzeug EiffelCase, wäre prädistiniert gewesen projektbegleitend eingesetzt zu werden. Mit Hilfe des Werkzeuges ist der Entwurf mit der Phase der Implementierung über eine einheitliche Codegestaltung, bedingt durch die automatische Generierung, garantiert. Die Gestaltung dieser Einheitlichkeit erfolgt ansonsten über die Festlegungen von bindenden Codierungsrichtlinien.

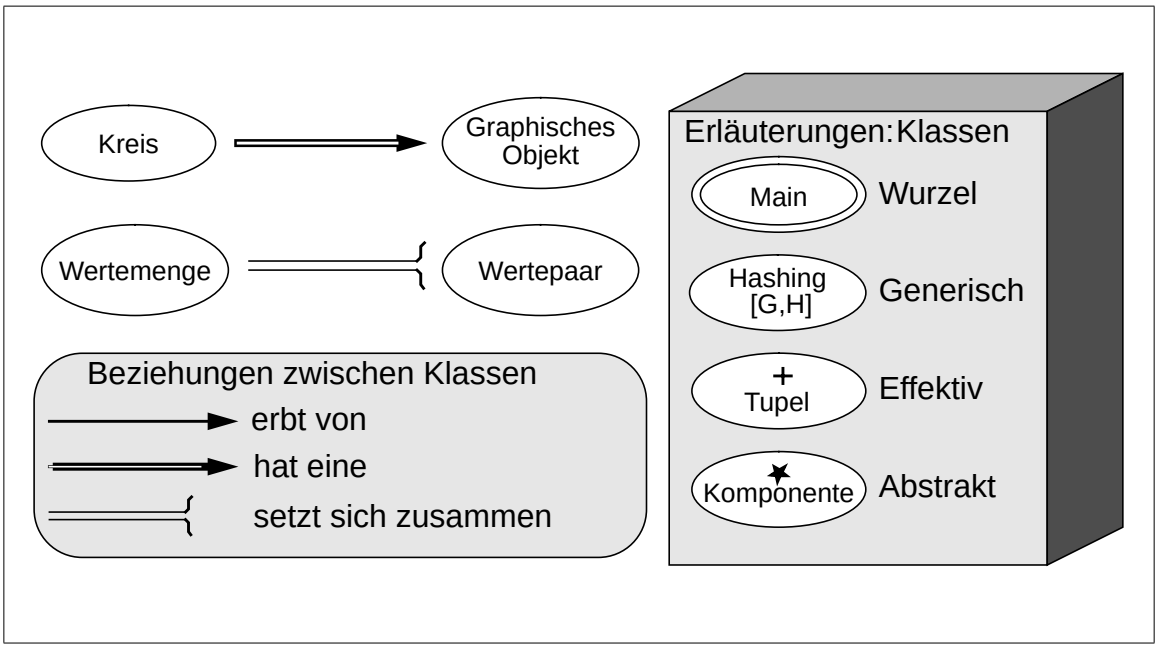

Abbildung 3-2: Graphische Darstellung nach BON

3.2.2 Notation nach Rumbaugh

Diese Notation ist graphisch weniger ausgeschmückt als die beiden anderen. In den unterstützten Konzepten unterscheiden sich die Notationen nach Rumbaugh und Booch nur in dem Botschaftenaustausch und den Echtzeitkriterien [StBa94].

Die Notation nach Rumbaugh zeichnet sich durch ihre einfache Gestalt aus und kann auch leicht mit Papier und Bleistift auf manuelle Art und Weise ausformuliert werden. Sie hat dadurch einen großen Verbreitungsgrad erreicht.

Durch ihre einfache Gestaltung kann die Notation sogar mit individuellen Erweiterungen eines Editors erreicht werden. Zumindestens so, daß die Beschreibungen eine Ähnlichkeit mit Wiedererkennungswert aufweisen. Dieser Methode fehlt natürlich die automatische Überprüfung der Konsistenz des zugrundegelegten Entwurfes.

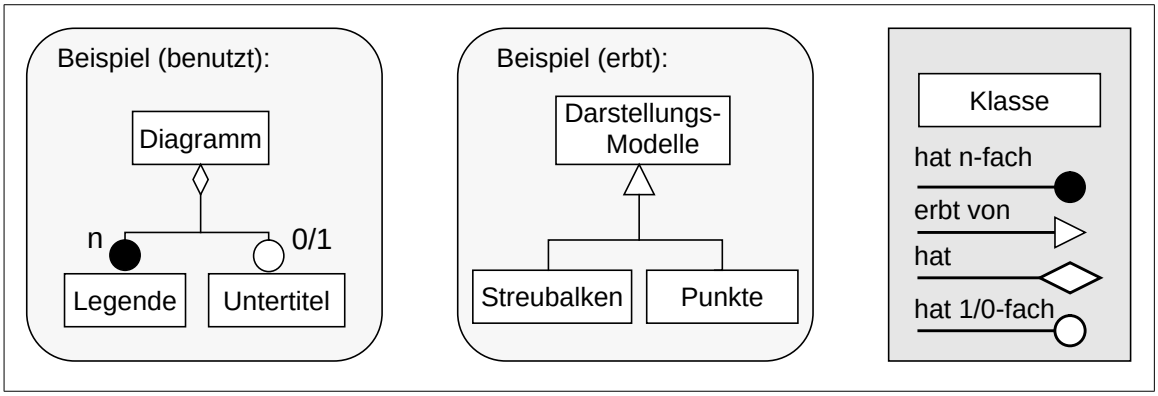

Abbildung 3-3: Notation nach Rumbaugh

3.2.3 Notation nach Booch

Die Notation nach Booch zeichnet sich auf den ersten Blick durch einen gewissen Grad an Verspieltheit in ihren graphischen Einzelkomponenten aus. Charakteristisch für Booch ist die Darstellung der Klassen in Form von „Wolken“. Die Notation ermöglicht einen Entwurf von sehr feiner Granularität und berücksichtigt daher viele Aspekte des Entwurfes. Defizite bestehen im Bereich von Echtzeitkomponenten bei Aspekten wie Reaktionsgeschwindigkeit und Prioritäten. Diese Aspekte finden derzeit innerhalb der Notation noch keine geeignete Repräsentation [StBa94]. Auf die Notation nach Booch wird im Kapitel 1.3 noch genauer eingegangen.

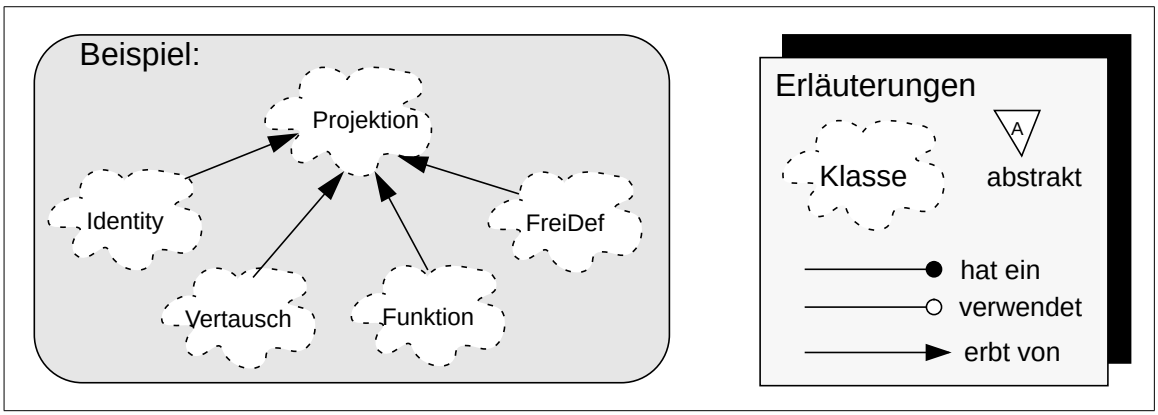

Abbildung 3-4: Beispiel für Booch Notation

Die Booch-Notation weist in ihren feingranularen Entwurfselementen eine ausgeprägte Vorliebe für die Sprache C++ auf. Die Beschreibung von grundlegenden Eigenschaften der Klasse oder die Sichtbarkeit von internen Objektstrukturen in Relation zu anderen Klassen erweist sich als 1:1 Abbildung auf Anteile der Programmiersprache C++ [Boo94][Str91].

3.2.4 Entscheidung über Notation

Die Auswahl fiel zugunsten der Notation nach Booch aus. Sie erfolgte rein intuitiv und nicht nach Kriterien einer objektiven Analyse und Auswertung. Die Notation nach Booch war dem Autor bereits geläufig. Wichtiger war das Argument, daß die Notation zumindest teilweise rechnerunterstützt durchgeführt werden konnte. Auf Anteile der Notation, die wesentlich durch die Kopplung mit C++ entstanden sind, mußte verzichtet werden. Ein Beispiel für solche Kopplungen ist die Möglichkeit, geschachtelte Klassen zu bilden, diese wird in EIFFEL nicht unterstützt [Mey92].

In der Designphase erfolgt unter Booch eine strikte Trennung nach Deklarations- und Implementationsmodulen. Dieser Unterschied ist in EIFFEL nicht erforderlich. Diese Trennung ist ein weiteres Beispiel für die Vorliebe der Notation in Bezug auf die Sprache C++. Auch in der Sprache C++ werden Deklarations- und Implementierungsmodule unterschieden.

3.3 Booch-Notation

Im folgenden Abschnitt wird genauer auf die Booch Notation [Boo94] eingegangen. Die bei dieser Methode zugrunde gelegte Vorgehensweise wird kurz beschrieben.

Die Booch Notation basiert auf folgendem Konzept:

- ein festgelegtes Model für den objektorientierten Entwurf
- eine Unterteilung der objektorientierten Entwicklung in Makro- und Mikroentwicklungsprozesse
- die Visualisierung des Entwurfes mit unterschiedlichen Diagrammarten
- Einführung einer einfachen Metrik als Qualitätsmaß

3.3.1 Modell des objektorientierten Entwurfs

Die Notation basiert auf dem im folgenden beschriebenen Modell des objektorientierten Entwurfes. In der Modellvorstellung wird zwischen zwei unabhängigen Beschreibungen von Eigenschaften eines Softwareprojektes unterschieden, zusätzlich werden die von der Notation bereitgestellten „Methoden“ mit eingebunden.

Drei unterschiedliche Angriffspunkte sind:

- Beschreibung der Entwurfsphase: Analyse und Design. In der Booch-Notation werden diese Phasen auch als logisches und physikalisches Modell bezeichnet.
- die Struktur des Entwurfs, das heißt die Modellierung des statischen und dynamischen Verhaltens

- die Bestandteile der Notation, die sich mit der Beschreibung der Klassen- und Objektstrukturen oder der Modul- und Prozessorarchitektur befassen

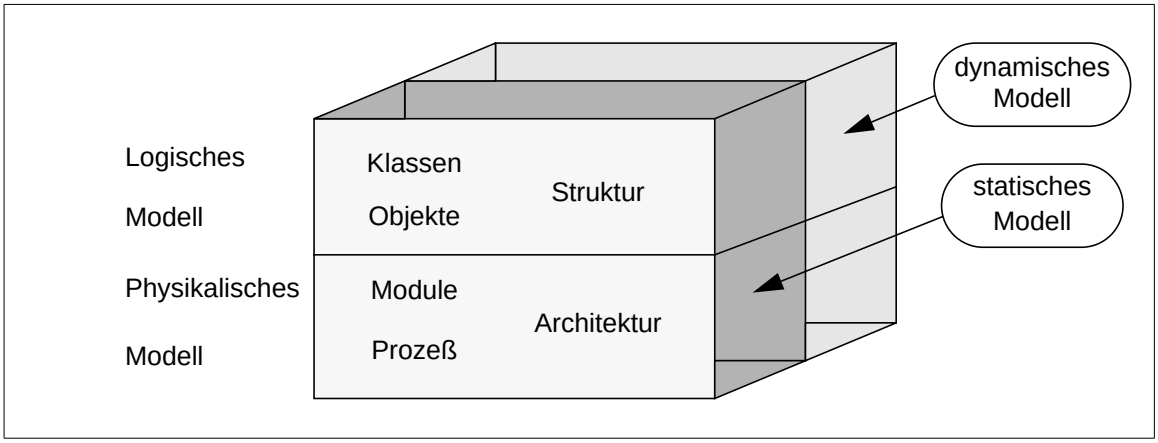

Abbildung 3-5: Entwurfsmodell nach Booch

Wie aus der Abbildung ersichtlich, kann das Modell als einfach und leicht nachvollziehbar bezeichnet werden.

3.3.2 Objektorientierter Entwicklungsprozeß

Der Entwurfsprozeß wird in zwei Granularitäten von Beschreibungen eingeteilt. Zuerst wird der Entwurf im Makroprozeß „grob“ beschrieben. Diese Beschreibung wird in der Phase des Mikroprozesses sukzessive verfeinert.

Die Makrophase umfaßt:

- Konzept: Herauskristallisieren fundamentaler Bestandteile
- Analyse: Entwicklung eines Modelles, daß dem vordefinierten Verhalten genügt
- Design: Erstellen anhand einer realen Architektur
- Implementierung: Codierung des Modelles
- Wartbarkeit: Ausbau und Fehlerbehebung

Die Mikrophase umfaßt:

- Zuordnung der Objekte und Klassen zu den Stufen der Abstraktion
- Festlegen der Semantik von Klassen und Objekten
- Festlegen der Schnittstelle mit nachfolgender Implementierung

3.3.3 Visualisierung durch spezielle Diagrammformen

Die Notation nach Booch wird durch die Gliederung in verschiedene Diagrammgestaltungen strukturiert. Jede der Darstellungsformen ist spezialisiert auf einen Teilbereich des Entwurfes.

Die Diagramme der Analysephase umfaßen:

- Klassendiagramm
- Diagramme für die Beschreibung von Zustandsautomaten
- Objektdiagramme
- Diagramme zur Beschreibung von Szenarien, Zeit-, Ablauf- und Transaktionsverhalten

Die Diagramme der Designphase:

- Diagramme zur Veranschaulichung der Modularisierung
- Physikalisches Abbildung zwischen Prozessen und Prozessoren

Das Klassendiagramm

Wenn in der Entwurfsphase die Grundstruktur der konzeptionellen Grundlage herauskristallisiert ist und die Analysephase beginnt, erfolgt die Suche nach geeigneten Objekten und deren Repräsentation in Klassen. Das Klassendiagramm stellt für diese Phase die geeigneten Darstellungsanteile bereit.

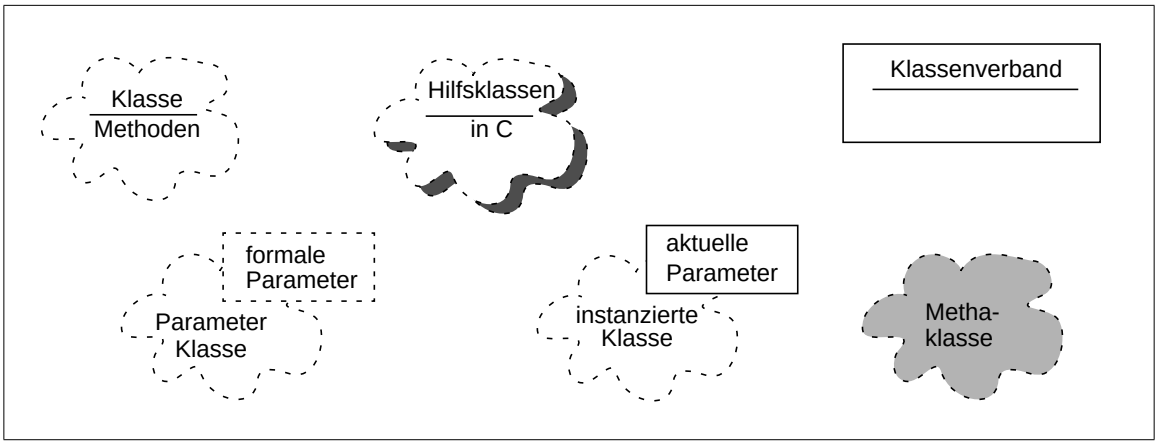

Abbildung 3-6: Die unterschiedlichen Klassen im Klassendiagramm

Das Diagramm setzt sich aus mehreren Beschreibungsanteilen zusammen. Den Hauptbestandteil bilden die Klassen und deren Abhängigkeit untereinander. Bei den graphischen Repräsentationen wird zwischen Kategorien hierarchisch tiefer liegender Verbände von Klassen und einfachen Klassen unterschieden.

Die Beschreibungen von Klassen teilen sich in Metaklasssen und Klassen die genauer spezifizierbar sind. Bei konkreten Ausprägungen von Klassen kann zwischen formalen und aktuellen Parametern der Klasse unterschieden werden. Als weitere Beschreibung für Klassen sind sogenannte „unterstützende“ Klassen vorgesehen. Dieser Notationsaspekt berücksichtigt die Möglichkeit der Sprache C++, auf bereits existierende Module und deren implementierte Funktionen in der Sprache C zurückgreifen zu können, ohne diese explizit in Klassen zu kapseln.

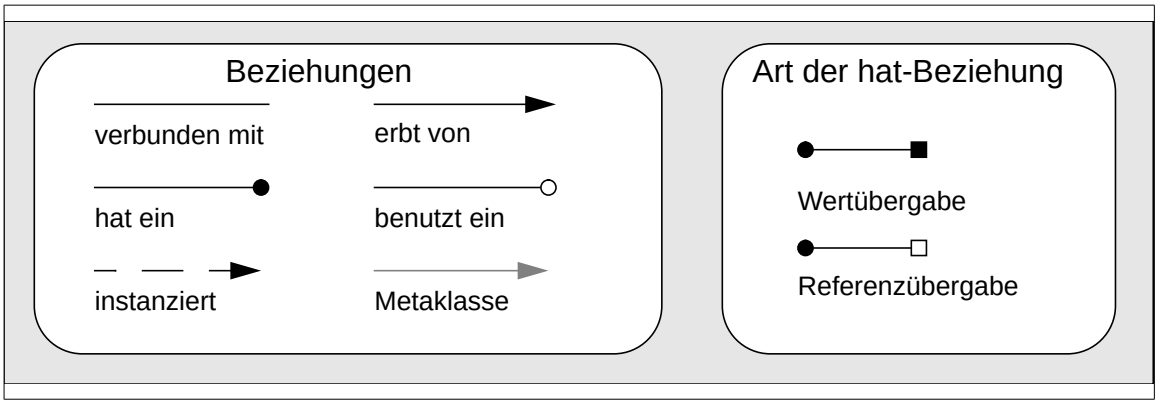

Abbildung 3-7: Booch Klassenbeziehungen

Es wird empfohlen nach Möglichkeit darauf zu verzichten. Die Verwendung solcher „Pseudoklassen“ verwässert und verunreinigt einen Entwurf und zusätzlich besteht das Risiko unzureichender Typisierung.

Bei den Verbindungen der Klassen untereinander wird unterschieden zwischen den Arten: „erben“, „enthalten“ und „verwenden“. Die Relation „enthalten“ teilt sich in Referenz und Wertbindung des Attributes der entsprechenden Klasse auf. Dies ist auf die entsprechende Abbildung in C++ zurückzuführen.

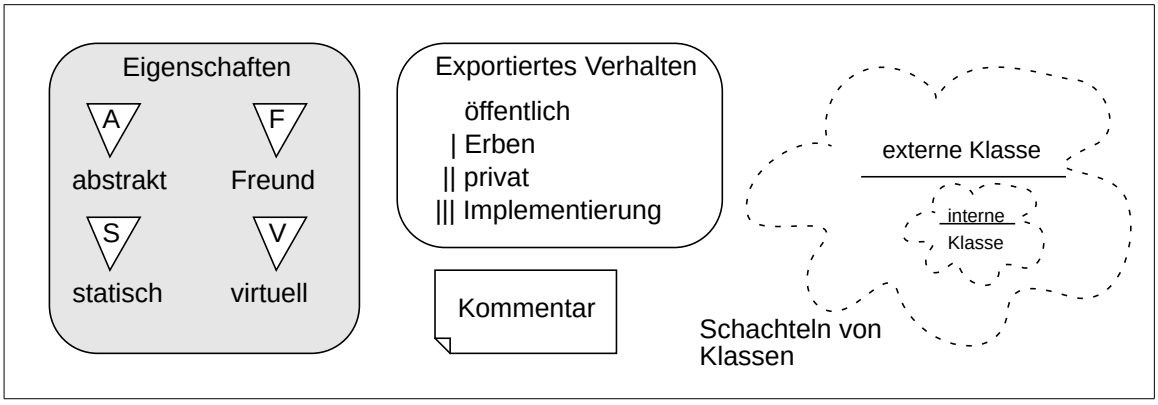

Abbildung 3-8: Zusätzliche Eigenschaften von Klassen

Weiter lassen sich Eigenschaften sowie Exportverhalten von Methoden und Attributen festlegen. Sie sind streng an der sprachlichen Definition und Semantik von C++ ausgerichtet, und lassen sich nicht auf ein ähnliches Verhalten unter EIFFEL abbilden. Es besteht die Möglichkeit Klassen zu schachteln und ihnen Kommentare und Notizen anzuhängen.

Das Diagramm zur Beschreibung der Zustandsautomaten

Ein wichtiger Bestandteil des objektorientierten Entwurfes ist die Beschreibung von Verhalten durch endliche Automaten. Oft erweist es sich als viel zu schwierig das Systemverhalten in einem einzigen, flachen Automaten zu beschreiben. Daher sollten große, endliche Automaten über eine Hierarchie von Automaten aufgebaut werden. Dies erfordert die Möglichkeit der Schachtelung von Automaten.

So wird der komplizierte, flache Automat in eine Hierarchie, bestehend aus einfachen Automaten, überführt. Der Begriff Hierarchie erscheint unpassend, da es sich von der Gestalt her um einen beliebigen Graph handeln kann. Es entspricht aber der Vorgehensweise bei dessen Konstruktion. Die Booch Notation unterstützt Beschreibungen von Zuständen, Übergängen und Schachtelungen von Zustandsautomaten.

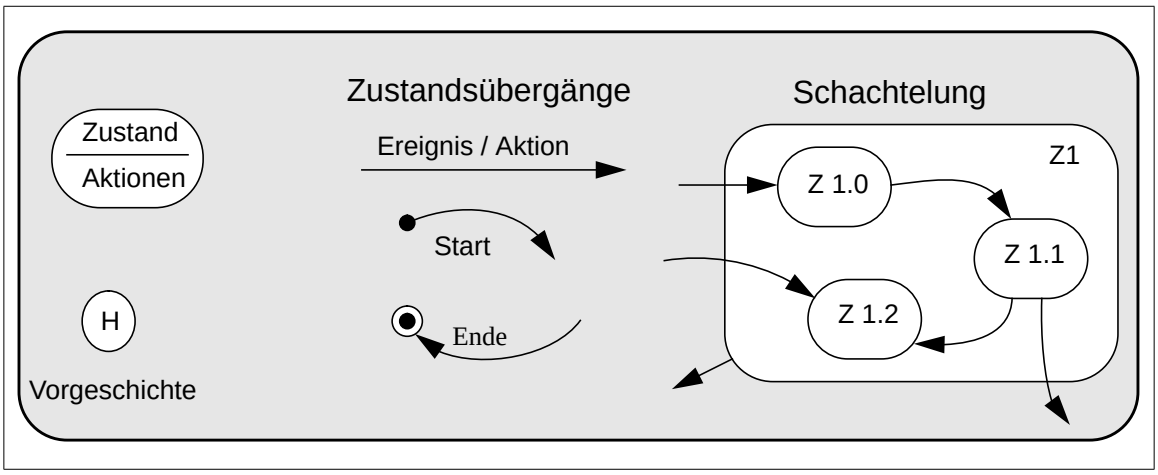

Abbildung 3-9: Beschreibung der Zustandsautomaten nach Booch

Das Objektdiagramm

Im Gegensatz zur generellen Beschreibung des Verhaltens in Klassen wird in der Objektbeschreibung das Verhalten einzelner Ausprägungen festgelegt. Dabei sind unterschiedliche Objekte miteinander verbunden. Diese Verbindungen werden auch als „Links“ bezeichnet. Die Verbindungen werden für den Austausch von Nachrichten verwendet. Das Zeitverhalten eines solchen Nachrichtenaustausches läßt sich spezifizieren. Zusätzlich lassen sich die Sichtbarkeitsregeln der internen Strukturen von

Objekten festlegen. Leider ist das Raster der Sichtbarkeitsregeln sehr unscharf. Es werden nur drei Arten in Anlehnung an C++ unterschieden: private (Klasse selbst), geschützte (Erben der Klasse) und öffentliche Methoden und Attribute.

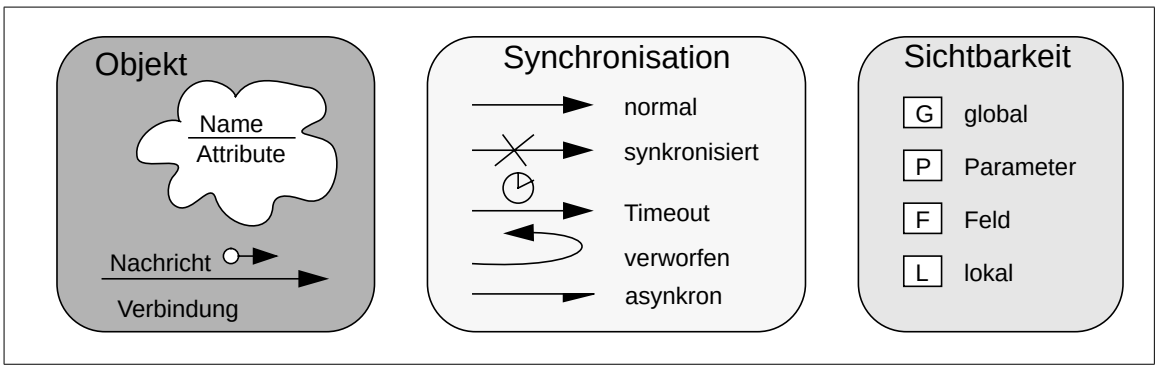

Abbildung 3-10: Objektdiagramm

Diagramm zur Beschreibung von Verhalten

Bei größeren Szenarien, die eine Beschreibung von Transaktionen oder einen zeitlichen Ablauf beschreiben sollen, werden diese Diagramme zur Darstellung des Verhaltens verwendet. Zusätzlich zu den Zustandsautomaten bringen sie ein Ablaufverhalten ins Spiel.

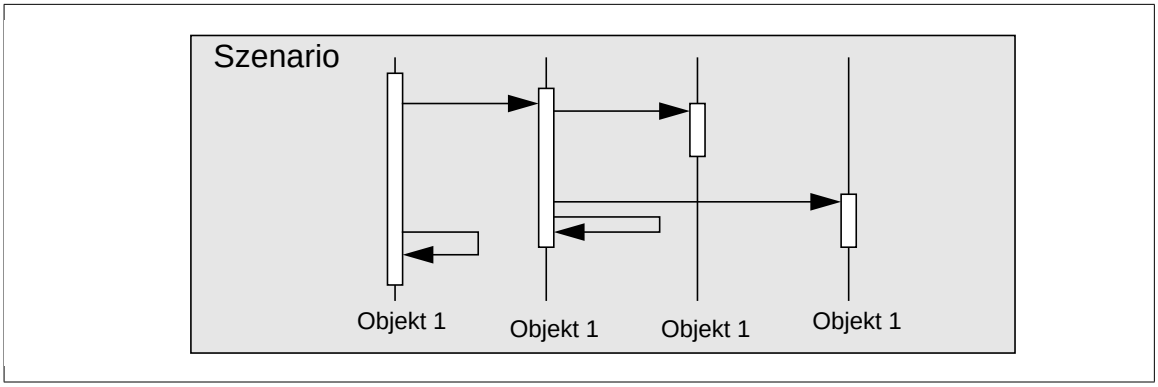

Abbildung 3-11: Beschreibung von Transaktionsverhalten

Diagramme der Designphase: Modul- und Prozeßdiagramm

Die Beschreibung der Designphase besteht in der Kopplung der Klassen mit den Modulen und einer Beschreibung der Prozesse und Prozessoren.

Als Bestandteile werden bereitgestellt:

- Hauptmodul
- Deklarationsmodul
- Implementierungsmodul

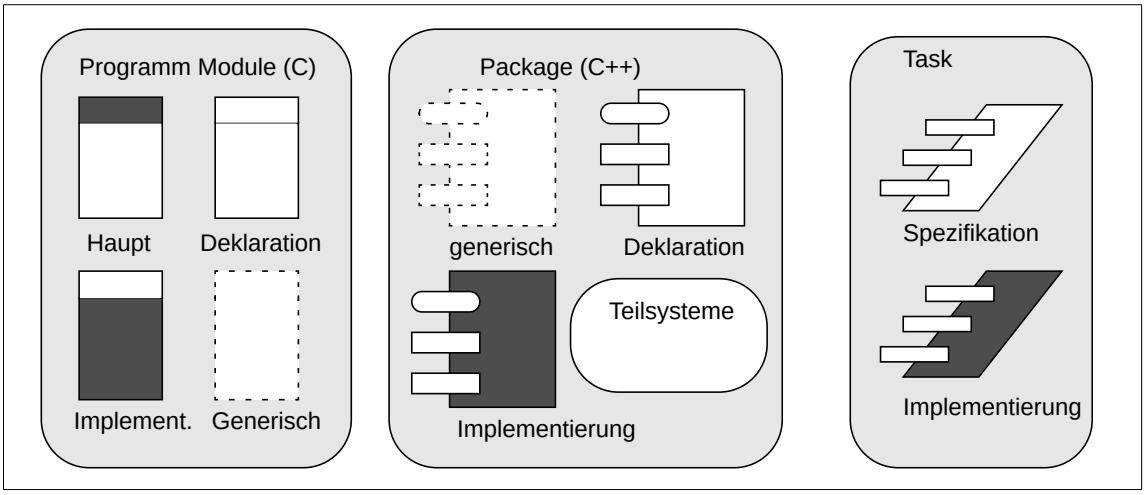

Abbildung 3-12: Modulbeschreibung

Die Beschreibung der Modulstruktur kann in Teilsysteme geschachtelt werden. Die Beschreibung im Prozessdiagramm beschränkt sich auf die Verwendung von Prozessen, Prozessoren, den Zugriff auf externe Peripheriegeräte und deren Verbindungen untereinander. Ob eine Beschreibung mit fester Kopplung zwischen Prozessen und Prozessoren sinnvoll ist, sei dahingestellt.

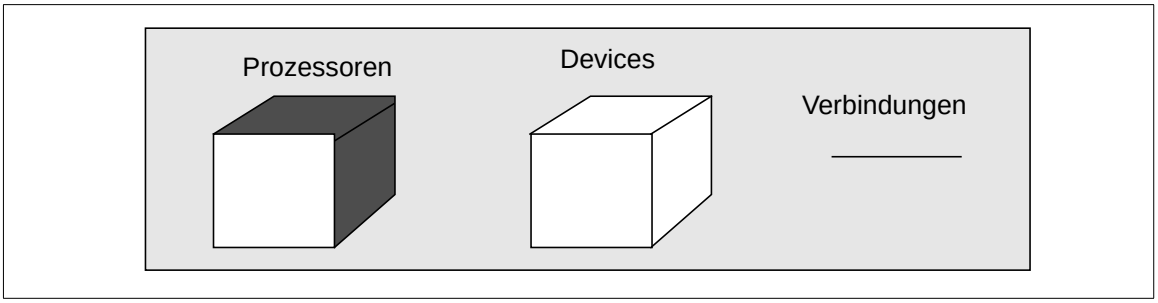

Abbildung 3-13: Prozeßbeschreibung

3.3.4 Vorstellung einer einfachen Metrik

Dem Entwurfsprozeß nach Booch sollte als Hintergedanken immer eine Metrik zur Bewertung des Entwurfs zugrunde gelegt werden. Ein einfaches Beispiel ist:

- Das Qualitätsverhalten des Systemes: Stabilität, Fehlersicherheit, Fehlerrate
- Bewertung der Größe und Komplexität: Absolute Anzahl der Klassen, Anzahl der Klassen pro Verband
- Qualitätskriterien für den Klassenentwurf: Anzahl von Operationen, Querschnitt der Vererbungsstruktur, Anzahl von Erben

4
Umgebung der Implementierung

Jede Implementierung setzt die Festlegung der Entwurfs- und Implementierungsumgebung voraus. Da diese Randbedingungen weitreichende Auswirkungen auf die konkrete Implementierung aufweisen, werden diese eingehend beschrieben. Die wichtigste Randbedingung der Implementierungsphase ist die Wahl einer geeigneten Programmiersprache. Weitere wichtige Randbedingungen bilden die bereitgestellten Werkzeuge und wann deren Einsatz sinnvoll und möglich ist.

Die Werkzeuge gliedern sich in solche, die in enger Verbindung zu der gewählten Sprache stehen und solche Werkzeuge, die generell, somit projektübergreifend, eingesetzt werden. Bei diesen Werkzeugen, zum Beispiel einem Editor, stellt sich die Frage, ob dieser durch individuelle Anpassung besser in das Projekt eingebunden werden kann.

4.1 Die Programmiersprache

Das in der Entwurfsphase auferlegte objektorientierte Paradigma, verlangt die Auswahl einer objektorientierten Sprache. Dadurch sind die in der Entwurfsphase ausformulierten Bedingungen einfacher auf die Implementierung abzubilden. Zu den Vertretern von objektorientierten Sprachen zählen weitverbreitete Vertreter wie C++[Str91], Smalltalk[GoRo89], sowie EIFFEL [Mey92].

Die Entscheidung fiel zugunsten von EIFFEL, weil es gegenüber Smalltalk oder C++ Vorteile aufweist, genauer gesagt Unzulänglichkeiten dieser Sprachen behebt. Leider mußte aus Mangel an Kenntnissen und Erfahrungen über Smalltalk auf einen genaueren Vergleich mit EIFFEL verzichtet werden.

4.1.1 Smalltalk

Die Sprache Smalltalk basiert auf einer Bytecode-Maschine, welche den Zwischencode, der von einem Interpreter erzeugt wird, abarbeitet. Eine häufige Fehlerquelle ist die dynamische Typisierung der Sprache. Dies zeigt sich insbesondere in Projekten an denen mehrere Entwickler arbeiten.

Der fehlende Test auf typkonforme Verwendung führt zu den schon sprichwörtlichen „Message not understood“ - Fehlermeldungen. Die dynamische Typisierung verlangt eine Verschiebung der Gewichtung von Entwicklungsphasen; der Testphase wird mehr Bedeutung zukommen, als es bei anderen Sprachen, die auf einem Compiler basieren, der Fall ist.

4.1.2 C++

Die Sprache C++ als rein objektorientiert zu bezeichnen, ist aufgrund der Möglichkeit der Einbindung von prozeduralen Modulen, nicht ganz nachzuvollziehen. Diese Einbindung kann ohne vorherige Kapselung in Klassen durchgeführt werden.

Die objektorientierte Sprache C++ basiert auf der gleichen Grundannahme wie ihr Vorgänger, die prozeduale Sprache C, bezüglich der Einschätzung von Fähigkeiten ihrer Entwickler. Die Ursachen weitverbreiteter Fehler sind daher ähnlich. Diese Grundannahme lautet: „Der Entwickler weiß immer genau, was er tut und allen seinen Wünschen ist Folge zu leisten“. Dies ist sehr überspitzt formuliert, trifft aber die Grundsatz der Sprache. Diese Wunschvorstellung mag auf versierte Entwickler zutreffen, eine solche Annahme über Teams und deren Projekten zutreffen, ist grob fahrläßig.

Wird C++ als Programmiersprache eingesetzt, ist es wichtig, daß die Entwickler sich vor Beginn über die Richtlinien der Codierung einigen und sich gegenseitig durch diese disziplinieren. Zu diesem Zwecke sollten auch „Reviews“ durchgeführt werden. Der Entwurf und die Implementierung sollten so durchgeführt werden, daß die Module auch von anderen beteiligten Entwicklern in angemessener Zeit nachvollzogen und bearbeitet werden können.

4.2 EIFFEL

Die Sprache EIFFEL beseitigt die gesamten Unzulänglichkeiten der Sprachen C++ und Smalltalk. EIFFEL ist im Gegensatz zu Smalltalk streng typisiert und basiert auf der Verwendung eines Compilers. Im Unterschied zu C++ zwingt EIFFEL den Entwickler objektorientiert zu programmieren, wogegen in C++ eine prozeduale Programmierung zwar nicht sinnvoll, aber durchaus möglich ist. In folgenden Abschnitten werden einige Konzepte der Sprache EIFFEL vorgestellt. Es werden nur hervorzuhebende Eigenschaften aufgelistet. Dies soll keine Sprachbeschreibung sein, hierfür wird auf [Mey92] verwiesen.

4.2.1 Vereinbarung einzuhaltender Bedingungen

Ein wichtiger Bestandteil der Sprache EIFFEL ist die Verwendung sogenannter Verträge zwischen aufrufenden und aufgerufenen Methoden. Dieser Bestandteil ist ein fundamentales Konzept der Sprache zur Qualitätsicherung. Es werden Vereinbarungen getroffen, die als Vor- und Nachbedingung der aufgerufenen Methode explizit ausformuliert werden.

```
Class Konto

abbuchen( ein_Betrag:WAEHRUNG )
     -- buche Betrag vom Konto des Besitzers ab
require
     Kunde_Liquide: ( kein_Limit_erreicht( ein_Betrag )
do
     ...
ensure
     Abbuchung_korrekt: ( kontrolliere_Buchung( ein_Betrag, old ein_Betrag )
end; -- abbuchen

invariante

     Kontoinhaber: Ist_Juristische_Person( der_Besitzer );
end -- Konto
```

Beispiele 4-1: Verträge zwischen Methoden und Klasseninvarianten

Die Vorbedingungen definieren Kriterien, denen die aktuellen Parameter der Eingabe genügen müssen. Diese Bedingungen werden durch das Schlüsselwort „*require*“ eingeleitet. Analog können Vereinbarungen über das Ergebnis einer Methodenausführung definiert werden, diesen geht das Schlüsselwort „*ensure*“ voran.

Konsistenzkriterien einer Klasse können als Invariante der Klasse gefordert werden. Diese Invarianten dürfen nur innerhalb der Ausführung einer Methode dieser Klasse verletzt werden; vor Eintritt und nach Verlassen der Methode müssen die Bedingungen zwingend erfüllt sein. Diese Zusicherungen lassen sich für jede Klasse des Systemes getrennt ein- und abschalten.

4.2.2 Ausnahmenbehandlung (Exception)

Bei der Verwendung von Vereinbarungen und Verträgen zwischen Methoden, besteht die Möglichkeit, diese zu verletzen. In diesem Fall tritt eine Ausnahmebehandlung in Kraft, die auch als Sonderfall bezeichnet wird. Diese Ausnahme gilt es abzufangen und gezielt zu bearbeiten. Das Fehlschlagen der Ausführung einer Methode ist ein weiterer Grund, einen aufgetretenen Sonderfall anzuzeigen. Dieses Signal wird an die aufrufende Methode weitergereicht. Die aufrufende Methode kann auftretende Signale für Ausnahmen ihrer aufgerufenen Methoden durch die Schlüsselworte „*rescue*“ mit

anschließender, erneuter Ausführung der Methode über „*retry*“ versuchen zu beheben. Der „retry“ ist nichts anderes, als daß die Methode an ihrem Startpunkt wieder aufsetzt und eine erneute Ausführung des Methodenrumpfes versucht wird. Dies verlangt das im „Rescue“ Bereich vor der Durchführung eines „retry“ alle bisher erfolgreichen „Modifikationen“ rückgängig gemacht werden.

Zur Verhinderung endlos wiederkehrender Aufrufe aus Ausnahme und erneutem Versuch sollte der Versuch mit Zählern kombiniert und bei Fehlschlagen der Methode ein Status extern gespeichert werden. Dieses Konzept der Behandlung von Sonderfällen ist wichtig und sollte bei Verwendung vertieft werden [Mey94][Str91].

```
alarm_rufe_polizei is
     -- alarm vn sensor gemeldet rufe polizei
local
     zeit_marke:ZEIT_NACH_VERWALTUNG
do
     if ( zeit_marke < maximal_zugelassene_marke ) then
          versuche_neuen_Anruf( 110 );
     else
          protokolliere_fehlschlag;
     end; -- if
rescue
     zeit_marke := fordere_neue_Zeitmarke_an;
     retry;
end; -- abbuchen
```

Beispiele 4-2: Ausnahmebehandlung in EIFFEL

4.2.3 Generische Klassen

Um bei objektorientierter Programmierung dem Gesichtspunkt der Codewiederverwendung gerecht zu werden, muß die Erzeugung generischer Klassen unterstützt werden. Dabei handelt es sich vorwiegend um Verwaltungstrukturen, die von spezialisierten Klassen verwendet werden. Einfache Beispiele sind ARRAY[G] und BINARY_TREE[G].

Die formalen Parameter können auf spezielle Klassen eingeschränkt werden, wie zum Beispiel bei der Klasse BINARY_SEARCH_TREE[G->COMPARABLE]. Hier müssen die aktuellen Parameter der Klasse indirekt von der Klasse COMPARABLE erben. Es lassen sich auch Klassen mit mehreren Parametern spezifizieren, ein Beispiel hierfür wäre GRAPH[G->KNOTEN, H-> KANTEN].

4.2.4 Vererbung

Ein wesentliches Merkmal der Sprache EIFFEL ist die feingranulare Beschreibungsmöglichkeiten im Bereich bereitgestellter Vererbungsmechanismen. EIFFEL unterstützt wie C++ Mehrfachvererbung. Die bei dieser Vererbungsvariante auftretenden Schwierigkeiten, beispielsweise durch Erben von Klassen die Methoden mit gleicher Bezeichnung enthalten, werden durch umfangreiche Mechanismen berücksichtigt.

Welche Konflikte können bei der Vererbung auftreten und wie werden diese durch sprachliche Bestandteile von EIFFEL gelöst:

Problem	Ansatz	Schlüsselwort
Geerbte Merkmale mit gleichem Bezeichner	Umbenennen von Bezeichnern	**rename**
Exportverhaltens, also Sichtbarkeitsregeln von geerbten Methoden und Attributen sollen nicht übernommen werden	Nachträgliches ändern dieser Eigenschaften, feingranular auf Klassenebene	**export**
Methoden, die geerbt wurden aber deren Verwendung überflüßig oder im Zusammenhang fehlerhaft ist	Entfernen dieser Methoden	**undefine**
Überlagern von Methoden mit eigener Semantik	Bereistellung von Mechanismen um eine neue Semantik zu beschreiben	**redefine**
Lassen sich Methoden bei wiederholtem Erben auf die selben Ausgangsmethode einer Klasse zurückführen, so entsteht Mehrdeutigkeit bei Zuweisungen auf diese Eltern-Klassen in Bezug auf die Verwendung der Ausgangsmethode.	Behebung dieser Mehrdeutigkeit	**select**

Tabelle 4.1: Mehrfachvererbung, Probleme und Behebung

Zur Verdeutlichung der Probleme, die sich aus der Mehrfachvererbung bei wiederholtem Erben ergeben können ein kleines Beispiel. Bei anschließender Zuweisung würde eine Mehrdeutigkeit entstehen, die es aufzulösen gilt.

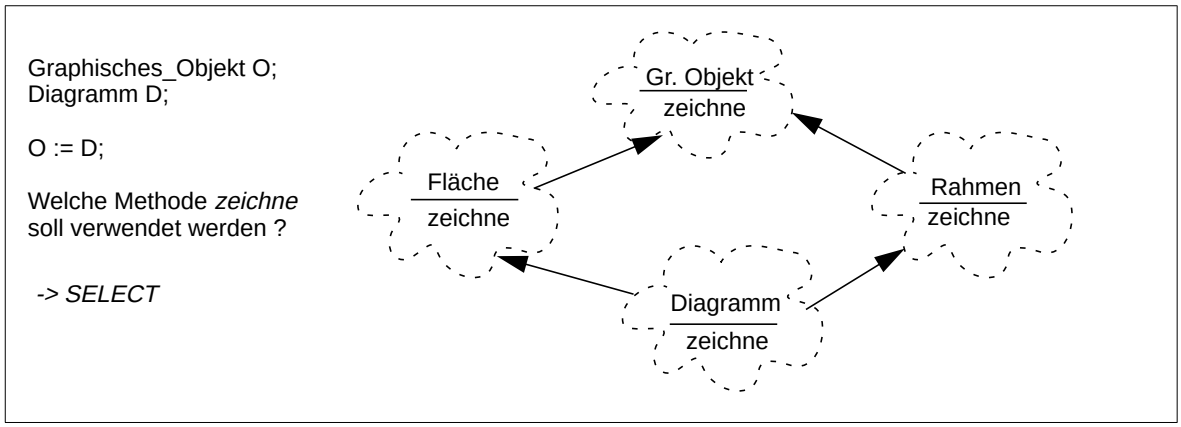

Abbildung 4-1: Probleme bei wiederholtem Erben

Um ein Gefühl für die Stärken von EIFFEL bei der Beschreibung von Vererbungenbeziehungen zu geben, hier ein kommentiertes Beispiel.

```
class Tupel[G->DOUBLE]
     -- verwaltet Wertepaare der Wertemenge
inherit

     TUPEL_COMPARABLE[G]
          rename
               make as make_tupel_comparable
          export
               {VALUESET} compare
          end; -- TUPEL_COMAPARABLE

     TUPEL_NUMERIC[G]
          rename
               make as make_tupel_numeric
          end;-- TUPEL_NUMERIC

end -- Tupel
```

Beispiele 4-3: Mehrfachvererbung

4.2.5 Polymorphismus und dynamisches Binden

Polymorphismus ist gemeinsam mit dem dynamischen Binden ein sehr mächtiger Mechanismus der objektorientierten Programmierung.

```
redraw_all_elements_in_container is
     -- zeichne recursiv alle Objekte des Containers
local
     a_container:CONTAINER;
do
     from
          elements.start
     until
          elements.after
     loop
          elements.draw;

          a_container ?= elements.item;
          if ( a_container /= Void ) then
               a_container.redraw_all_elements_in_container;
          end; -- if

          elements.next
     end; -- loop
end; -- redraw_all_elements_in_the_container
```

Beispiele 4-4: Polymorphismus und dynamisches Binden

Um Polymorphismus anwenden zu können, ist statische Typisierung notwendig, diese wird im weiteren Kontext als „*Signatur*“ bezeichnet. Polymorphie bedeutet, daß eine Klasse sich verhalten kann, wie eine Klasse ihrer Vererbungshierarchie.

Werden Methoden redefiniert, so müssen die Methoden aus diesem Grund ein konforme SIGNATUR zur ursprüngliche Methode aufweisen. Die neue Methoden lassen sich dann problemlos anwenden, da sie eine konforme Signatur besitzen.

4.2.6 Abstrakte Klassen

EIFFEL bietet wie andere objektorientierte Sprachen die Möglichkeit, abstrakte Klassen zu definieren. Solche Klassen dürfen keine eigene reale Ausprägung besitzen, können aber Methoden und Attribute festlegen oder abstrakte Methoden deklarieren und so Signaturen festlegen. Die abstrakten Methoden erzwingen in effektiven Klassen eine effektive Methode der gleichen Signatur.

4.2.7 Die Speicherverwaltung

Es werden zwei unterschiedliche Vorgehensweisen der Speicherverwaltung getrennt angeboten. Der erste Ansatz basiert auf der expliziten Anforderungen und Freigabe von Speicher. Der andere Ansatz führt zu einer impliziten, automatischen Vergabe ohne Einflußnahme des Entwicklers.

Die einfache, fehleranfällige Methode ist die explizite Anforderung an die Speicherverwaltung. Das heißt, der Entwickler fordert Speicherbereiche an und gibt diese später frei in der Hoffnung, daß keine weiteren Verweise auf den gleichen Speicherbereich existieren. Erfolgt bei Freigabe nicht höchste Disziplin und Koordination zwischen Entwicklern, ist ein „Minenfeld“ aus Fehlern im Speicherzugriff vorprogrammiert.

Als Alternative bietet sich eine Speicherverwaltung mit sogenannter „Garbage Collection“ an. Diese Bestandteil des Laufzeitsystemes führt implizit die Speicherverwaltung durch. Der benötigte Speicher wird belegt und in festen Zeitintervallen wird überprüft, ob Bereiche existieren zu denen keine Verweise existieren. Diese werden dann freigegeben.

Welche Methode vorteilhafter ist, ist durch die Rahmenbedingungen von Projekten festgelegt. Unzweifelhaft ist die Methode unter Verwendung der automatischen Speicherrückgewinnung weniger fehleranfällig.

Der Vorteil, der expliziten Anforderung und Freigabe liegt bei kurzzeitigem, hohen Speicherbedarf, wie er zum Beispiel bei aufwendigen Berechnung oder Datenaustausch unter Echtzeitkriterien anfallen kann. Hier kann der Entwickler nach abgeschlossener Ausführung den Speicher sofort freigeben, wogegen die „Garbage Collection“ erst ein Zeitintervall verstreichen lassen muß.

Die in zeitlichen Abständen immer wiederkehrende Speicherbereinigung wirkt sich bei Applikationen mit gewünschtem Echtzeitverhalten als K.O.-Kriterium aus. Ein zweites Argument für die explizite Speicheranforderung, sind ungenutzte Referenzen, die fehlerhafterweise nicht wieder initialisiert werden und dadurch Speicherleichen nachsichziehen.

EIFFEL verwendet voreingestellt die Speicherverwaltung mit automatischer Rückgewinnung. Glücklicherweise wird trotzdem die Möglichkeit einer expliziten Anforderung und Freigabe von Speicher über eine Klasse MEMORY offengelassen. Die Nutzung dieser Klasse sollte jedoch mit Bedacht und nur punktuell geschehen.

4.3 Werkzeuge der Programmierumgebung

Die von ISE-EIFFEL bereitgestellten Werkzeuge umfassen eine „*Workbench*“. Diese unterstützt die Programmierung durch unterschiedliche Ansichten auf die Klassenhierarchie und bei der Fehlersuche. Der Oberflächengenerator EiffelBuild ist bei der Erstellung von Dialogfenstern und Hauptfenstern behilflich. Der Entwurfsprozeß wird durch das Werkzeug EiffelCase unterstützt. Als generelles Werkzeug wird der Editor EMACS ausgewählt und dessen Nutzbarkeit unter EIFFEL kurz skizziert.

4.3.1 Der EIFFEL Compiler

Der EIFFEL-Compiler von ISE arbeitet nach dem Prinzip des „sukzessiven Einfrierens“ von Code in der Implementierungsphase [Mey90]. Das zugrunde gelegte Konzept ist ein Aufbrechen der Implementierung in drei unterschiedliche Vorgehensweisen. Jede der Vorgehensweisen bewertet Kriterien wie Qualität des Zielcode oder der Zeitaufwand für einen geschlossenen Compile-Edit Zyklus anders. Diese Vorgehensweise wird im folgenden als „*Melting*“, „*Freeze*“ und „*Finalize*“ bezeichnet und ist mit unterschiedlichen Teilabschnitten bei der Entwicklung von Software gleichzusetzen.

Unter dem Vorgehen beim „*Melting*“ wird verstanden, daß kleine Änderungen an bestehenden Klassen vorgenommen werden. Es wird ein neuer Verbund aus Klassen aufgebaut, die der Entwickler nach Programmierung von einigen Methoden, kurz ausprobieren kann. Der Entwickler will bei dieser Vorgehensweise kurze Compile-Try-Edit Zyklen durchführen, um den bestehenden Code zu erweitern. Unter kürzeren Übersetzungszeiten werden solche verstanden, die nicht wie in C++ bei kleiner Änderung in einer Datei eine Übersetzung des kompletten Projektpaketes über mehrere Stunden oder gar Tage erforderlich machen.

Beim „*Freeze*“ werden sämtliche Änderungen der Meltingphase, also aller Klassen die zwischenzeitlich modifiziert wurden, in einer längere Übersetzung im Bereich Laufzeitverhalten, optimiert. Dieser Vorgang sollte in festgelegten Zeiträumen wiederholt werden, um Übersetzungen nach dem Meltingprinzip nicht zu überfordern.

Ein „*Finalize*“ erfordert einen sehr viel aufwendigeren Übersetzungsaufwand. Der Vorgang führt in mehreren Ausführungschritten eine Entfernung von nicht verwendetem Code durch. Der Compiler entfernt und analysiert bei diesem Vorgang mit einer Rastergröße, eingestellt auf Methoden. Das heißt, daß sämtliche Methoden der Klassen die im System nicht verwendet werden, aus dem endgültigen Code beseitigt worden sind.

Diese Übersetzung kann daher bei großen Projekten, die 50000 und mehr Methoden umfassen, sehr aufwendig sein. Sie führt aber bei Verwendung von Bibliotheken aus Klassen, bei denen pro Klasse nur eine Handvoll verschiedener Methoden verwendet werden, zur erheblicher Verkleinerung des ausführbaren Programmes. Aus Gründen der Kontrolle von Zeitverhalten und Größe des ausführbaren Programmes sollte dieser Vorgang alle paar Wochen durchgeführt werden.

Der überwiegende Teil der Entwicklung bedient sich der Übersetzung nach dem Prinzip des „*Meltings*“. Der Entwickler teilt seine Arbeit in kleine Teilabschnitte ein. Ist ein Abschnitt erledigt wird ein „*Freeze*“-Vorgang eingeleitet.

Als eine erhebliche Erleichterung für Entwickler stellt EIFFEL die Berechnung des Abhängigkeitsgraphen zwischen den eingebundenen Klassen, wie er für die Erzeugung von Make-Dateien notwendig ist, bereit. Es ist für den Entwickler nicht erforderlich durch topologische Sortierung, explizite Abhängigkeiten zwischen den Klassen aufzustellen. Dies ist Bestandteil des Übersetzungsvorganges. Manuelle Fehler und die dadurch bedingten zyklischen Aufrufe bei der Erzeugung werden so umgangen.

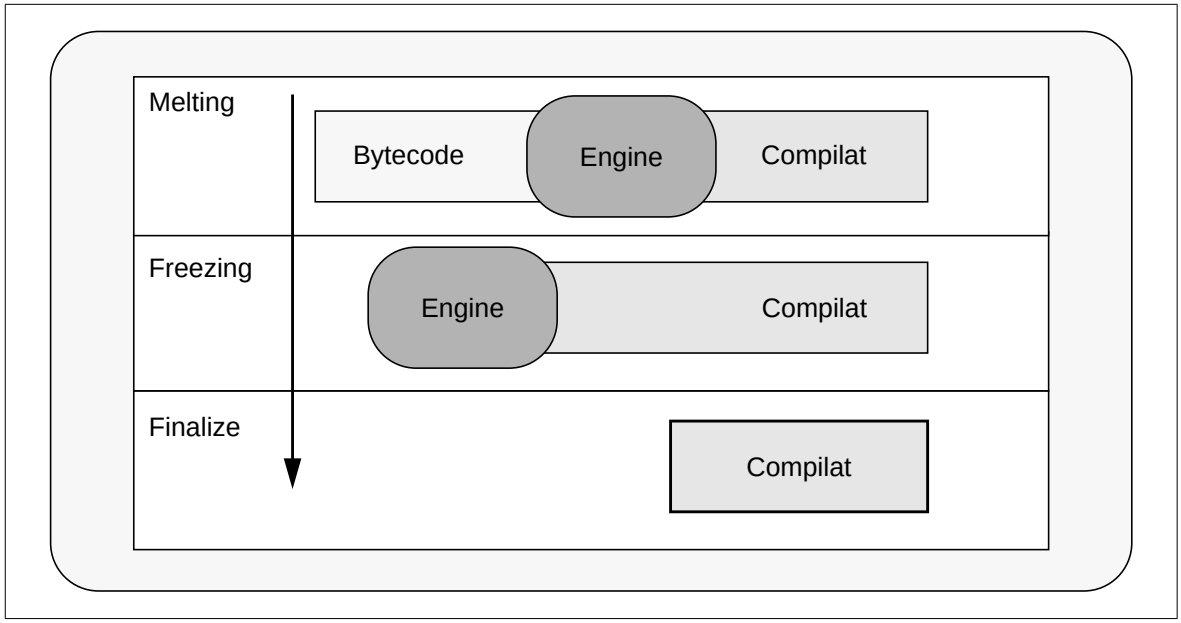

Abbildung 4-2: Ausführungsmodus im ISE-EIFFEL-Compiler

Bei ISE-EIFFEL gestaltet sich die Abbildung der einzelnen Vorgehensweisen der Implementierung sehr aufwendig: Um die Übersetzungszeiten möglichst klein zu halten, besteht das Compile- und Laufzeitsystem aus einem Gespann von compiliertem Code und einem erweiterbaren Bytecode, der über eine Bytecode Engine interpretiert wird. Wird eine Methode neu geschrieben oder überarbeitet, so wird ihre compilierte Übersetzung durch den interpretierbaren Anteil ersetzt. Dieses Konzept reduziert den Übersetzungsaufwand erheblich, aber es entsteht ein schlechteres Laufzeitverhalten. Bei häufigem „*Melten*“ mit sehr vielen einbezogenen Methoden wächst der Bytecode schnell über ein erträgliches Maß hinaus, die Übersetzung wird langsamer und das Laufzeitverhalten wird schlechter. Das ist der Zeitpunkt an dem der Entwickler den „*Freeze*“ Vorgang durchführen sollte, um bestehende Änderungen aus dem Bytecode in den compilierten Code zu verschieben.

Der Freezevorgang ist ein kompletter Übersetzungsvorgang bei dem C als Zwischencode erzeugt und dieser Zwischencode in einer anschließenden Übersetzung das Programm erzeugt wird. Es werden Optimierungen dahingehend durchgeführt, daß die verwendeten Klassen auf die nicht zugegriffen wird, nicht als Zwischencode in C generiert werden. In der Übersetzung des C-Codes wird natürlich auch optimiert.

Im Gegensatz zum Vorgehen beim „Freeze“ führt der „Finalize“ eine zusätzliche Optimitierung zur Entfernung nicht verwendetem Codes durch. Diese Entfernung erfolgt im Feinraster der Methoden. Die Optimierung ist aufwendig und wird in mehreren Stufen durchgeführt. Als Erstes wird ein Zwischencode mit Endung „.x“ generiert, aus dem in einer weiteren Übersetzung C Module erzeugt werden.

Die Bereitstellung des Vorgangs nach „Melting Ice“-Prinzip führt zu Einschränkungen. Bisher wird für Übersetzungen nur eine vorkompilierte Bibliothek zugelassen. Diese wird in die Übersetzung nicht einbezogen. Eine Erweiterung des Konzeptes dahingehend, daß mehrere Bibliotheken zulässig sind, wird zukünftig sicher durchgeführt werden. Diese Erweiterung würde das Arbeiten mehrerer Entwickler am gleichen Projekt erheblich erleichtern.

Als schwerwiegende Einschränkung ist, wegen der Verzahnung von bereits kompilierten Programmanteilen mit einem über eine Bytecode Engine interpretierten Anteil, das Fehlen eines Konzept für den Bereich Multitasking und Multithreading anzusehen. Es ist dringend notwendig ein Konzept zur Unterstützung von Multithreading anzubieten. Hier werden zukünftig einige Änderungen in Bezug auf das bisherige Konzept erforderlich sein. Diese sind notwendig, da es Anwendern bei aufwendigeren Berechnungen und deren Darstellung nicht zuzumuten ist, auf Ereignisse zu warten und zwischenzeitig von der Interaktion abgekoppelt zu werden. In diesem Bereich ist bei tragfähigem Konzept mit Umstellungen zu rechnen, die in ihrer Tragweite auch Auswirkungen auf bereits abgeschlossene Projekte haben werden.

Nicht alle EIFFEL-Compiler unterstützen die sehr aufwendigen und mit einigen Einschränkungen verbundene Melting Ice Technologie. Tower-Eiffel bildet direkt auf seinen Zwischencode in C ab. Seine Übersetzungzeiten verhalten sich daher analog zu denen eines C++ Compilers.

4.3.2 Die Arbeitsoberfläche: EiffelBench

Das wichtigste Werkzeug der Umgebung ist die EiffelBench. Es stellt eine für EIFFEL konzipierte Arbeitsoberfläche bereit. Das Werkzeug kombiniert die verschiedenen Übersetzungsmodi und deren aufeinanderfolgende Übersetzungsprozesse. In der Oberfläche intergriert ist die Möglichkeit unterschiedliche Ansichten auf das gesamte System aus allen Klassen, auch als „*Universum*“ bezeichnet, zu generieren. Zusätzlich wird durch Bestandteile eines Debugger eine Fehlersuche ermöglicht. Dem „*Debugger*“ fehlt leider die „*Watchpoint*“-Unterstützung, die bei jeder Änderung an einem Attribut ein Anhalten an der entsprechenden Zeile im Modul durchführt. Aus Sicht des Entwicklers wäre ein Ausbau um weitere Funktionalität angemessen.

4.3.3 Der Oberflächengenerator: EiffelBuild

Das Werkzeug EiffelBuild soll bei der Erstellung von Fenstern und Dialogen sowie deren zugrunde gelegten Zustandsdiagramme samt notwendiger Kontextbetrachtung behilflich sein. Auf eine Unterstützung bei der Erstellung von Zustandsautomaten sollte, da das gesamte Verhalten der Applikation in einem flachen Automaten beschrieben werden muß, verzichtet werden. Hier ist eine Erweiterung zur Hierarchiebildung von Automaten dringend erforderlich. Die Erzeugung der Fenster und Dialoge selbst ist intuitiv und leicht durchführbar. Leider erweist sich der generierte Code als ungeeignet. Für Erweiterungen, die projektspezifisch erforderlich sind, erweist sich der generierte Code als zu statisch. Das gesamte Konzept des Werkzeugs sollte daher komplett überarbeitet werden. Der generierte Code dient im Rahmen des Projektes als autodidaktisches Mittel, um die fehlende Dokumentation über die Bibliothek EiffelVision zu beheben.

4.3.4 Das Entwurfswerkzeug: EiffelCase

Das Werkzeug EiffelCase soll eine Unterstützung im Bereich des Entwurfes bereitstellen. Konzeptionell ist das Werkzeug sauber und intuitiv nachvollziehbar aufgebaut. Leider erweist sich die Implementierung als unglücklich umgesetzt oder zu früh freigegeben. Das Werkzeug unterstützt die „Business Object Notation“ und es werden sowohl die Generierung ausgehend von Entwurf in BON nach der Implementierungssprache EIFFEL, als auch deren Umkehrung, dem sogenannten Redesign angeboten.

Leider fiel die maschinelle Unterstützung durch das Werkzeug EiffelCase aus. Es handelt sich um ein Produkt der Version 1.0. Diese zeichnet sich durch überwiegend instabiles Verhalten aus.

Dies zeigt sich in folgendem Verhalten:

- wiederholte Programmabstürze
- irreparable Zerstörung von Entwurfsdaten
- Die Möglichkeit der Erzeugung von inkonsistenten Systementwürfen
- nicht nachvollziehbares Verhalten beim Redesign.
- Der unterstützte Anteil der BON-Notation ist erweiterungsbedürftig

Das Werkzeug ist zumindest soweit es die aktuelle Version betrifft, als Basis der Entwurfsmethode ungeeignet. Für zukünftige Projekte darf man auf die Weiterentwicklung des Werkzeuges gespannt sein.

4.3.5 Der Editor: EMACS

Als Beispiel eines Editors wurde das „Allzweckwerkzeug“ EMACS, in Bezug auf dessen Möglichkeiten für Projekte mit EIFFEL eingesetzt werden zu können, ausgewählt. Es soll in diesem Abschnitt nur kurz auf den Editor eingegangen werden. Genaue Beschreibungen der vielen Möglichkeiten dieses Editors sind vielfach aufgezeigt worden [CaRo91].

Speziell für die Programmierung in EIFFEL wurde eine öffentlich nutzbare Erweiterung des Modus „eiffel.el“ geschrieben. Mit rudimentären Kenntnissen in LISP lassen sich leicht Makros erstellen und mit Tastenbelegungen koppeln. So läßt sich die Erzeugung oft wiederholter Kontrollflußsequenzen stark vereinheitlichen und beschleunigen. Mit wenigen Überlegungen läßt sich dadurch eine Codierungsrichtlinie erarbeiten und festlegen, die teilweise durch den Editor unterstützt wird. Als hilfreich erweist sich die einheitlichen Dateibeschreibungen und die Erzeugung von abruchsicheren Schleifen.

Fehlgeschlagen ist leider die Bemühung einen Compile-Edit Zyklus herzustellen, bei dem nach versuchter Übersetzung das fehlerhafte Modul geöffnet wird, und der Cursor auf der entsprechenden Position steht, um dem Entwickler Arbeit abzunehmen. Dies erwies sich in einem angemessenen Zeitraum als nicht durchführbar, da der Compiler nicht immer Informationen bezüglich Zeilenindex des aufgetretenen Fehlers lieferte.

Daß eine solche Einbindung möglich ist, wird durch Tower-Eiffel aufgezeigt. Hier dient der EMACS als sogenannte Workbench. Welcher Ansatz den Vorzug erhält, liegt im Ermessen des jeweiligen Entwicklers.

5
Mathematische Grundlagen

Die Mathematik beschäftigt sich unter anderem mit der Erstellung von Methoden zur Analyse von n-dimensionalen Wertemengen. Im Fall der Arbeit handelt es sich um die Analyse diskrete Wertemengen, daher ist die Anwendung von Methoden aus dem Bereich der Statistik naheliegend.

5.1 Statistik

Für die statistische Auswertung und Beurteilung von n-dimensionalen Wertemengen diskreter Punktepaare werden zwei für diese Arbeit wichtige Teilaspekte herausgegriffen. Diese Aspekte dienen der Beurteilung eines hypothetischen Verhaltens und der Aussagekraft dieser Hypothese.

Die beiden Analysemethoden sind:

- Regressionsanalyse (zur Vertiefung siehe Kapitel Anhang A)
  spezifiziert funktionalen Zusammenhang, mit welcher Funktion die diskreten Werte abgeschätzt werden sollen und wie ihre Konstanten zu wählen sind
- Korrelationanalyse (zur Vertiefung siehe Kapitel Anhang B)
  quantitatives Maß über den Zusammenhang zwischen Funktion und diskreten Wertepaaren

5.2 Die Korrelationsanalyse

Die Korrelation liefert ein Maß für den Grad der Abhängigkeiten verschiedener Merkmale. Zu nennen ist hier der Test auf Signifikanz eines Zusammenhangs, das heißt, der linearen Unabhängigkeit, sowie die Vergleiche zwischen Korrelationen verschiedener Merkmalskombinationen.

Grundlegend muß zwischen den betrachteten Merkmalen ein nachvollziehbarer Zusammenhang bestehen, da sonst eine sogenannte Nonsens-Korrelation ermittelt wird. Vorsicht ist auch bei den sogenannten Scheinkorrelationen geboten, wenn zwei nicht korrelierende Merkmale über ein drittes korreliert werden. In diesem Fall sind partielle Koeffizienten anzuwenden; diese sind Bestandteil der multiplen Korrelationsanalyse. Sie sind ein Maß dafür, wie quantitativ gut sich ein Merkmal Y durch Merkmale X_1,..., X_n erklären läßt.

5.2.1 Korrelation zweier normalverteilter Merkmale

Für die Korrelation zweier Zufallsvariablen X und Y gilt:

$$\delta = \text{CORR}(x, y) = \frac{\text{Kovarianz}(x, y)}{\sqrt{\text{Varianz}(x) \cdot \text{Varianz}(y)}} \in (-1, 1)$$

Formel 5-1: Korrelation zweier Merkmale

mit der

$$\text{Kovarianz}(x, y) = \sum_{i=0}^{n} (x_i - \bar{x}) \bullet (y_i - \bar{y})$$

und der

$$\text{Varianz}(x) = \left(\sum_{i=0}^{n} (x_i - \bar{x}) \right)^2$$

wenn die Variablen X und Y unkorreliert sind dann gilt $\delta = 0$.

Als Schätzer bzw. Stichprobenkorrelation wird folgende Berechnung angewendet:

$$r_{XY} = \frac{s_{XY}}{s_X \cdot s_Y} = \frac{\sum_{i=0}^{n} (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\left(\sum_{i=0}^{n} (x_i - \bar{x}) \right)^2 \cdot \left(\sum_{i=0}^{n} (y_i - \bar{y}) \right)^2}}$$

$$= \frac{\sum_{i=0}^{n} (x_i \cdot y_i) - n\bar{x}\bar{y}}{\sqrt{\left(\sum_{i=0}^{n} x_i^2 - n\bar{x}^2 \right) \cdot \left(\sum_{i=0}^{n} y_i^2 - n\bar{y}^2 \right)}}$$

Formel 5-2: Schätzer für zwei Merkmale

Diese wird auch als *Pearson Korrelation* bezeichnet.

Sind $X_1,...,X_n$ normalverteilte Zufallsvariablen gilt:

$$R = \begin{bmatrix} 1 & r_{X_1X_2} & \cdots & r_{X_1X_n} \\ r_{X_2X_1} & 1 & \cdots & r_{X_2X_n} \\ \cdots & \cdots & \cdots & \cdots \\ r_{X_nX_1} & \cdots & r_{X_nX_{n-1}} & 1 \end{bmatrix}$$

Formel 5-3: Korrelationsmatrix nach Pearson für n Dimensionen

Für den Fall von drei Merkmalen gilt also:

$$R = \begin{bmatrix} 1 & r_{XY} & r_{XZ} \\ r_{XY} & 1 & r_{YZ} \\ r_{XZ} & r_{YZ} & 1 \end{bmatrix}$$

Formel 5-4: Korrelationmatrix nach Pearson für drei Dimensionen

5.2.2 Partielle Korrelation

Oft läßt sich eine Korrelation zwischen zwei Meßreihen X,Y nur finden, weil beide Merkmale mit einem dritten Merkmal U korreliert sind. Die Korrelation zwischen X,Y ist dann eine reine Scheinkorrelation. Daher ist es oftmals von Interesse, Korrelationen zwischen X und Y, unter Partialisierung eines Merkmales U, das heißt, die Korrelation zwischen X und Y, die ohne Einfluß von U vorhanden ist, zu bestimmen. Eine solche Korrelation $\delta_{(X,Y)/U}$ wird auch als partielle Korrelation von X,Y unter U bezeichnet.

$$\delta_{(X,Y)/U} = \frac{\delta_{XY} - \delta_{XU}\delta_{YU}}{\sqrt{(1-\delta_{XU}^2)\cdot(1-\delta_{YU}^2)}}$$

Formel 5-5: Partielle Korrelation

Partielle Korrelation mit normalverteilten Merkmalen

Als Schätzer kann dann der *Pearson Korrelationskoeffizent* verwendet werden.

$$r_{(X,Y)/U} = \frac{r_{XY} - r_{XU}r_{YU}}{\sqrt{(1-r_{XU}^2)\cdot(1-r_{YU}^2)}}$$

Formel 5-6: Koeffizienten der partiellen Korrelation nach Pearson

5.2.3 Bi-partielle Korrelation

Das heißt, ist ein Merkmal X mit U und Y mit V korreliert und es besteht eine Korrelation zwischen U und V, so wird die Korrelation zwischen X,Y von U,V stark beeinflußt. Will man anhand einer Wertemenge die Korrelation zwischen X,Y schätzen, so emp-

fiehlt es sich, die Merkmale X von V und Y von U zu partialisieren. Diese Korrelation heißt bi-partielle Korrelation von X unter Partielisierung von U und Y unter Partialisierung von V.

$$\delta_{X/U,\,Y/V} = \frac{\delta_{XY} - \delta_{XU}\delta_{YV} - \delta_{XV}\delta_{YU} + \delta_{XU}\delta_{UV}\delta_{YV}}{\sqrt{(1-\delta_{XU}^2)\,(1-\delta_{YV}^2)}}$$

Formel 5-7: Pearson-Koeffizienten bei bi-partieller Korrelation

Bei Normalverteilung gilt:

$$\tau_{X/U,\,Y/V} = \frac{\tau_{XY} - \tau_{XU}\tau_{YV} - \tau_{XV}\tau_{YU} + \tau_{XU}\tau_{UV}\tau_{YV}}{\sqrt{(1-\tau_{XU}^2)\,(1-\tau_{YV}^2)}}$$

Formel 5-8: Kendall-Koeffizenten bei bi-partieller Korrellation

5.2.4 Multiple Korrelation

Die multiple Korrelation ist ein Maß für die Abhängigkeit eines Merkmals Y von p anderen Merkmalen $X_1,..,X_p$, dabei wird vorausgesetzt, daß die Merkmale in ihrer Gesamtheit normalverteilt sind. Sie ist definiert als die betragsmäßig größte Korrelation unter den Korrelationen zwischen dem Merkmal Y und allen möglichen Linearkombinationen $\sum_{i=1}^{p} a_i X_i$ der Merkmale X_i mit der Gewichtung a_i.

Will man die multiple Korrelation $\delta_{Y,\,(X_1,...,X_p)}$ abschätzen, so schätzt man zunächst alle möglichen einfachen Korrelationen δ_{YX_i} und $\delta_{X_iX_j}$ mittels der Pearsonkoeffizenten ab.

Für $p = 1$ ergibt sich so $r_{Y,\,(X_1)} = \left|r_{Y,\,(X_1)}\right|$

und für $p = 2$ ergibt sich:

$$r_{Y,\,(X_1,\,X_2)} = \sqrt{\frac{r_{YX_1}^2 + r_{YX_2}^2 - 2r_{YX_1}r_{YX_2}r_{X_1X_2}}{1 - r_{X_1X_2}^2}}$$

Formel 5-9: Pearson-Koeffizenten bei multipler Korrelation mit Dimensionsgrad 2

Allgemein gilt: $r_{Y,\,(X_1,..,X_p)} = \sqrt{r_{YX}^T R_{XX}^{-1} r_{YX}}$ siehe dazu Formel 5-10

$$r_{Y,\,(X_1,...,X_p)} = \left[\left[r_{YX_1}, ..., r_{YX_p} \right] \begin{bmatrix} 1 & r_{X_1X_2} & ... & r_{X_1X_p} \\ r_{X_2X_1} & 1 & ... & r_{X_2X_p} \\ ... & ... & ... & ... \\ r_{X_pX_1} & ... & r_{X_pX_{p-1}} & 1 \end{bmatrix}^{-1} \begin{bmatrix} r_{YX_1} \\ r_{YX_2} \\ ... \\ r_{YX_p} \end{bmatrix} \right]^{\frac{1}{2}}$$

Formel 5-10: Pearson-Koeffizenten bei multipler Korrelation mit Dimensionsgrad n

Die Größe $B = r_{Y,\,(X_1,...,X_p)}$ wird das *Bestimmungsmaß* der multiplen Regression von Y genannt. Es ist ein Maß dafür, wie gut das Merkmal Y sich durch die Merkmale $X_1,...,X_p$ erklären läßt. B gibt den Anteil der Varianz wieder, der durch die anderen p Merkmale erklärt werden kann. Es gilt, die multiple Korrelation ist genau dann gleich 0, wenn alle einfachen Korrelationen gleich 0 sind.

5.3 Die Regressionsanalyse

Die Regression versucht einen funktionalen Zusammenhang zu liefern. Die Ziele dabei sind unter anderem:

- Nachweis von bekannten Beziehungen
- Schätzen von Parametern einer bekannten funktionalen Beziehung
- Erkennen von funktionalen Zusammenhängen
- Repräsentation großer Datenmengen
- Interpolation fehlender bzw. Prognose zukünftiger Werte

Interessiert nur der Zusammenhang zweier Merkmale X und Y mit Ausprägungsvariablen x und y, Regressor und Regressand genannt, so wird eine Beziehung $y = f(x)$ spezifiziert. Dies wird als Regression von Y auf X bezeichnet. Mittels einer Stichprobe wird dann der funktionale Zusammenhang geschätzt. Die einfachste, funktionale Abhängigkeit ist linear, von daher wird sie als lineare Regression bezeichnet. In der Residuenanalyse wird eine Überprüfung des Ansatzes vorgestellt. Viele Zusammenhänge lassen sich aber besser durch nichtlineare Regression modellieren. Zu nennen sind die polynominale und multiple Regression.

5.3.1 Lineare Regression

Vermutet man einen zumindest näherungsweisen, linearen Zusammenhang zwischen zwei Merkmalen in ihrer Gesamtheit, so kann mittels linerar Regression dieser Zusammenhang näher spezifiziert und untersucht werden. Es wird in diesem Fall davon ausgegangen, daß dann gilt: $y_i = \alpha + \beta \cdot x + e_i$ für $i = 1,...,n$ mit Verschiebung α, Steigung β und dem zufälligen Fehler e_i. Nachfolgend werden Punkt-, Intervallabschätzung und Tests für die Parameter α und β des Regressionsproblemes angegeben. Der dabei entstehende zufällige Fehler e_i wird genauer analysiert.

Methode der kleinsten Quadrate

Die Punktabschätzung von a und b für die Parameter α und β eines linearen Regressionsproblemes wollen wir so bestimmen, daß durch die Grade $\hat{y} = \alpha + \beta \cdot x$ eine möglichst gute Schätzung für die Ausprägung y geliefert wird. Als Kriterium der Güte dieser Abschätzung ist die Summe der Abweichungsquadrate (Residualquadrate) ein geeignetes Maß.

$$S^2 = \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 = \sum_{i=1}^{n} \left(y_i - (a + b \cdot x_i) \right)^2$$

Formel 5-11: Methode der kleinsten Quadrate

Somit ergeben sich a und b als Lösung des Normalengleichungssystem

$$\frac{\partial S}{\partial a} = -2 \sum_{i=1}^{n} \left(y_i - (a + b \cdot x_i) \right) = 0$$

$$\frac{\partial S}{\partial b} = -2 \sum_{i=1}^{n} x_i \cdot \left(y_i - (a + b \cdot x_i) \right) = 0$$

Formel 5-12: Normalengleichung

das heißt die Schätzer a und b sind:

$$b = \frac{\sum_{i=1}^{n} (x_i - \bar{x})\ (y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \quad \text{und} \quad a = \bar{y} - b \cdot \bar{x}$$

Formel 5-13: Die kleinsten Quadratschätzer

Eine nach der Methode der kleinsten Quadrate geschätzte Regressionsgerade schneidet stets den Punkt $(\bar{x}, \bar{y})$ und bei fehlerfreier Messung sind die Fehlerterme $e_1,...,e_n$ unabhängig mit dem Erwartungswert 0 und der Varianz σ^2.

Schätzen der Fehlervarianz

Da die Varianzen σ_a^2 und σ_b^2 der kleinsten Quadratschätzer von der Varianz σ^2 der zufälligen Fehler $e_1,...,e_n$ abhängen, möchte man σ^2 erwartungstreu abschätzen. Schätze die theoretischen Residuen mit $e_i = y_i - (\alpha + \beta x_i)$ durch $\hat{e}_i = y_i - (a + bx_i)$ und nehme für σ^2 die Schätzung, die unter allen erwartungstreuen Schätzern die kleinste Varianz besitzt.

$$s^2 = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n-2} = \frac{\sum_{i=1}^{n} (\hat{e}_i)^2}{n-2} = \frac{S^2}{n-2}$$

Formel 5-14: Schätzer für Varianzverhalten

Zusammenhang zwischen Regression und Korrelation

Der Pearson Koeffizent r_{XY} zweier Merkmale steht im engem Zusammenhang zum dem Schätzer b des Steigungsparameters β. Es gilt:

$$b = r_{XY} \cdot \frac{s_{\hat{Y}}}{s_Y}$$

Formel 5-15: Zusammenhang Regression und Korrelation

Aus der Korrelation r_{XY} kann also der Steigungsparameter β der linearen Regression von Y nach X geschätzt werden und umgekehrt. Als Maß für die Güte der Anpassung, die eine Regression erzielt, dient das *Bestimmtheitsmaß* der Regression,

$$B_{X,Y} = r_{XY} \cdot \frac{s_{\hat{Y}}}{s_Y}$$

Formel 5-16: Bestimmtheitsmaß der Regression

also das Verhältnis der Varianzen der geschätzten Werte $\hat{y}_i$ und den beobachteten Werten, genauer der Anteil an der Varianz Y, der durch das Merkmal X erklärt werden kann. Es gilt stets: $0 < B_{X,Y} < 1$ eine optimale Anpassung ist erreicht, wenn gilt $B_{X,Y} = 1$, denn dann wird Y durch die Regression vollständig erklärt.

Die Größe $U_{X,Y} = 1 - B_{X,Y}$ nennt man das *Unbestimmtheitsmaß* einer Regression. Speziell bei der linearen Regression ist das Bestimmtheitsmaß identisch mit dem Quadrat der Korrelationen von X und Y, das heißt es gilt: $B_{X,Y} = r_{XY}^2$.

5.3.2 Transformationen

Es kann bei einem anderen als dem linearen Verhalten eine Transformation der Werte sinnvoll sein. Bei der Transformation ändern sich die Parameterschätzungen durch die Methode der kleinsten Quadrate in gleicher Weise, wie sich die Parameter selbst ändern.

Funktion	transformierte Funktion
$y = \alpha x^{\beta}$	$\ln(y) = \ln(\alpha) + \beta \cdot \ln(x)$
$y = \alpha e^{\beta x}$	$\ln(y) = \ln(\alpha) + \beta \cdot x$
$y = (\alpha + \beta x)^{-1}$	$\frac{1}{y} = \alpha + \beta x$
$y = \frac{x}{\alpha + \beta x}$	$\frac{1}{y} = \alpha + \frac{\beta}{x}$
$y = \alpha e^{\beta / x}$	$\ln(y) = \ln(\alpha) + \frac{\beta}{x}$
$y = \frac{1}{\alpha + \beta e^{x}}$	$\frac{1}{y} = \alpha + \beta e^{x}$
$y = \alpha x^{\beta} e^{\gamma x}$	$\ln(y) = \ln(\alpha) + \beta \ln(x) + \gamma x$

Formel 5-17: Transformationstabelle

Es kann in Situationen, in denen eine einfache Regression nicht angebracht ist, die Methode der multiplen Regression anwendet werden.

5.3.3 Multiple Regression

Betrachte die k+1 Merkmale $Y, X_1, ..., X_k$ und versuche einen Ansatz:

$$y(x_1,...,x_k) = \beta_0 + \beta_1 x_1 + ... + \beta_k x_k = \sum_{i=0}^{k} \beta_i x_i$$

Formel 5-18: Multiple Regression

So nennt man diese Funktion eine multiple Regressionsgleichung von Regressand Y und Regressoren $X_1, ..., X_k$.

Der Ansatz über ein Normalengleichungssystem für Wertepaare $(y_i, x_{1i}, ..., x_{ki})$ liefert

$$\begin{bmatrix} b_1 SQ_{X_1, X_1} + b_2 SQ_{X_1, X_2} + ... + b_k SQ_{X_1, X_k} \\ b_1 SQ_{X_2, X_1} + b_2 SQ_{X_2, X_2} + ... + b_k SQ_{X_2, X_k} \\ ... \\ b_1 SQ_{X_k, X_1} + b_2 SQ_{X_k, X_2} + ... + b_k SQ_{X_k, X_k} \end{bmatrix} = \begin{bmatrix} SQ_{X_1, Y} \\ SQ_{X_2, Y} \\ ... \\ SQ_{X_k, Y} \end{bmatrix}$$

Formel 5-19: Normalengleichung der multiplen Regression

die kleinsten Quadratschätzer $b_1,..,b_k$ sind. Dabei ist

$$SQ_{U,V} = \sum_{i=1}^{k} (u_i - \bar{u})(v_i - \bar{v})$$

Formel 5-20: Elemente der Normalengleichung

und für das Absolutglied gilt: $a = \bar{y} - (b_1 \bar{x}_1 + ... + b_k \bar{x}_k)$. Setzt man

$$\tilde{Y} = \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_n \end{bmatrix}, \; X = \begin{bmatrix} 1 & x_{11} & .. & x_{k1} \\ 1 & x_{12} & .. & x_{k2} \\ ... & ... & .. & .. \\ 1 & x_{1n} & .. & x_{kn} \end{bmatrix}, \; e = \begin{bmatrix} e_1 \\ e_2 \\ ... \\ e_n \end{bmatrix}, \; \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ ... \\ \beta_k \end{bmatrix}$$

Formel 5-21: Matrizenansatz für multiple Regression

so läßt sich die allgemeine Modellgleichung in Matrixdarstellung schreiben, als $\tilde{Y} = X\beta + e$ mit dem Fehlervektor e. Für diesen gilt: $E(e_i) = 0$ und $Var(e_i) = \sigma^2$

Der Schätzvektor b für β ergibt sich somit aus den Normalengleichungen als:

$$X^T X b = X^T \tilde{Y}$$

Formel 5-22: Gleichung der multiplen Regression

und falls $X^T X$ invertierbar ist ergibt sich ein eindeutiger Schätzer für β.

$$b = (X^T X)^{-1} X^T \tilde{Y}$$

Formel 5-23: Schätzvektor der multiplen Regression

6
Grober Entwurf

In diesem Kapitel wird der Entwurf der Ausarbeitung grob skizziert. Die daran anschließenden Kapitel 7,8 und 9 verfeinern sukzzessive, die hier aufgeführten Komponenten. Diese Vorgehensweise entspricht in der Konzeptionsphase dem Zusammenfassen der gewünschten Funktionalität in disjunkten Teilaufgaben. Jede der Aufgaben wird in der Analysephase auf einen Klassenverband abgebildet und mehrere Verbände dieser Klassen bilden so jeweils eine Bibliothek.

6.1 Gliederung des Entwurfes

Die Gliederung ist hierarchisch aufgebaut, so wird eine einfache, leicht nachvollziehbare Struktur erreicht und die Aufgabe in einzelne, erweiterbare Bestandteile aus Klassenverbänden zerlegt. Diese Aufbau erleichtert zukünftige Modifikationen wie das Erweitern, Ändern oder Ersetzen von Teilen der Bibliotheken. Die Trennung in mehrere Bibliotheken ist sinnvoll um eine Verwendung für zukünftige Anwendungen zu garantieren, ohne das diese auf den gesamten Umfang der Bibliothek zurückgreifen müssen. Die Visualisierung und Beurteilung von n-dimensionalen Wertemengen bestehend aus Wertepaaren $(Y,X_1,...,X_{n-1})$ gliedert sich in folgende Aufgabengebiete:

- Verwaltung von Wertemengen
- Interpretation und Analyse
- Visualisierung

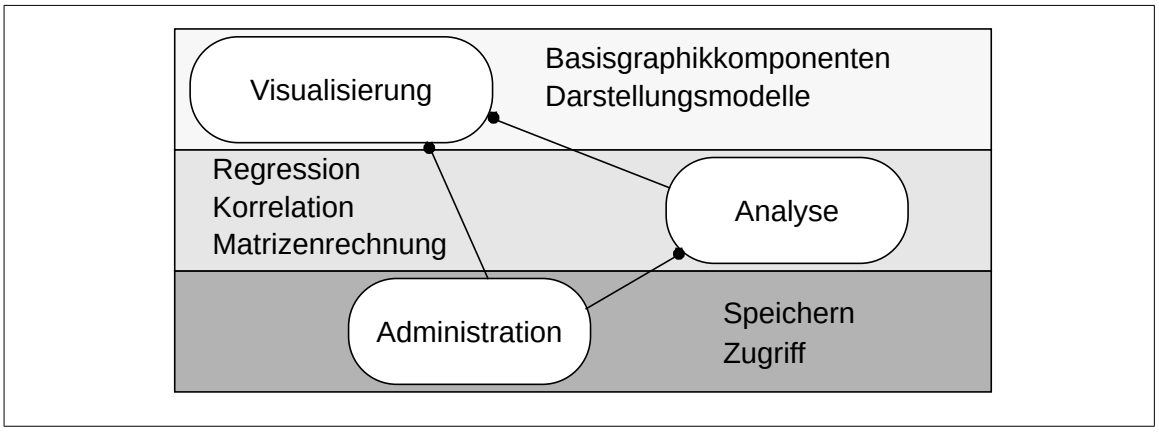

Abbildung 6-1: Grobgliederung des Entwurfes in drei Aufgabenbereiche

6.1.1 Verwaltung und Bearbeitung von Wertemengen

Der Ausgangspunkt für die Erstellung einer Bibliothek zur Bearbeitung von n-dimensionalen Wertemengen ist die Bereitstellung von Methoden, die eine Verwaltung einer solchen Menge ermöglicht. Bei den Wertemengen ist zu beachten, daß die Häufigkeit der Wertepaare auf Verlangen mitberücksichtigt wird.

Die Wertepaare werden im folgenden Kontext auch als Tupel bezeichnet. Das Hauptaugenmerk liegt hierbei auf der Struktur zur einfachen Verwaltung von Tupeln. Diese muß so ausgelegt sein, daß auch eine Bearbeitung auf einem Teilintervall der Ausgangsmenge leicht durchgeführt werden kann.

Die Aufgabe gliedert sich in folgende Bereiche:

- Die Modellierung von Tupeln und n-dimesionalen Wertemengen:
    - Die Wertemengen werden durch zwei Realisierungen abgebildet. Es hat sich im Laufe der Arbeit gezeigt, daß die Forderung der Berücksichtigung der Häufigkeit und der schnellen Bearbeitung über Teilintervalle der Menge unvereinbar in Bezug auf einer optimalen Verwaltungstruktur sind. Da sich die Aufgaben aber strikt trennen lassen, wurde entschieden, beide Strukturen anzubieten.
    - Eine Struktur ist optimiert in Lösch-, Schreib- und Einfügemethoden.
    - Die andere Struktur ist kompakter und geht von unveränderlichen Wertemengen aus. Diese Repräsentation dient in der Variante mit reduziertem Dimensionsgrad, als Ausgangspunkt der Visualisierung.
    - Transformationen zwischen den unterschiedlichen Darstellungsformen.
    - Projektionsverfahren von n-dimensionellen Wertepaaren auf darstellbare Wertepaare, mit maximal drei Dimensionen.
- Beispiele der Anwendung durch Einbindung in Dialogstrukturen. Diese umfassen Projektionen sowie Verwaltungsstrukturen zur Steuerung der Ein- und Ausgabe über Textdaten oder den Verträglichkeitstest der extern gespeicherten textuellen Repräsentation.
    - Diese Erweiterungen sind aber nur temporärer Natur; naheliegender ist hier ein Ansatz mit persistenten Objekten. Dieser Ansatz übersteigt aber den Umfang der Arbeit.

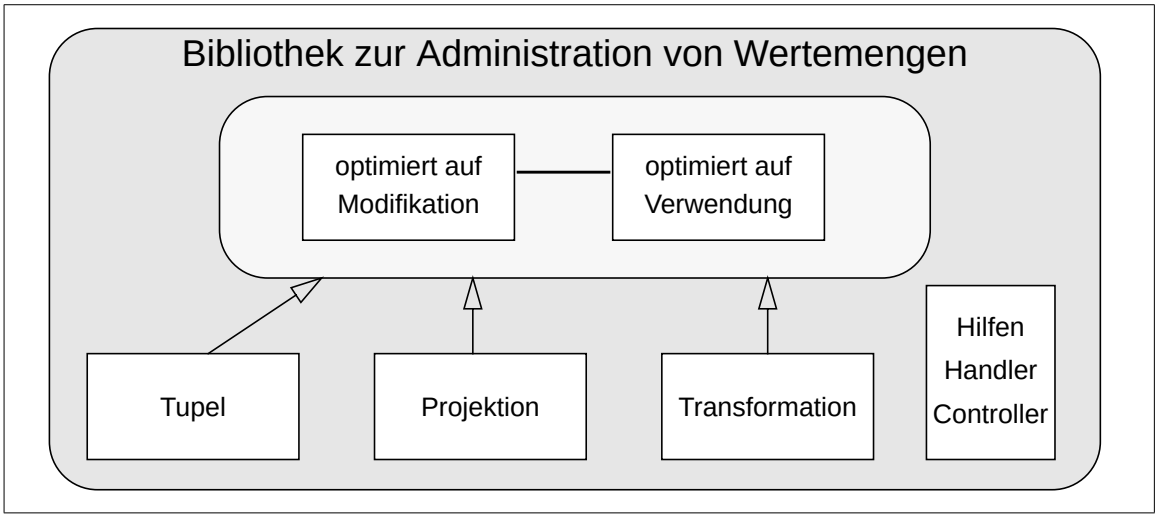

Abbildung 6-2: Administration von n-dimensionalen Wertemengen

6.1.2 Analyse und Interpretation von Wertemengen

Für die Beurteilung und Analyse von Wertemengen werden Operatoren und Strukturen über diese Menge benötigt.

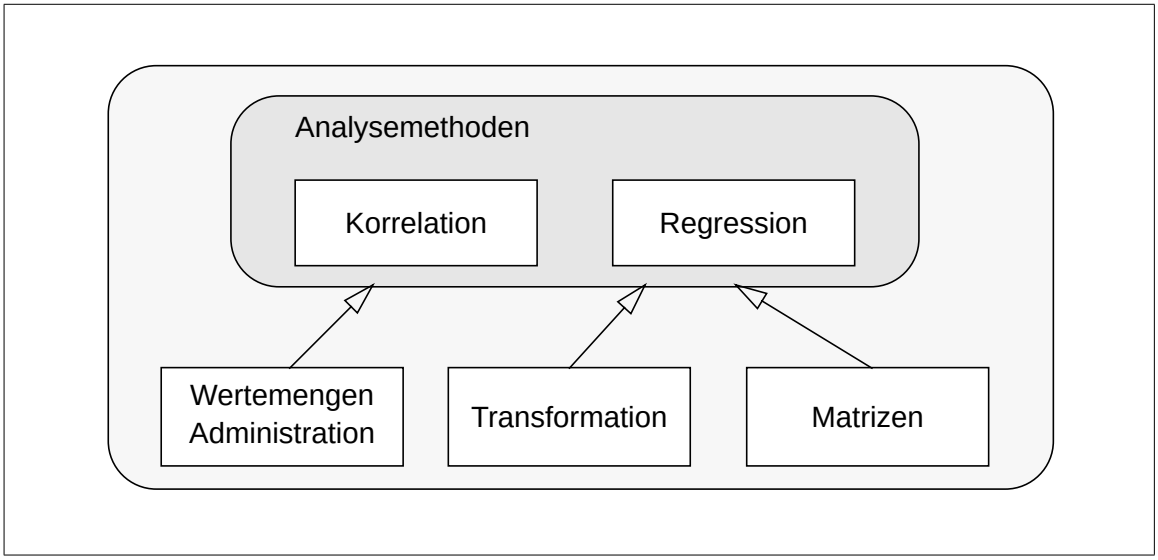

Abbildung 6-3: Analyse der Wertemengen

Im Zusammenhang mit dieser Arbeit wird die Analyse auf die Regression- und Korrelationsanalyse aus dem Bereich der mathematischen Statistik beschränkt. Die bereitgestellten Methoden sollen auch Analysen über Teilintervalle durchführen können.

Die Analyse läßt sich gliedern in:

- Operatoren und Strukturen der Regressions- und Korrelationsanalyse
- mathematische Basisstrukturen
    - Zu nennen sind die Repräsentation für Matrizen und verwendete Operatoren.
- Operatoren für die Transformation
    - Kompliziertere Funktionen werden auf lineare abgebildet und nach erfolgter Berechnung rücktransformiert.

6.1.3 Visualisierung von Wertemengen

Der dritte Aufgabebereich befaßte sich mit der Visualisierung von Wertemengen, er stellt somit Methoden und Mechanismen für diese Aufgabe bereit. Die Visualisierung baut auf den beiden anderen Teilaufgaben auf und stellt einen Baukasten für die Darstellung der Mengen zur Verfügung. Zusätzlich werden noch Verbände eingegliedert, die für unterstützende Aufgaben benötigt werden.

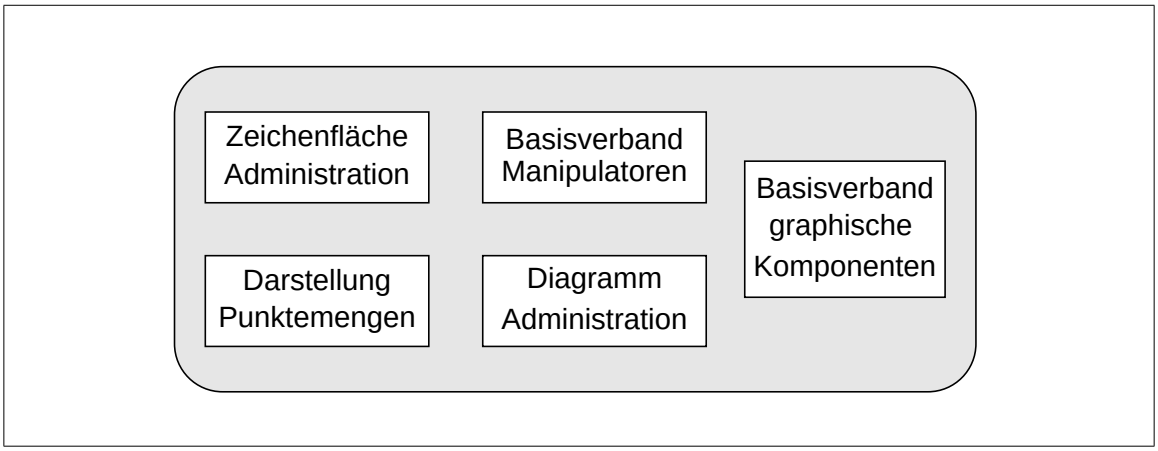

Abbildung 6-4: Visualisierung von Wertemengen

Im Einzelnen sind die zur Visualisierung benötigten Bestandteile:

- Basisbestandteile der Visualisierung
    - Legenden, Texte, Grafikprimitive wie Marker, Frames, Achsen, Raster
- Verwaltungsstrukturen für die Administration von Elementen
    - Berücksichtigung der überdeckten Fläche eines Objektes und seine Zeichenreihenfolge oder Anordnung in der „Tiefe“ (Z-Ordnung)
- geschachtelte Strukturen, die Aufgaben im visuellen Bereich unterstützen und zulassen

- Visualisierungsmodelle für Wertemengen, das heißt welche Interpretation und Auswertung wird verwendet, und wie werden die Wertemengen dargestellt. Welche Wertepunkte werden für ein Darstellungselement berücksichtigt und wie werden die darzustellenden Objekte aus der Menge erzeugt.

  Beispiele für Arten von Interpretationen sind:

  - Punktewolke; wieviele Punkte der Wertepaare sind in einem Darstellungspunkt zusammengefaßt und wie ist diese Häufung in der Darstellung zu berücksichtigen.
  - Streudiagramme; welches Intervall des Wertebereiches soll zu einem Darstellungsobjekt zusammengefaßt werden, dies impliziert ein Intervall im Darstellungsbereich der Zeichenfläche für die Menge.
  - Regressionskurve; welche Regressionsfunktion soll in welchem Intervall der Wertebereiches verwendet werden.

- Bilden von Wertemengen aus darstellbaren Punkten und Berechnung der Regression und Korrelation über diesen Wertemengen
- Beispiele um aus der Klassenbibliothek andere Diagrammarten aufzubauen
- Anbieten von einheitlicher Modellierung für die nachträgliche Erzeugung von Manipulatoren zur interaktiven Einstellung von Darstellungsattributen oder Beurteilungen von Basisbestandteilen der Diagramme

Viele Bestandteile sind einzeln aufgeführt, weil sich im Laufe der Implementierung herausstellte, das ganze Bereiche wie Basiskomponenten oder geschachtelte Verwaltungsstrukturen in den Bibliotheken der Entwicklungsumgebung noch nicht integriert sind.

Zusätzlich lassen sich bestehende Basiskomponenten der Bibliothek EiffelVision nicht angemessen in die zugrundegelegte Modellierung eingliedern, da die Basiskomponenten auf rein visuelle Darstellung und nicht auf geeignete Modellierung ausgerichtet sind.

Auf die Bildung erweiterter Diagrammdarstellungen wird in Kapitel 10 eingegangen. Die „Schnittstelle“ für zukünftige Manipulatoren soll dazu dienen nachträglich Aspekte wie Farbe der Punktemenge, Interpretationsart, sowie Sichtbarkeitsbereich interaktiv verändern zu können. Die Art der Manipulation unterscheidet sich in zwei grundsätzlichen Handhabungsweisen. Dabei ändert die Manipulation das zugrundegelegte Modell oder dessen visuelle Darstellung.

6.2 Entwurfsmuster

Dem Entwurf zugrunde gelegt werden eine Anzahl von verbreiteten Entwurfsmustern. Ein Entwurfsmuster ist ein einfaches, wiederkehrendes Schema, das als Leitfaden für Konzeption, Entwurf und Implementierung herangezogen werden kann. Viele Konzepte und Modelle in der Softwareentwicklung lassen sich auf Basisschemata, im weiteren Kontext als Entwurfsmuster bezeichnet, zurückführen.

Um einige verwendete Entwurfsmuster (“Pattern”) zu nennen :

- Observer - Pattern
- Composite - Pattern
- Chain of Responsibility - Pattern
- Iterator - Pattern

Es handelt sich um allgemein gebräuchliche Entwurfsmuster, diese werden kurz skizziert. Eine weitergehende Beschreibung und zusätzliche Muster finden sich in der Literatur [Gam95].

6.2.1 Observer Pattern

Das Observer-Pattern ist ein Entwurfsmuster, bei dem n Ausprägungen von Objekten von einem anderen abhängig sind und auf dessen Änderungen reagieren.

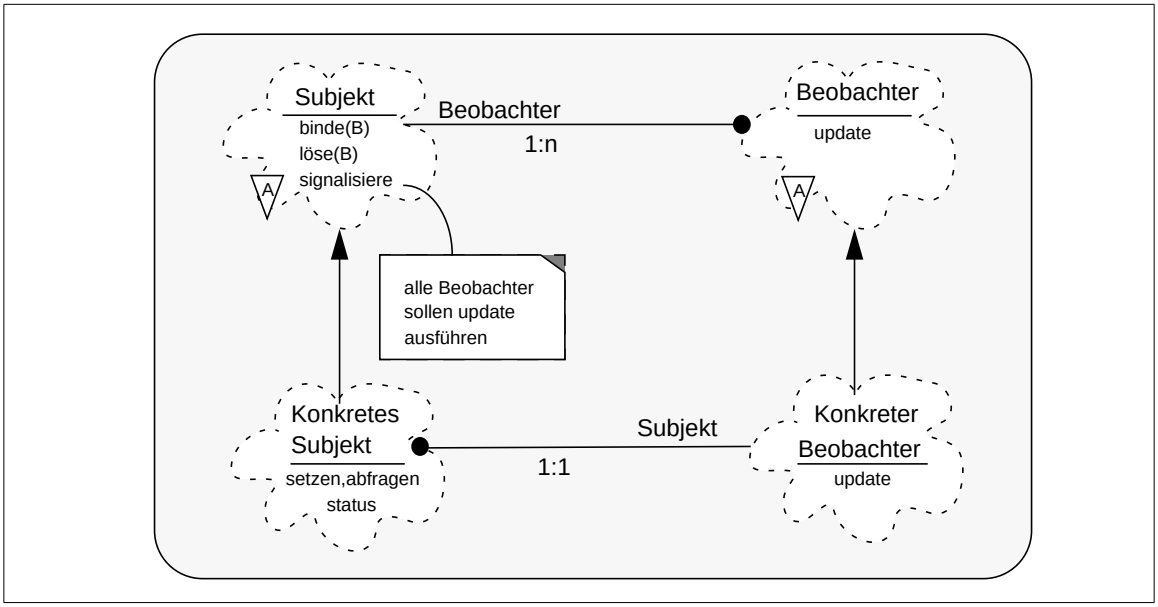

Abbildung 6-5: Observer Pattern

Werden Änderungen an dem beobachteten Objekt durchgeführt, so werden alle abhängigen Beobachter informiert. Das Entwurfsmuster basiert in seiner Kommunikationsstruktur auf dem „*Change - Update*“ Protokoll. Abhängige Objekte müssen wissen, ob und wie sie auf eine Änderungen des beobachteten Objektes zu reagieren haben. Der Ansatz bildet die Basis des in Entwicklungsumgebungen basierend auf Smalltalk [GoRo89] verwendeten Ansatzes nach dem „*Model-View-Controller*“ Prinzip.

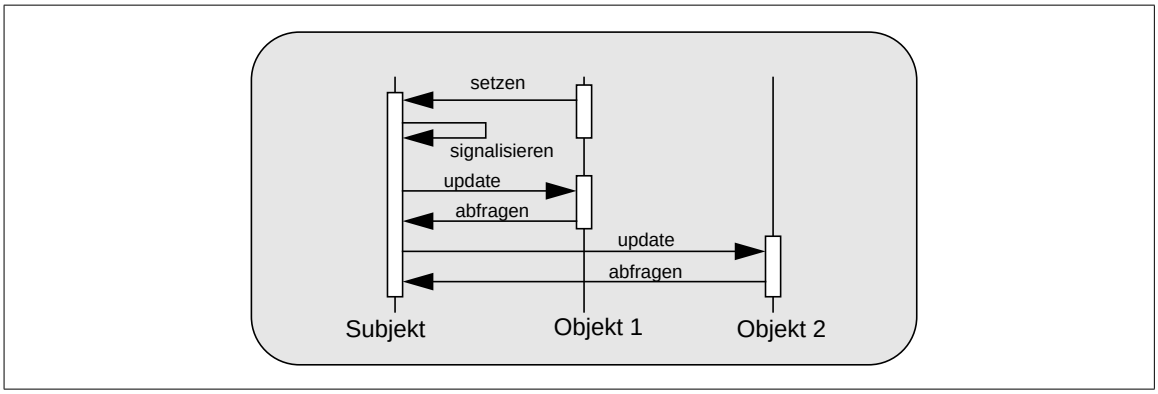

Abbildung 6-6: Interaktionsdiagramm des Observer Patterns

In Kombination mit anderen Entwurfsmustern kann es sinnvoll sein, die Kommunikation in anderer, leicht abgewandelter Form durchzuführen. Eine Beschreibung, wodurch sich Änderungen in der Nachrichtenstruktur bei Kombination ergeben können, findet sich in Kapitel 6.2.5.

Model-View-Controller

Der *Modell-View-Controller*, im folgenden mit MVC bezeichnet, ist eine Spezialisierung des allgemeinen Observer Pattern.

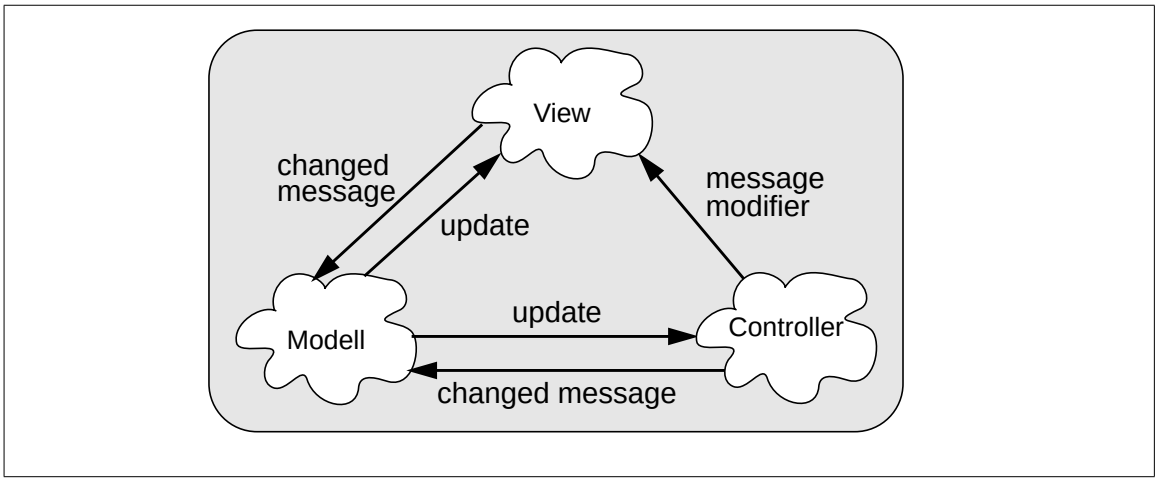

Abbildung 6-7: Der Model-View-Controller

Die beobachtenden Objekte werden im MVC mit den visuellen Repräsentation gleichgesetzt und die Modelle legen den Ausgangspunkt der Darstellung fest. Werden Änderungen am Modell durchgeführt, müssen diese Änderungen den Darstellungen durch eine Nachricht oder Methodenaufrufe bekannt gegeben werden.

Die Darstellungsarten müssen die Änderungswünsche berücksichtigen und sich neu zeichnen. Der Controller nimmt externe Ereignisse der Benutzer oder verwendeter Bereiche entgegen und steuert die visuellen Darstellungen und das Modell.

6.2.2 Composite Pattern

Das Schema, das für das Entwurfsmuster *Composite* zugrunde gelegt wird, ist eine Hierarchie aus Basiskomponenten. Die Basiskomponenten können beliebiger Gestalt sein. Eine Basiskomponente repräsentiert die Komponentegruppe (*„Composite"*).

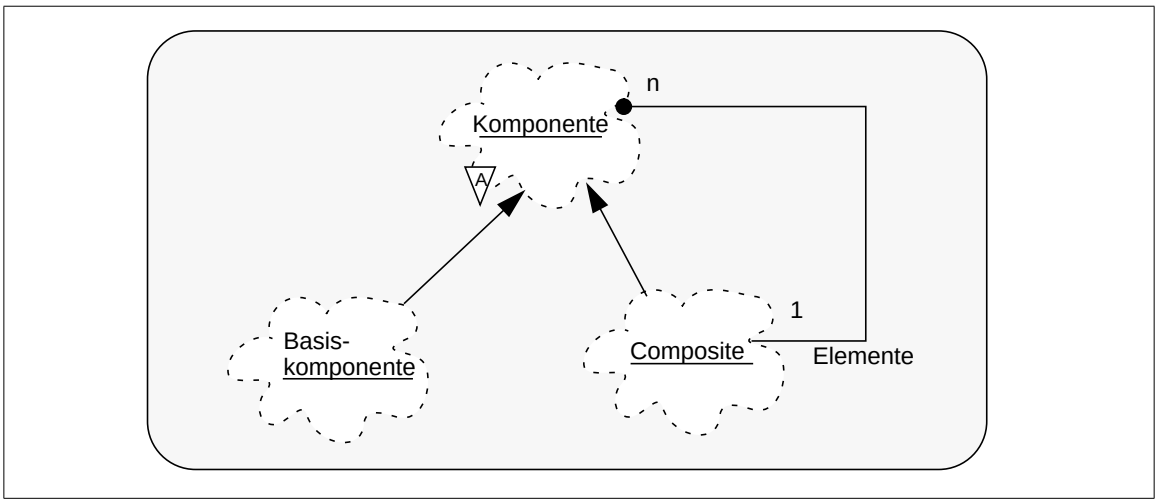

Abbildung 6-8: Das Composite Pattern

Die Gruppe kann eine unbestimmte Anzahl von Komponenten enthalten, so daß auch Schachtelungen aus Gruppen möglich sind. Wenn zusätzlich gefordert wird, daß keine Basiskomponente in zwei unterschiedlichen Gruppen als direktes Mitglied geführt werden darf, ergibt sich eine strenge Hierarchie an deren Wurzel die Applikation stehen kann.

6.2.3 Chain of Responsibility Pattern

Das Muster „Chain of Responsibility" ist ein Schema, das häufig im Bereich administrativer Hierarchien seine Anwendung findet. Falls Verwaltungstrukturen dieser Form verwendet werden, sind Methoden, die sich auf administrative Belange beziehen, oft aus übergeordneten Strukturen ähnlicher Bauart anzufordern.

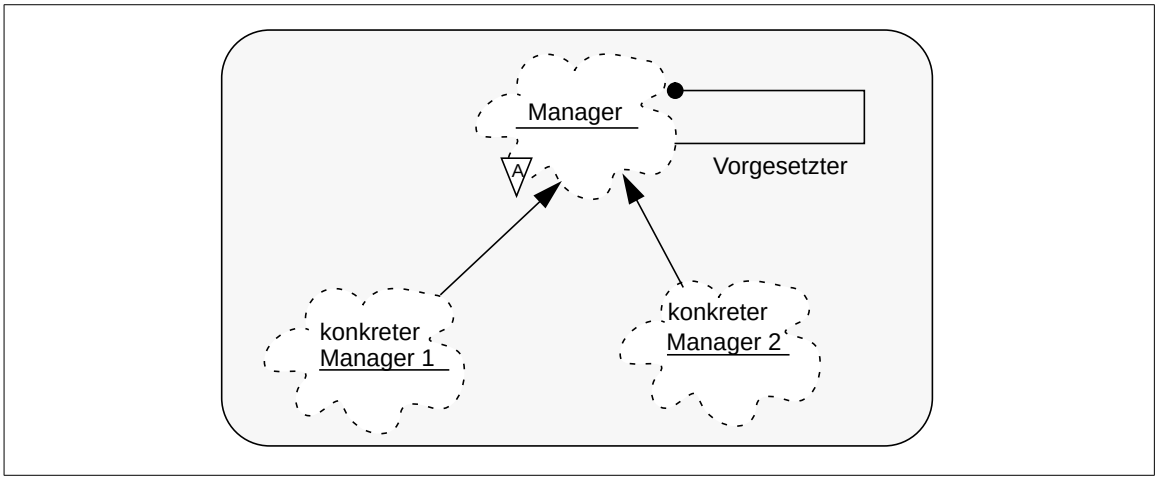

Abbildung 6-9: Das Chain of Responsibility Pattern

Dieses Verhalten läßt sich vergleichen mit den unterschiedlichen Führungsebenen in einem Betrieb. Übersteigt eine Entscheidung die Kompetenz der jeweiligen Ebene, so wird die Anfrage an die nächsthöhere Instanz weitergeleitet. Spätestens an der höchsten Hierarchiestufe muß dann eine Entscheidung erzwungen werden.

Bei Hierarchien mit vielen Schachtelungsebenen kann es erforderlich sein, das Entscheidungenskompetenzen an untergeordnete Ebenen weitergeleitet werden, um die Rekursionstiefe geringer zu halten.

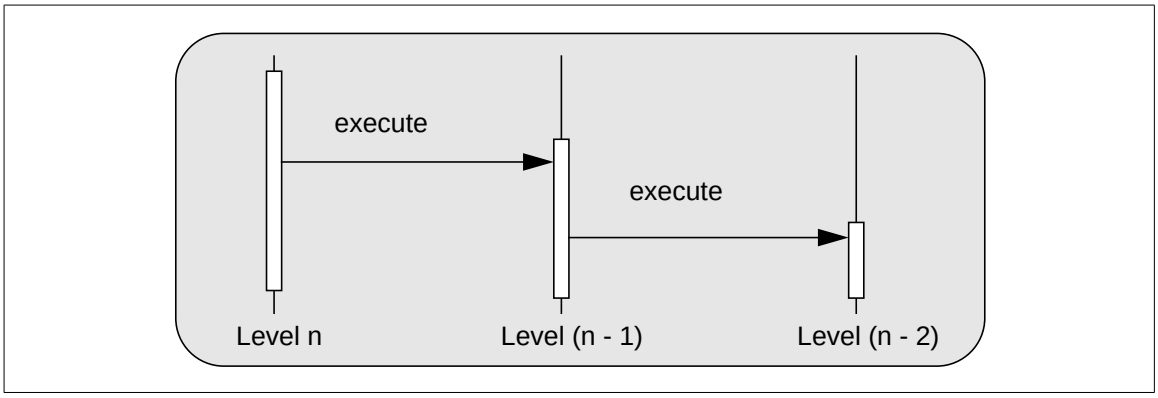

Abbildung 6-10: Interaktionsdiagramm der Chain of Responsibility

Das verwendende Objekt „*leiht*" sich die Methode aus übergeordneten Strukturen aus und soll sich nicht darum kümmern, wer die Methode bearbeitet. Die Ausführung wird solange nach oben gereicht, bis eine Struktur die angeforderte Ausführung durchführt.

6.2.4 Iterator Pattern

Das Muster ermöglicht einen sequenziellen Zugriff auf Elemente einer Ansammlung von Objekten, ohne konkrete Beschreibung der darunterliegenden realen Struktur der Menge. Die Struktur muß nur sequenziell abgearbeitet werden können. Verwendung findet dieses Muster zum Beispiel bei Summenoperatoren über Wertemengen.

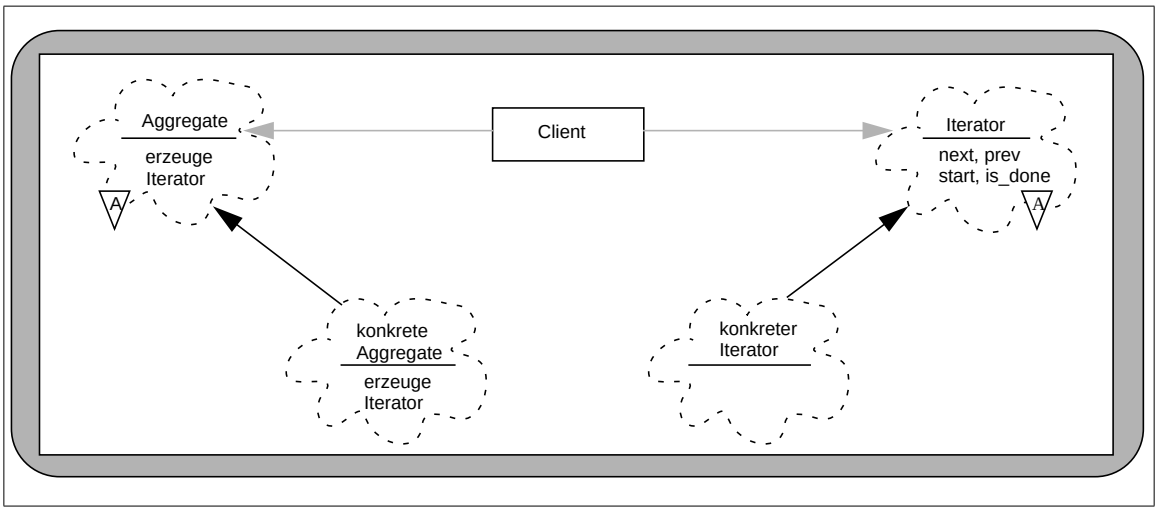

Abbildung 6-11: Iterator Pattern

6.2.5 Kombinationen aus Entwurfsmustern

Bei der Kombination von Entwurfsmustern, wie zum Beispiel bei Kombination der Muster Observer, Composite kann es erforderlich sein, die Muster in geänderter Form einzusetzen. Im Falle der genannten Muster ergeben sich in Folge der Hierarchie Mehrdeutigkeiten bei der Nachrichtenstruktur des Observer Patterns.

Bei einer Hierarchie von Observer-Mustern stellt sich folgendes Problem ein: Wird seitens der beobachteten Strukturen an mehreren hierarchisch, untergeordneten Strukturen eine Änderung durchgeführt, besteht die Möglichkeit, daß jede betroffene Struktur direkt ihre abhängigen Beobachter unterrichtet und die Beobachter die Veränderung in Berücksichtigung ziehen.

Hieraus kann sich das Problem ergeben, daß die Reihenfolge der Änderungen auf Seiten der Modellhierarchie nicht der Reihenfolge entspricht, die in der Ansichtenhierarchie sinnvollerweise durchzuführen ist. Vielmehr kann es in einem solchen Fall richtig sein, erst alle Änderungen durchzuführen und dann von einer übergeordneten, beobachteten Ausprägung, die erforderliche „*Update*"-Nachricht an dessen abhängige Beobachter zu verschicken. Diese übergeordneten Beobachter wiederum propagieren die stattgefundende Änderung innerhalb ihrer Hierarchie. Dabei läßt sich entweder eine Propagierung nach Art der Tiefen- oder Breitensuche durchführen.

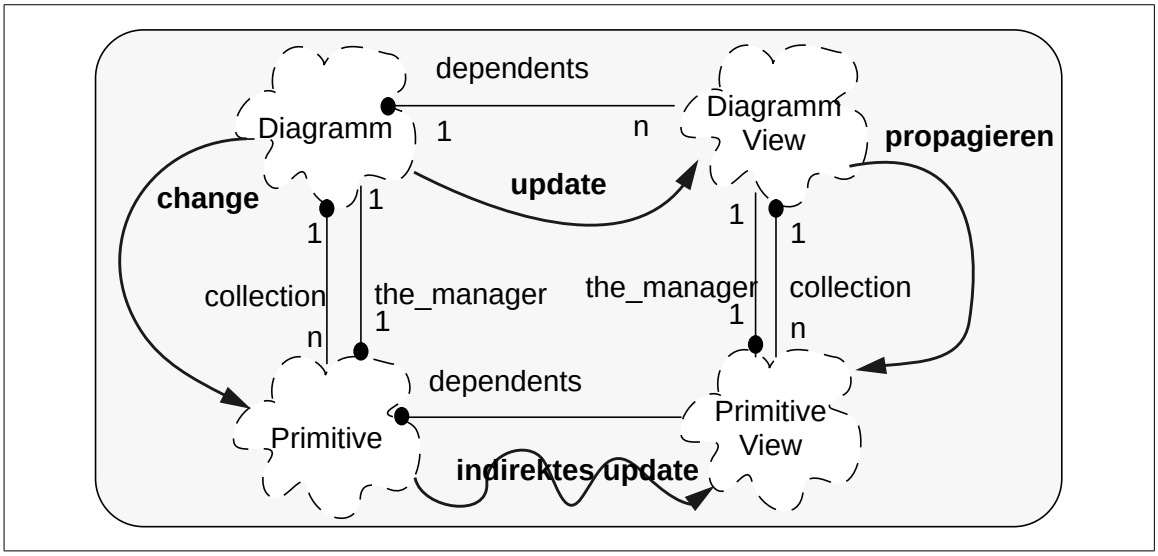

Abbildung 6-12: Kombination von Entwurfsmustern

Noch schwieriger wird es wenn die Reihenfolge der „*Update*“-Nachrichten an die Beobachter abhängig von anderen Mustern ist, wie es zum Beispiel bei einander überdekkenden Objekten der Fall ist. Hier muß die Z-Ordnung berücksichtigt werden.

6.3 Einbindungen ins Application Framework

Die Einbindung der Diagrammklasse in das „Application Framework“ [Grh96][Grh96] läßt sich am besten graphisch erklären.

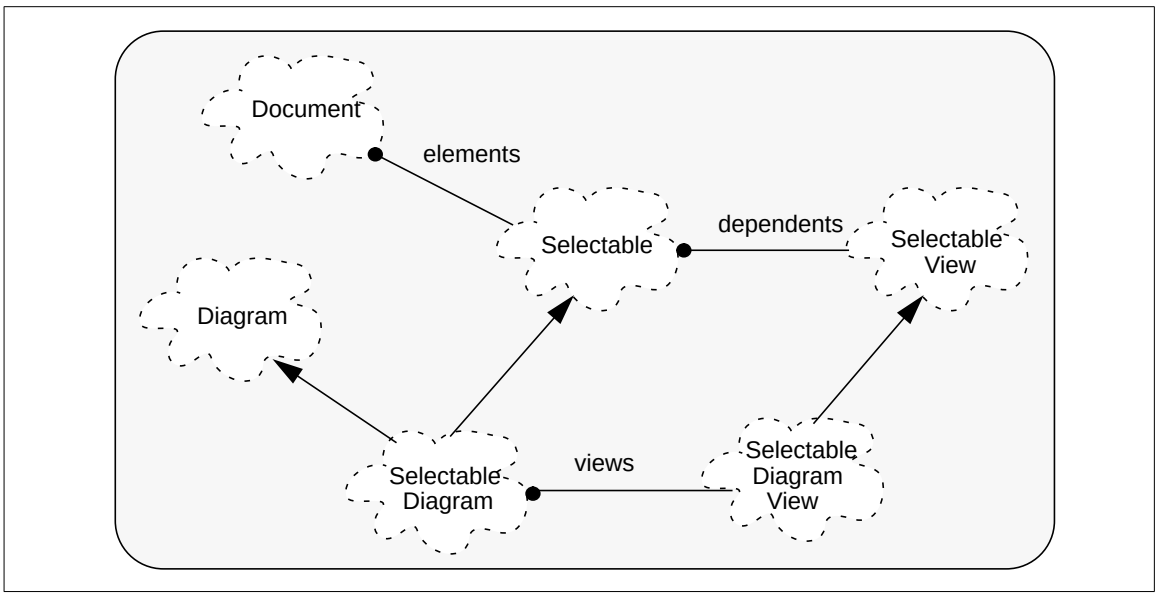

Abbildung 6-13: Einbindung in bestehendes Applications Framework

Sehr vereinfacht besteht der Entwurf aus einem Modell-View-Controller Ansatz bei dem die Applikation mehrere Dokumente verwalten kann. Jedes Dokument wiederum kann mehrere Dokumentansichten, sogenannte „*Documentviews*“ bereitstellen. Innerhalb eines Dokumentes sind die Grundelemente als selektierbare Komponenten festgelegt. Jede Basiskomponente erbt somit die Eigenschaft selektierbar zu sein. Es wird die Möglichkeit der Bildung von geschachtelten Gruppen aus Basiskomponenten angeboten, sogenannte Composites.

Die selektierbaren Diagramme werden als Basiskomponenten in das Konzept mit eingebunden, dadurch lassen sie sich verschieben oder in der Größe und ihrem Seitenverhältnis variieren. Von einer Einbindung als Gruppe von Basiskomponenten wurde abgesehen, da eine Zerlegung des Diagrammes als Basiskomponente des Dokumentes weder vorgesehen noch sinnvoll ist. Ein Diagramm aus Sicht des Dokumentes ist eine unteilbare Basiskomponente, sinnvoller ist hier der Ansatz des gemeinschaftlichen erbens aus einer Klasse *Graphikelemente.*

Es bestehen noch weitere Abhängigkeiten nähmlich zur Klasse Applikation. Hier werden alle applikationsweit, verfügbaren Strukturen und deren Verwalter eingeglieder, außerdem wird die Menüstruktur um die Funktionalität Ein- und Ausgabemechanismen erweitert. Im folgenden Kontext werden diese als „*Controller*“ und „*IO-Handler*“ bezeichnet.

6.4 Gliederung in Klassenverbände

Die Gliederung in Klassenverbänden basiert auf der Gliederung im Entwurf. Es werden aber zusätzlich noch weitere Verbände mit eingebunden, diese zeichnen sich in erster Linie durch ihren unterstützenden Charakter aus. Optional ist auch die Erweiterung bezüglich Dialogunterstützung vorhanden.

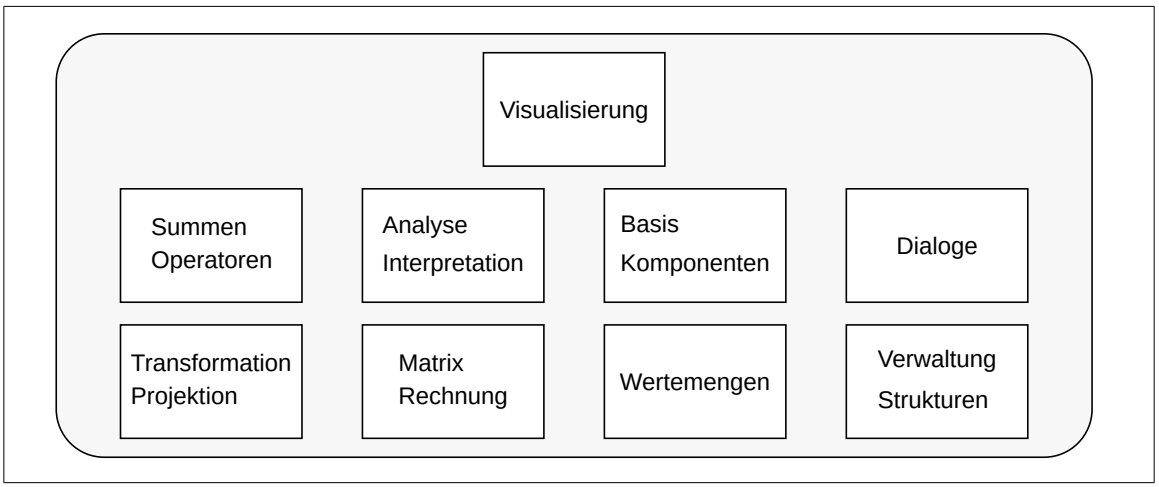

Abbildung 6-14: Gliederung der Klassenverbände

Die Klassenverbände sind:

- Verband zur Repäsentation von Wertemengen
- Verband zur Auswertung von Wertemengen
- Verband zur Visualisierung von Wertemengen
- Verband für Dialoge
- Verbände für unterstützende Strukturen, wie visuelle Basiskomponenten

6.4.1 Verband zur Repäsentation von Wertemengen

Der Verband zur Repräsentation von n - dimensionalen Wertemengen gliedert sich in die folgenden Bereiche, eine genauere Beschreibung erfolgt in Kapitel 7.

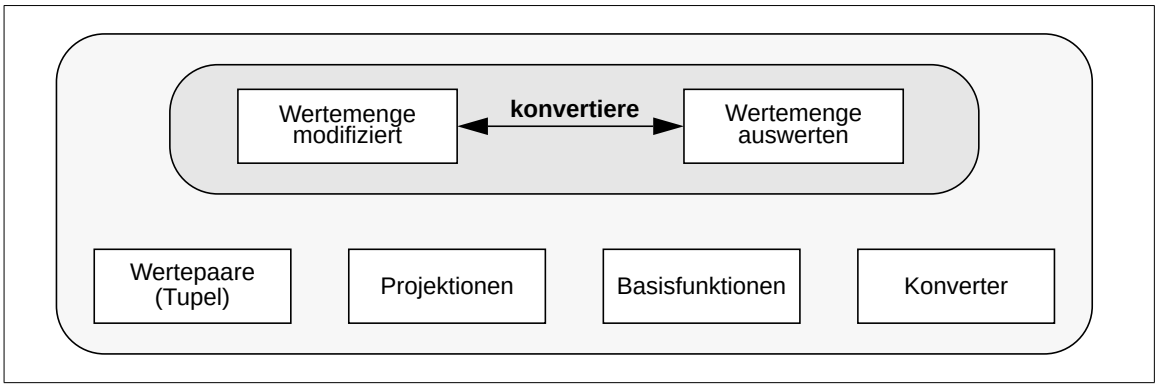

Abbildung 6-15: Verwaltung von Wertemengen

- Repräsentation der mehrwertigen Tupel als Basiskomponenten der Wertemengen
- Komponenten zur Verwaltung von Wertemengen, optimiert im Bezug auf Modifikation
    - Die Repräsentation wird bei externen Eingabedaten von unsortierten Wertemengen verwendet um Duplikate der Tupelausprägungen zu finden und dabei die Mehrwertigkeit der Tupel zu berücksichtigen.
- Komponenten zur Verwaltung von Wertemengen, die hauptsächlich der Auswertung dienen und an denen keine Änderung mehr vorgenommen werden sollen.
- Konvertierungen zwischen den beiden Darstellungen.
- Projektionen
    - Reduktion des Dimensionsgrades zwecks Darstellbarkeit
- Basisfunktionen für die Projektion, diese kommen aus einem mathematischen Hilfsverband.

6.4.2 Verband zur Auswertung von Wertemengen

Bei der Auswertung und Interpretation von Wertemengen werden Repräsentationen aus der Mathematik im Bereich Lineare Algebra, wie Matrizenrechnung und Iteratoren unter dem Gesichtspunkt der Summenbildung verwendet, eine Vertiefung erfolgt in Kapitel 8.

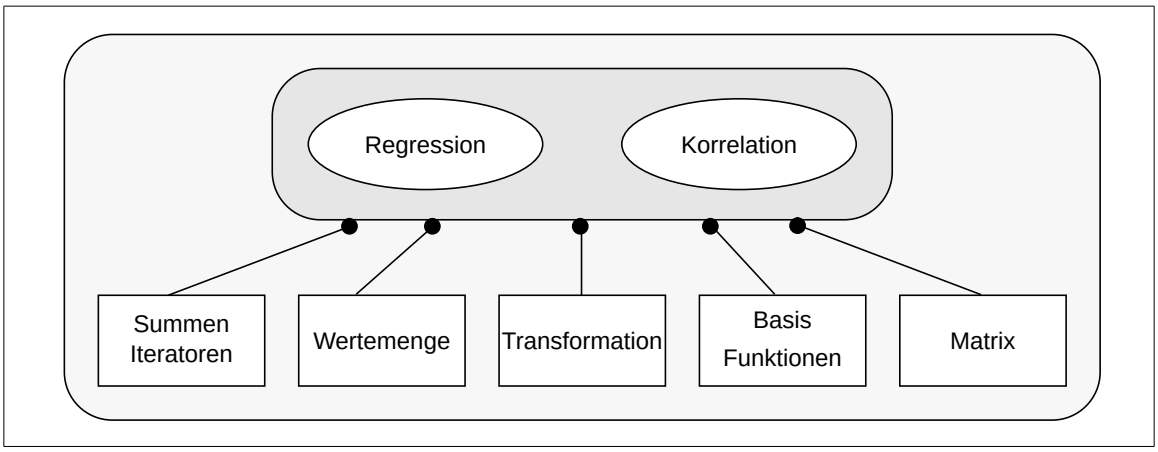

Abbildung 6-16: Analyse und Bewertung von Wertemengen

Die einzelnen Bestandteile sind von ihren Aufgaben her:

- Operatoren der Regressions- und Korrelationsrechnung
  - Verwendung eines Iterators auf einem Intervall der Wertemenge um mathematische Basisoperationen, wie endliche Reihensummen zu berechnen.
- Zur Berechnung der obigen Operatoren werden mathematische Grundstrukturen aus dem Bereich der Matrizenrechnung notwendig
  - Die Matrizenrechnung geht von einer n-dimesionalen Matrix, auf der nur Basisoperationen erlaubt sind, aus.
  - Zur Optimierung werden alle Bereiche, die nur 2-dimensional sind, eigenständig implementiert.
  - Quadratische Matrizen sind ein Sonderfall, diese werden aber durch Zusicherung und nicht in Form weiterer Spezialisierung erhalten.
  - Transformationen, Basisfunktionen und Summeniteratoren
- Der Ausgangspunkt der Berechnung sind die Wertemengen optimiert auf Auswertung

6.4.3 Verband zur Visualisierung von Wertemengen

Die Visualisierung basiert auf dem Verständnis von hierarchischen Zuständigkeiten, genauere Beschreibungen diesbezüglich erfolgen in Kapitel 9. Der Verband zur Visualisierung gliedert sich in folgende Bestandteile, dabei soll die Aufzählung keine Verbindung zu den Hierarchiestufen darstellen.

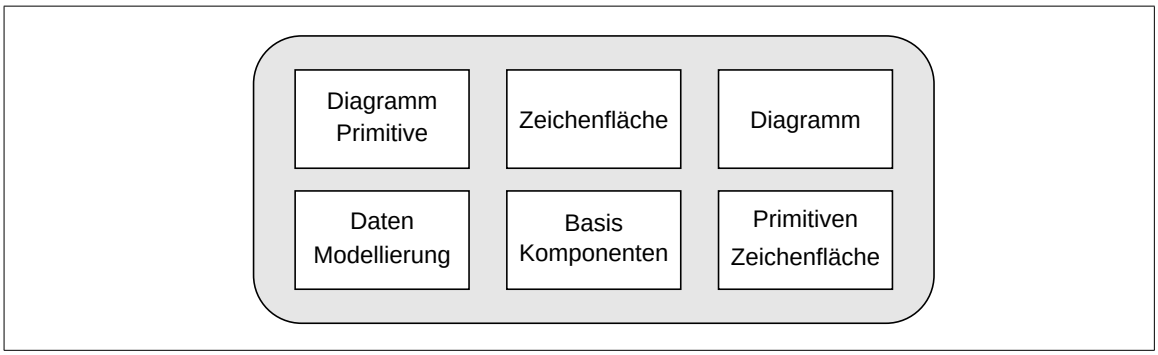

Abbildung 6-17: Visualisierung von Wertemengen

- Ein Diagramm ist eine rechtwinklige Darstellungsfläche, die als Container für die Basiskomponenten des Diagrammes dient.
- Basiskomponenten von Diagrammen sind: Texte, Primitive, graphische Flächen, Zeichenflächen für Diagramme und Gruppen aus Basiskomponenten wie zum Beispiel die Legenden bestehend aus Basissymbolen und Text.
- Eine Zeichenfläche wiederum ist ein Container für Basiskomponenten der Zeichenfläche. Diese ähneln den Basiskomponenten der Diagramme. Zusätzlich gibt es Raster, Achsen, Rahmenbeschriftungen und Darstellungsmodelle für darstellbare Wertemengen.
- Darstellungsmodelle sind Festlegungen wie die entsprechende Wertemenge zu interpretieren sind, zum Beispiel ob die Wertemenge in Form eines Streubalken interpretiert werden soll.

6.4.4 Verband für Dialoge

Die Hilfsklassen zur Dialoggestaltung werden für das Einlesen und das Schreiben von Wertemengen verwendet. Die Dialogführung in der Kopplung von Diagrammdaten durch freidefinierte Projektionen erwies sich als zu aufwendig. Hierbei sind sehr viele Einstellungsmöglichkeiten vorhanden und die Definition der Projektion erfordert die Berücksichtigung von n-dimensionalen Wertemengen und deren Dimensionsbeschreibungen. Da diese nur unterstützende Verbände darstellen, soll nicht weiter auf diese eingegangen werden.

6.4.5 Verbände für unterstützende Strukturen

Hierunter werden in erster Linie solche Strukturen zusammengefaßt, die ihre Funktionalität den anderen Verbänden in unterstützender Tätigkeit bereitstellen. Weiterhin sind hier die Primitivklassen eingebunden, die für mögliche Inspektoren einen festen Angriffspunkt bereithalten.

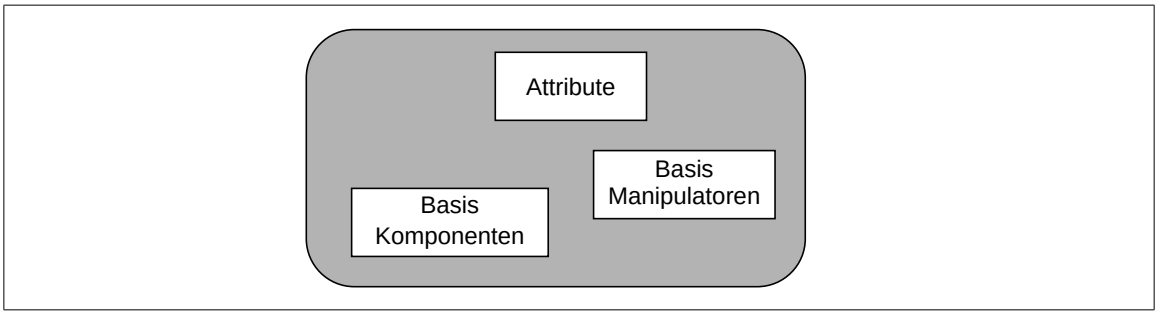

Abbildung 6-18: Gemeinsam genützte Klassenverbände

7
Die Verwaltung von Wertemengen

In diesem Kapitel werden die Strukturen und die Mechanismen beschrieben, die für die Verwaltung von n-dimensionalen Wertemengen erforderlich sind. Es wird die hierarchische Struktur der Verwaltung erläutert und auf interessante Teilaspekte eingegangen. Der Verbund aus Klassen zur Verwaltung der Mengen wird so ausgelegt, daß dieser nicht nur im Fall von Zahlenwerten sondern prinzipiell auf andere Werte angewendet werden kann.

Die Verwaltung muß die absolute Häufigkeit der einzelnen Wertepaare (X_1,...,X_n,Y) berücksichtigen. Die Berücksichtigung der Häufigkeit erfordert ein schnelles Auffinden von Wertepaaren, von daher wird die Arbeit auf sortierbare Tupel beschränkt, das heißt, auf solche über die eine Ordnung definiert ist. Somit wird von den Tupeln verlangt, daß über diesen ein Vergleichsoperator definiert ist, der wiederum die Vergleichbarkeit der einzelnen Attribute eines Tupels fordert.

Durch die möglichst offengehaltene Konzeption ist der Entwurf in Klassenstrukturen natürlich feinkörniger geworden. Eine Trennung, die durch die Restriktionen der generischen Attribute auferlegt wird, hat wiederholt zur Aufspaltung in getrennte Klassen geführt. Zum Beispiel verlangen arithmetische Tupel definierte numerische Methoden diese werden auf Methoden der Tupelattribute zurückgeführt.

Der Vorteil dieser „feinkörnigen“ Einteilung in Klassen liegt in der Wiederverwendung geschriebenen Codes zum Beispiel bei einer Anbindung an eine relationale Datenbank.

7.1 Analyse der Verwaltungsstruktur

Die Wertemengen setzen sich aus einer beliebigen Anzahl von Tupeln gleichen Dimensionsgrades, das heißt gleicher Anzahl von Tupelattributen zusammen. Die Menge entspricht von daher der Vorstellung des Begriffes „*Tabelle*“ in der Sprache SQL bei relationalen Datenbanken.

Wegen der zu berücksichtigenden Häufigkeit der Tupel müssen diese Darstellungen mit Zählern ausgestattet sein. Es dürfen aus diesem Grunde auch keine Duplikate zu eindeutigen Tupel innerhalb einer Wertemenge existieren, da ansonsten die Konsistenz der Menge nicht garantiert ist.

Die Wertemengen unterscheiden deswegen zwischen der Total- und Maximalanzahl von Tupeln. Die Totalanzahl von Tupelausprägungen zählt alle Tupel samt Duplikate, die in der Menge gespeichert sind, wohingegen die Maximalanzahl einer Menge die Anzahl unterschiedlicher Tupel angibt.

Die Struktur der Wertemengen muß so ausgelegt sein, daß die Suche nach Tupeln schnell durchzuführen ist. Zusätzlich muß für die Bearbeitung und Analyse in der Regressions- und Korrelationsanalyse eine effektive Verwendung von Summeniteratoren zur Berechnung von endlichen Reihen vorhanden sein.

7.1.1 Dualismus der Anforderungen

Die Anforderung der Verhinderung von Duplikaten legt eine baumartige Struktur nahe, wogegen für die Verwendung der Iteratoren die Verwendung von Feldern naheliegender ist. Keiner der beiden Strukturen konnte im Rahmen der Arbeit ein eindeutiger Vorzug gegeben werden. Die Defizite der einen Repräsentation bilden die Stärke der anderen und umgekehrt. Daher werden beide Repräsentationen im Rahmen der entwickelten Klassenbibliothek bereitgestellt.

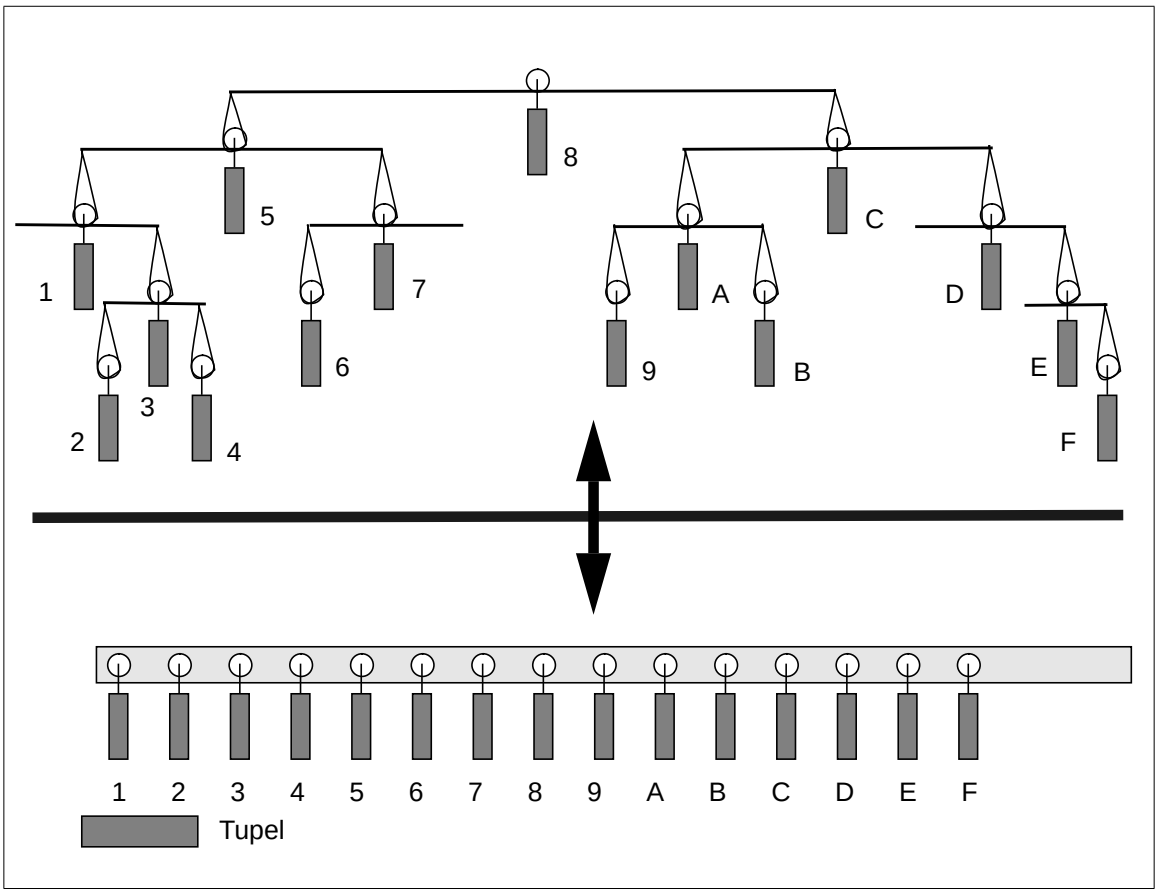

Abbildung 7-1: Konkurrierende Darstellungen der Wertemengen

7.1.2 Projektion

Die Visualisierung der Wertemengen erfordert, daß die n-dimensionalen Wertepaare der Menge auf unter drei Dimensionen einer neuen Wertemengen projeziert werden. Die erforderlichen Projektionen und Basismethoden werden zur Verfügung gestellt.

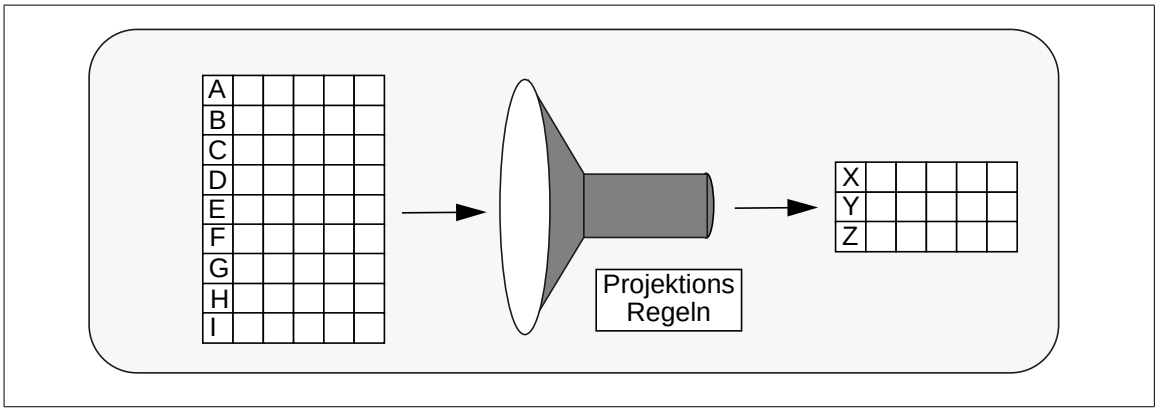

Abbildung 7-2: Projektion von Wertemengen

7.1.3 Konvertierung

Die beiden unterschiedlichen Repräsentationen von Wertemengen erfordern zusätzlich die Erstellung eines Konverters zur Transformation zwischen den Darstellungen. Die Umwandlung wird in beiden Richtungen bereitgestellt, zusätzlich wird noch die Möglichkeit einer vorherigen Intervallrestriktion angeboten.

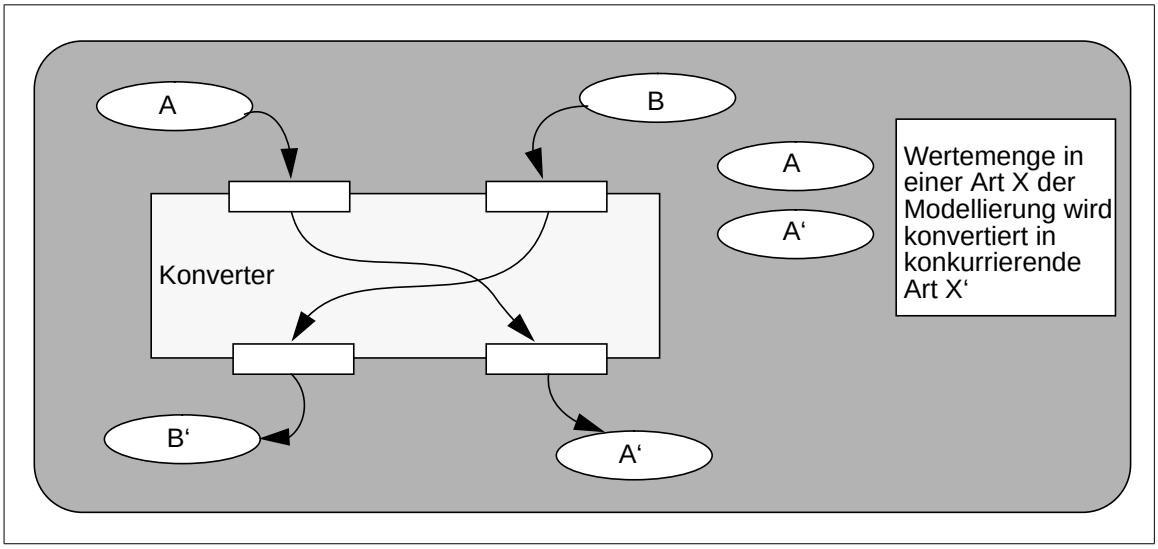

Abbildung 7-3: Konvertierung von Wertemengen

7.1.4 Aufbau der Verwaltung von Wertemengen

Der Verband zur Verwaltung gliedert sich in die folgenden Teilverbände:

- Diagram_Data, Wertemenge optimiert auf Auswertung
- Valueset, Wertemenge optimiert auf Modifikation
- Konverter
- Projektion
- Basisfunktionen
- Tupel
- Iterationsoperatoren, bestehend aus Iteratoren die auf einzelnen Tupeln arbeiten aber iterativ über der Wertemenge operieren.
- IO_Handler und Controller

Diese sind untereinander folgendermaßen gekoppelt:

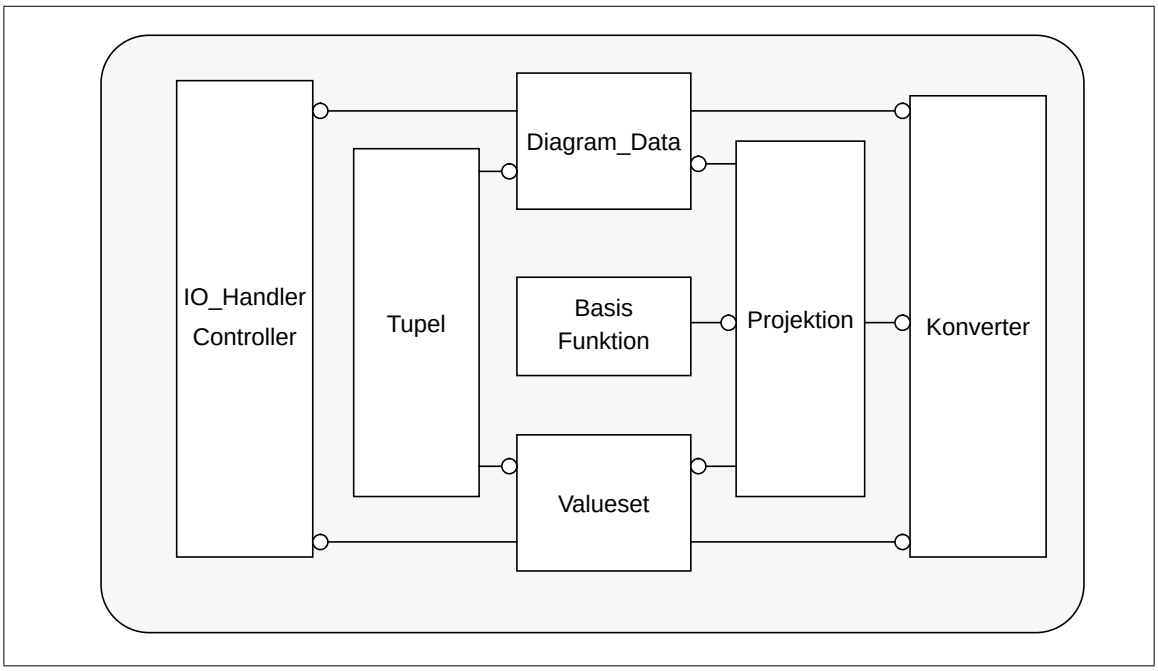

Abbildung 7-4: Aufbau des Verbandes

7.2 Die Repräsentation der Wertemenge

Die Abbildung der n-dimensionalen Wertemenge teilt sich in den zwei getrennte Aufgabenbereiche: der Darstellung optimiert auf Modifikation von Wertemengen und der Darstellung, die für die Auswertung ohne Veränderung der Menge zu verwenden ist.

Die Wertemengen setzen sich aus Mengen von eindeutigen Tupeln zusammen, wobei die Darstellung der Tupel in der Repräsentation, die optimiert auf Modifikationen ist explizit und in der anderen Abbildungsvorschrift nur noch implizit abgebildet werden. Zur optimierten Verwaltung werden die Tupel sortiert in beide Repräsentationen eingebunden, um so ein Auffinden bereits vorhandener Tupel zu erleichtern.

7.2.1 Darstellung der Wertemengen optimiert auf Modifikation

Die Repräsentation der Menge, die auf die Modifikation und Erzeugung der Menge optimiert ausgelegt ist, wird bevorzugt bei der Erzeugung ausgehend von unsortierten externen Wertemengen aus Tupeln gleichen Dimensionsgrades verwendet. Die Stärken dieser Darstellung liegen in der Erweiterung der Wertemengen um weitere Tupel oder in der Modifikation entsprechender Tupeleinträge in Bezug auf deren Häufigkeitszähler.

Naheliegenderweise wurde hierfür die Struktur eines Binärbaumes verwendet. Die geforderte Sortierung impliziert eine Ordnung über den Tupeln. Die Knoten des Baumes verweisen auf die angebundenen Tupel. Der Baum wird auf die Struktur eines binären Suchbaumes abgebildet. Leider konnte der Suchbaum der Entwicklungsumgebung nicht verwendet werden. Hierfür gab es zwei Gründe, zum einen sollen für die Verwaltung Tupel mit den gleichen Attributwerten und unterschiedlichem Zählerwert als gleich erkannt werden. Der zweite Grund war, daß die zugrundegelegte Struktur des Suchbaunes der Entwicklungsumgebung wegen der umfangreichen Vererbungshierarchie nicht nachvollzogen werden konnte.

Der binäre Suchbaum ist außer mit den Methoden von Suchen, Finden und Test auf Vorhandensein auch mit weiteren Funktionalitäten, wie der Optimierung durch Balancierung des Baumes, ausgestattet. Diese Erweiterung erweist sich bei der Erzeugung der Wertemengen aus unsortierten mit redundanten Tupeln gefüllten externen Speicherungen als sinnvoll. Diese Operation zur Balancierung des Baumes ist kostspielig, aber bei einer Sequenz von Änderungen über dem Baum, aus Gründen der Effizienz anzuraten.

Testläufe mit unbereinigten Daten, das heißt, solche mit Duplikaten und unsortierem Aufbau, haben beim Einlesen gezeigt, daß die Erzeugung einer Wertemenge aus mehreren tausend Tupeln durch die Verschmelzung von Duplikaten und der, nach festgelegter Anzahl von Änderungen der Baumstruktur, wiederholt durchgeführten Balancierung, die Zugriffzeiten wegen des für jeden Tupeleintrag erforderlichen Suchvorganges, erheblich reduziert.

Die Ausprägung des Baumes ist also nicht immer höhenbalanciert. Dies erschient dem Autor unter den Gesichtspunkten des Auftretens von Modifikationssequenzen nicht sinnvoll. Der Baum wird erst durch expliziten Aufruf der Methode höhenbalanciert. Leicht läßt sich der Baum zum permanent höhenbalancierten Baum in der Manier

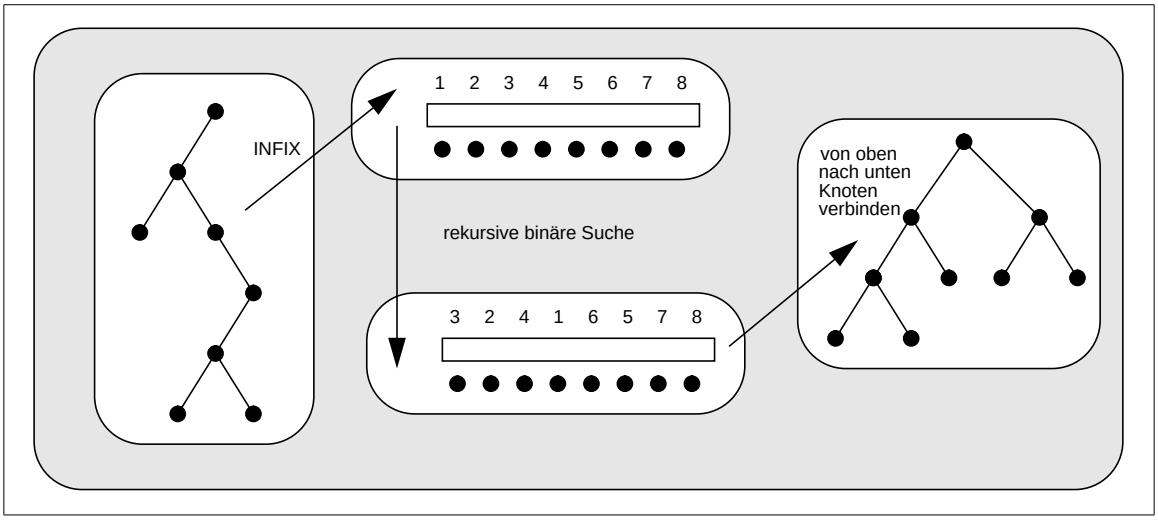

Abbildung 7-5: Balancierung der Wertemengen

eines AVL-Baumes ausbauen. Im schlimmsten Fall ist eine längere Modifikationssequenz mit wiederholter Strukturänderung des Baumes durchzuführen. Die Balance muß bei jedem neu eingefügten Tupel oder gar beim Löschen von Tupeln, wieder herzustellen sein. Für jedes Tupel sind dazu im schlechtesten Fall log(m) Änderungen am Baum durchzuführen, wobei m die Anzahl bereit eingebungener Tupel angibt. Daher wurde von permanenter Balancierung Abstand genommen.

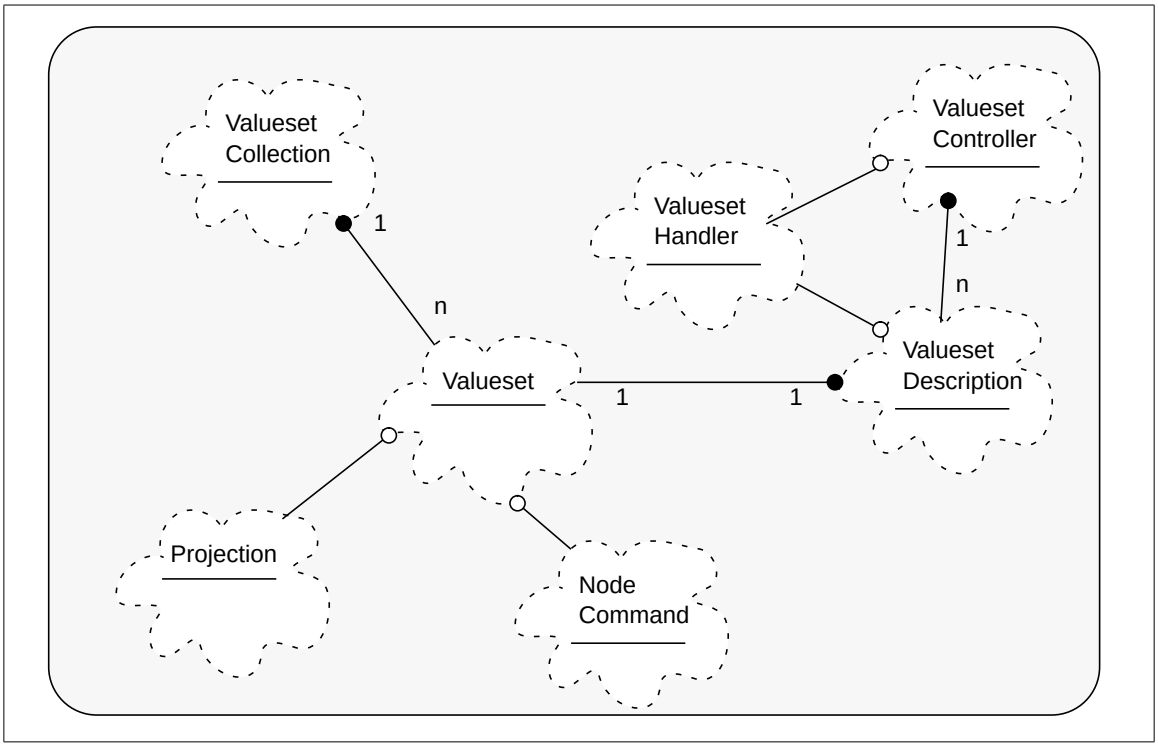

Abbildung 7-6: Struktur der Wertemenge VALUESET

Sinnvoller erscheint dem Autor der Ansatz angelehnt an Transaktionskonzepte. Am Anfang wird die Wertemenge in den Zustand gesetzt, daß jetzt eine Modifikationssequenz folgen wird. Wurde eine Anzahl neuer Tupel eingefügt wird temporär eine Optimierung durchgeführt. Nach Abschluß der Modifikationssequenz erfolgt der explizite Aufruf zur Balancierung des Baumes.

So können viele Änderungen an der Wertemenge durchgeführt und in festgelegten Intervallen eine Optimierung durchgeführt werden. Die Anzahl durchgeführter Änderungen auf der Wertemenge bestimmt wann die Höhe des Baumes balanciert wird. Diese Balanzierung wird natürlich am Schluß nochmals durchgeführt.

Der Suchbaum aus Tupeln wurde so ausgelegt, daß er möglichst viele Bereiche abdeckt, die auch zukünftig an die Repräsentationen von n-dimensionalen Wertemengen gestellt werden könnten.

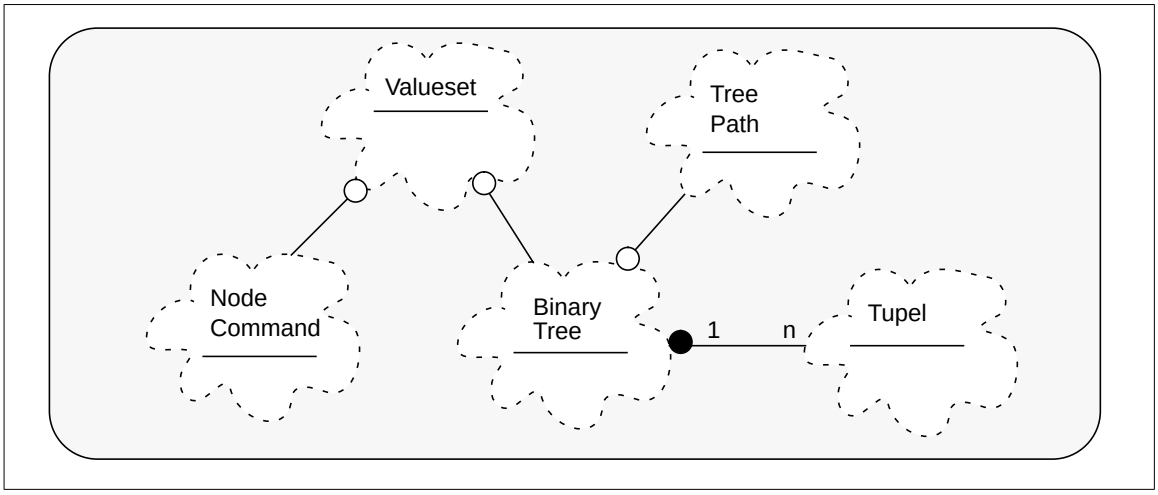

Abbildung 7-7: Darstellung von Kopplung Baum und Wertemenge

Diese umfassen:

- Das freie Festlegen der rekursiven Bearbeitungsreihenfolge :
  - in hierarchischen Strukturen:prefix, infix, postfix
- Das Festlegen eines Knotenkommandos; dies bedeutet, die auf ein Tupel der Wertemenge auszuführenden Operationen. Diese Möglichkeit der Ausführung wird in erster Linie zur rekursiven Bearbeitung über Teile des Baumes verwendet.
- Rekursive Bearbeitung über den Baum und dessen Teilbäume unter Berücksichtigung der Bearbeitungsreihenfolge, das heißt, Bereitstellen einer FOREACH Methode.
- Abbilden des Baumes auf das Verhalten eines Iterators durch Zugrundelegung der aktuell festgelegten Reihenfolge der Bearbeitung.

- Bereitstellen der Methoden des Iterators: start, finish, next, previous

Das erste eingefügte Tupel, also bis zur ersten Optimierung immer die Wurzel, legt den festen Dimensionsgrad der Wertemenge fest. Es werden daraufhin nur noch Tupel akzeptiert, die über genau die gleiche Anzahl an Attributen verfügen.

Dieser erweiterte Funktionsumfang erleichtert die Handhabung bei durchzuführenden Transformationen und Projektionen. Zur Veranschaulichung der Verwendung der FOREACH Methode ein konkreteres Beispiel.

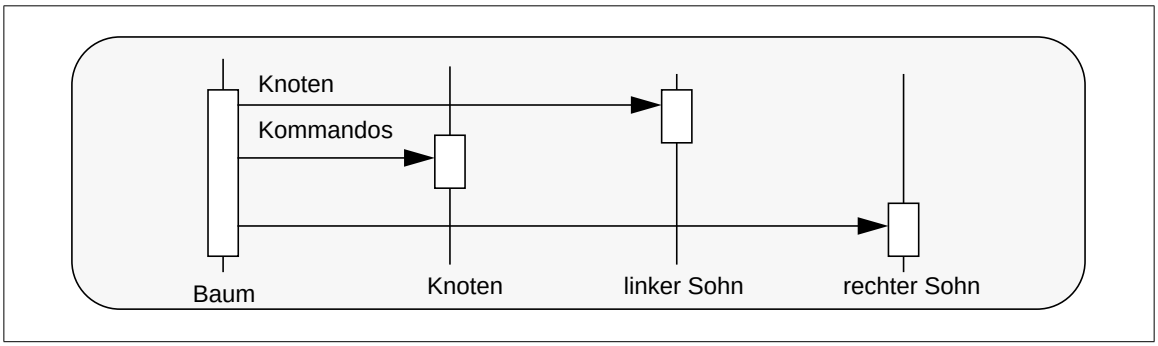

Abbildung 7-8: Szenario Foreach

Die Methode basierend auf denen eines Iterators ist sicherlich sehr wünschenswert, aber von ihrer Abarbeitungsreichenfolge nicht direkt anschaulich auf den Baum abzubilden. Durch die Bewegungsrichtung wird ein feste Reihenfolge der Knoten festgelegt, so daß es zu jedem Knoten genau einen eindeutigen Vorgänger und Nachfolger gibt.

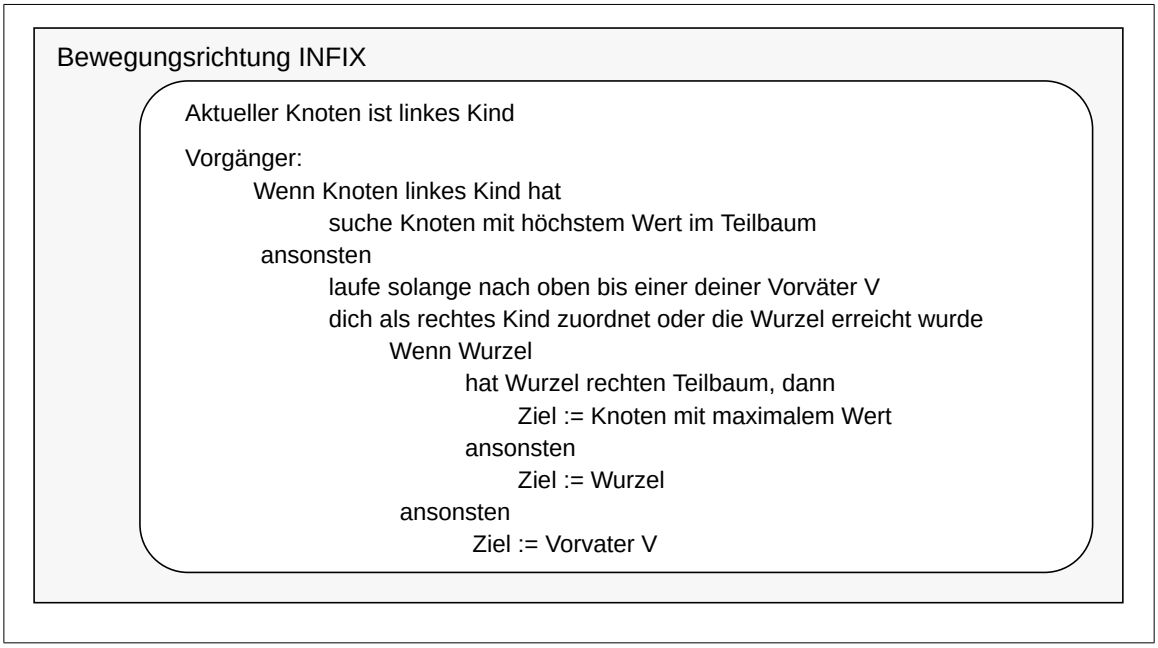

Abbildung 7-9: Arbeitsweise des Iterators: Vorgänger

Anders als bei der rekursiven Ausführung der FOREACH Methode, die ihren jeweiligen als nächsten zu bearbeitenden Knoten aus der Berarbeitungsreihenfolge und dem verbleibenden Stack implizit als Information abgreifbar hält, ist bei den Methoden der Iteratoren der nächste zu besuchende Knoten ausgehend vom aktuellen Knoten, zu berechnen.

Diese Berechnung ist nicht auf einen Blick ersichtlich, von daher wird ein Beispiel herausgegriffen und der Algorithmus beschrieben.

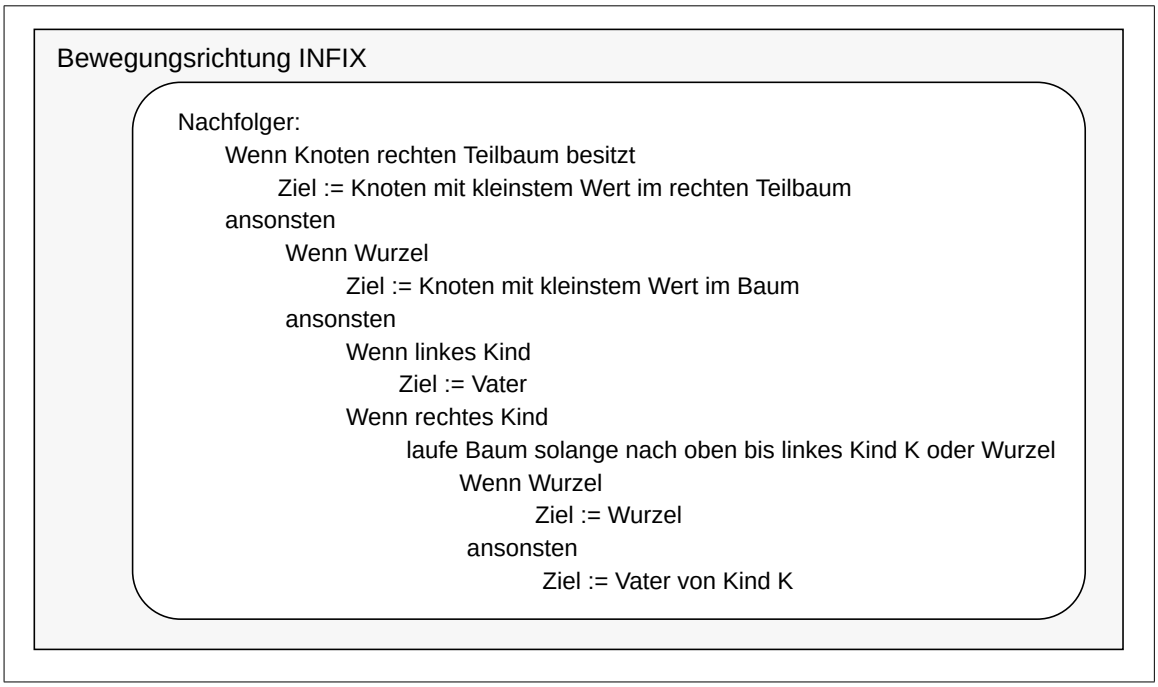

Abbildung 7-10: Beschreibung Arbeitsweise des Iterators: Nachfolger

7.2.2 Wertemenge optimiert auf Analyseaspekte

Die Darstellung der Wertemengen, die ihre Stärken im Aufgabenbereich der Auswertung aufweist, basiert auf Feldern und ist daher einfach und anschaulich. Sie wird im Rahmen der Ausarbeitung so ausführlich beschrieben, da im Kapitel 8 der Rückgriff auf diese Struktur unweigerlich notwendig wird.

Eine mögliche Abbildung ist ein Feld fester Länge aus Tupelreferenzen. Auf diese Variante wurde aus Geschwindigkeitsaspekten bei der iterativen Auswertung verzichtet. Der Vorteil dieser Darstellung ist, daß die Repräsentation der Tupel explizit erhalten bleibt, der Nachteil ist die dauernde Indirektion um auf Tupelwerte zugreifen zu können.

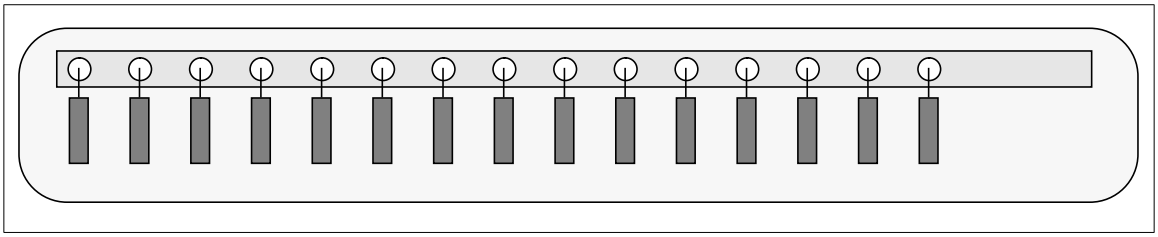

Abbildung 7-11: Einfachste Abbildung der Wertemenge

Eine zweite Möglichkeit ist allen Attributen und dem Zähler der Tupel jeweils ein Feld zuzuordnen. Die Möglichkeit verlangt aber wegen der nicht festgelegten Anzahl an Dimensionen eine Struktur ARRAY[ARRAY[Tupelattribute]]. Diese Abbildung wäre in ihrem Aufbau leicht nachvollziehbar und handhabbar, aber eine solche Struktur verlangt die Verwaltung von n Verweisen auf die aktuelle Position, um schnell die Werte an den Referenzen auszulesen oder die Verwaltung eines Indexwertes, der aber bei jedem Zugriff auf ein Attribute eine Indirektion durchführt.

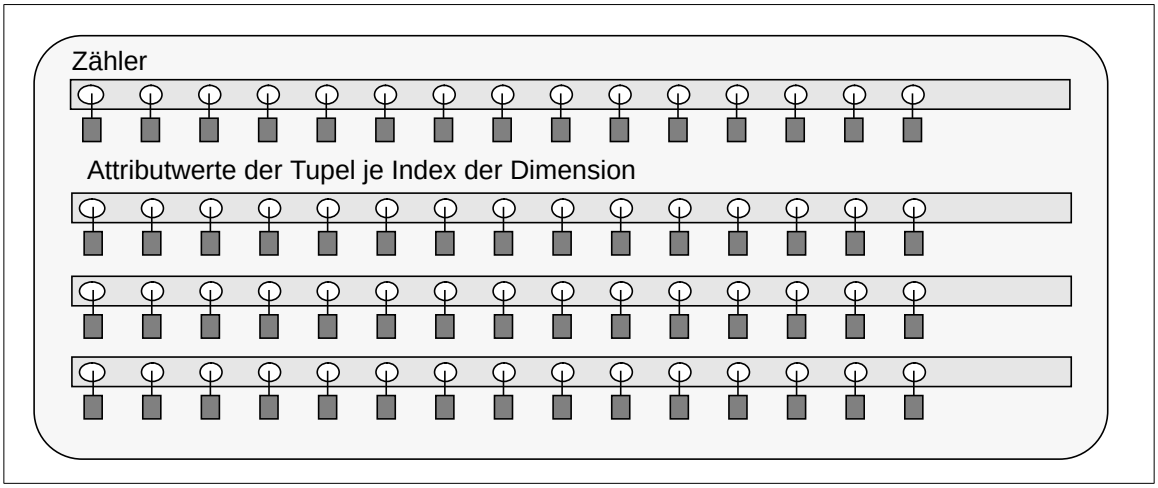

Abbildung 7-12: Zweite Möglichkeit der einfachen Abbildung

Daher wurde der folgenden Struktur der Vorzug gegeben. Jeweils für die Zähler und die Werte der Tupel werden getrennte Felder angelegt, so lassen sich leicht auch Wertemengen ohne Zähler mitverwalten.

Die Werte aller Attribute stehen in einem einzigen Feld, dabei stehen alle Werte eines Tupels direkt hintereinander. Der Versatz zwischen Tupelattributen mit gleichem Index von aufeinanderfolgenden Tupeln entspricht dem Dimensionsgrad der Tupel. Die Sortierung der Tupel erfolgt immer in der Wertigkeit der Tupelattribute. Der niedrigste Index eines Attributes innerhalb des Tupels hat die höchste Wertigkeit.

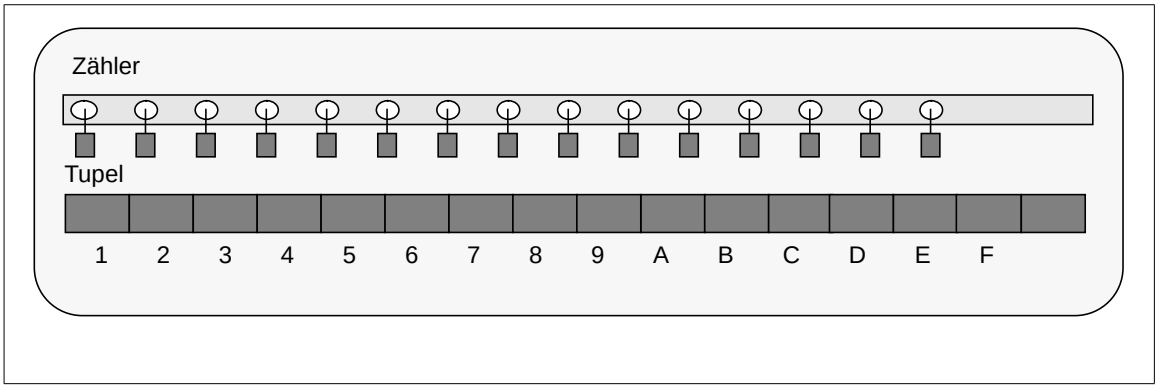

Abbildung 7-13: Die gewählte Abbildung für Wertemenge

Das heißt, gleiche Werte bei Attribut am ersten Index stehen nahe beieinander. Um zur Verdeutlichung ein Beispiel zu nennen: sind Wertepaare (X,Y,Z) gegeben und soll daraus ein Streugraph ermittelt werden, so werden Bereiche X und Y zusammengefaßt, daraus alle Z-Werte ermittelt und aus diesen Min/Max/Avg berechnet. Die Intervalle in X Richtung lassen sich wegen der Sortierung leicht beachten.

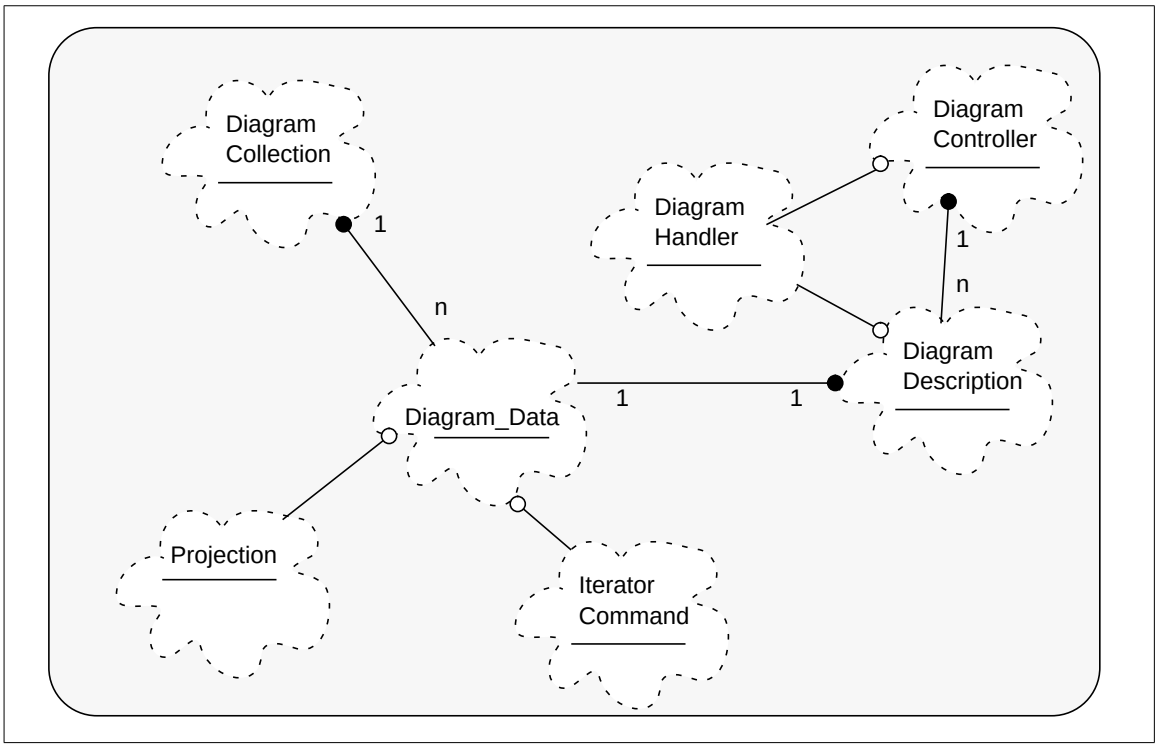

Abbildung 7-14: Verband für Diagramm Daten

Jedes dieser Intervalle gleicher X Werte hat eine Anzahl sortierter Y Werte. Diese müssen auf Intevallzugehörigkeit getestet und markiert werden. Wird eine Auswertung mit anderer Gewichtung der Indizes der Attribute vorgenommen, dann muß die Wertemenge durch Änderung der Indexpositionen und anschließender Sortierung umgebaut werden.

Wegen der gerade beschriebenen Eigenschaft bezeichnet der Autor die Darstellung der Tupel als implizit. Wird ein Wertepaar angefordert, muß dieses erst explizit erzeugt und mit den Werten gefüllt werden. Wegen der naheliegenden Verwendung und der geringen Eignung zur Modifikation wird diese Repräsentation im folgenden Kontext auch als Diagrammdaten bezeichnet. Im Falle der Visualisierung bezieht sich dies natürlich auf Dimensionsgrade unter drei. Die Struktur des Verbandes für Diagrammdaten gliedert sich wie in Abb. 7-15 beschrieben.

7.3 Repräsentation von Wertepaaren

Die Repräsentation der Tupel erfolgt wiederum durch ein Feld aus Tupelattributen und einem Zähler. Wichtiger als die Darstellung des Tupels ist hierbei die Vererbungsstruktur bei Verwendung der Tupel. Um die Gestaltung nicht auf FELDER festzulegen, werden abstrakte Klassen „vorgeschaltet“. Die Signatur der minimal anzubietenden Methoden ist dadurch festgelegt.

Bei einer Abbildung auf die generische Klasse FELD könnte der Einwand erfolgen, daß alle Attribute von gleicher Art sein müssen, also unterschiedlich Attribute innerhalb eines Tupels, wie sie bei Relationalen Datenbanken verwendet werden, nicht darstellbar sind. Diese Aussage ist prinzipiell richtig. Es ist erforderlich, daß eine einheitliche Oberklasse für die Gesamtheit aller möglichen Attribute zu definieren ist.

Diese eigentliche Repräsentation beruht auf der folgenden Aufgabenteilung für Tupel:

- Basisaufgaben, die für alle Tupel, auch solche ohne definierte Ordnung, wie zum Beispiel. das Auslesen von Attributen, vorhanden sein müssen.
    - Auslesen, Einfügen von Attributen
    - Konsistenztest, ob alle Einträge vorhanden sind, keine unzulässigen Werte existieren und Test, ob innerhalb des Tupels gleiche Attribute existieren
- Methoden, für den Vergleich von Tupeln. Der Vergleich basiert auf dem Vergleich der Einzelattribute, hierbei erhalten die Attribute mit niederem Index die höhere Wertigkeit.
    - Basismethoden für die Wertvergleiche „=“, „/=“, „<“ , „<=“ und *compare*
    - Flexiblere Methoden für die Vergleiche, die es erlauben Attribute von Tupellen unterschiedlicher Indizes zu vergleichen.

- Methoden für arithmetische Operationen beruhen wiederum auf der Durchführung der Einzelausführungen über den korrespondierenden Attributen.
  - Basismethoden für die Arithmetik aus der Klasse NUMERIC
  - Methoden, erweitert um die Vertauschung von Indizes.
- temporäre Methoden für die Ein/Ausgabe von Tupeln auf Datei
  - Diese sind nicht für das einzelne Tupel gedacht vielmehr werden sie bei der rekursiven Bearbeitung der Tupel im Suchbaum verwendet.

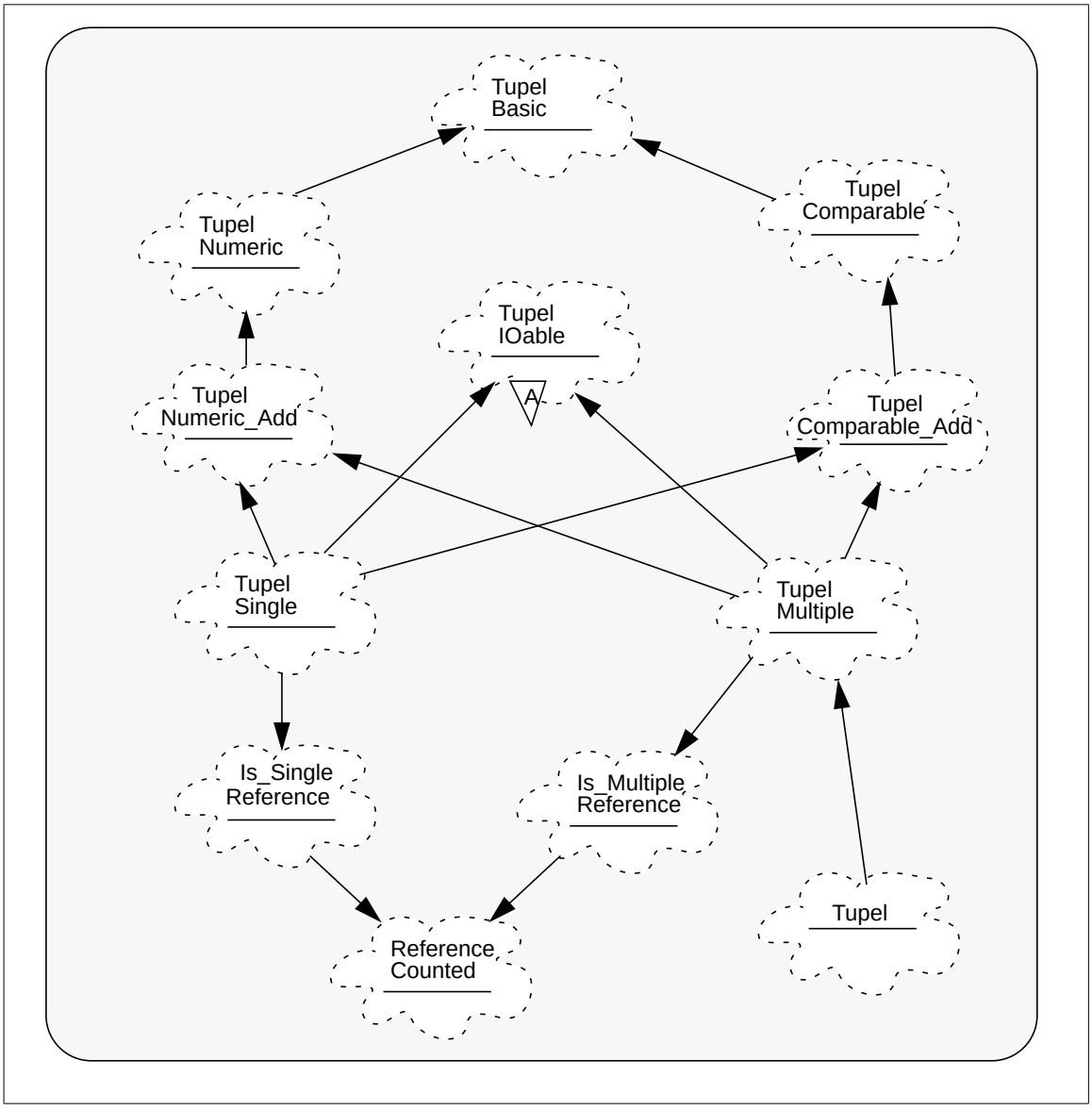

Abbildung 7-15: Verband für Tupelverwaltung

So ergibt sich die in Abb. 7-15 abgebildete Klassenhierarchie für den Verband zur Tupelverwaltung. Diese lassen sich durch Erben und anschließende Spezialisierung leicht auf Tupel mit anderen Attributen und solche mit unterschiedlichen Attributen ausweiten.

7.4 Konvertierung zwischen Repräsentationen

Die Konvertierung zwischen den unterschiedlichen Darstellungen der Wertemengen verlangt Konverter. Der Konverter stellt die Methoden zur Umwandlung zwischen den unterschiedlichen Repräsentationen bereit. Zusätzlich läßt sich der Intervallbereich der Konvertierung noch einschränken. Es wird bei der Ausführung eine neue Repräsentation erzeugt. Die Konvertierung läuft bei gegebener Intervallrestriktion zweistufig über einen Bitvektor. Erst werden alle zu konvertierenden Tupel gezählt und im Bitvektor markiert, dann wird daraus die erforderliche Größe der Zielrepräsentation ermittelt. Dieser Schritt entfällt bei der Konvertierung des gesamten Wertebereiches der Menge. In diesem Fall ist die Anzahl unterschiedlicher Tupel direkt abrufbar.

7.4.1 Konvertierung der Darstellung Baum nach Feld

Bei der Konvertierung von „Darstellung optimiert für Modifikationen“ nach „Darstellung optimiert auf die Analyse“ wird die Ausführungsrichtung auf Infix festgelegt, so daß die Knoten in der Reihenfolge ihrer Sortierung besucht werden. Zusätzlich wird ein Knotenkommando erzeugt, welches das im aktuellen Knoten befestigte Tupel dahingehend testet, ob dessen Attribute berücksichtigt werden sollen und danach eingefügt wird oder nicht. Hier wird auch eine Projektion eingebunden, falls diese gewünscht wird. Danach werden die Tupelattribute eingetragen und der Cursor der Zielrepräsentation um einen Index weitergesetzt.

7.4.2 Konvertierung der Darstellung Feld nach Baum

Bei der Konvertierung von Feld nach Baum wird analog vorgegangen. Es wird zur Erzeugung des Baumes eine temporäre Speicherung der Knoten benötigt. Nach Erzeugen des Bitvektors und Bestimmung der Gesamtanzahl zu konvertierender Tupeleinträge, es handelt sich ja nicht um wirkliche Einzelrepräsentationen von Tupeln, wird ein Feld aus Baumknoten erzeugt. Dieses Knotenfeld wird ausgehend von der Darstellung der Diagrammdaten unter Zuhilfenahme des Bitvektors mit frisch erzeugten Tupeln sortiert gefüllt.

Das Feld aus Baumknoten wird danach analog der binären Suche im Feld durchlaufen und die betroffenen Knoten rekursiv in den Baum eingefügt. Der entstehende neue Baum ist so konstruktionsbedingt höhenbalanciert. Bei der Konvertierung kann wiederum direkt eine Projektion miteingebunden werden.

7.5 Dimensionsreduktion durch Projektion

Der Aufgabenbereich der Projektion von Wertemengen ist erforderlich, um zum Beispiel Wertemessungen mit zehn charakteristischen Werten auf unter drei Dimensionen zu projezieren und diese dann darstellen zu können. Die Projektion wurde dabei auf einfache Basisoperationen beschränkt. Eine Erweiterung diesbezüglich sollte aber einfach sein.

Die Projektion ist in zwei Bereiche aufgeteilt. Die erste stellt einfache Lösungen bereit, das heißt in diesen Projektionen werden die überfüssigen Dimensionen durch einfaches wegschneiden, ausgeblendet. Bei der allgemeinen Projektion ist dagegen für die Tupel der Wertemenge eine explizite Projektionsvorschrift festgelegt. Die allgemeinste Form von Projektionsvorschriften lautet: Bilde einen neuen Attributwert aus einer Funktion die über alle Attribute des Tupels parametrisiert wird, also $Y_i = Func(X_1,...,X_n)$.

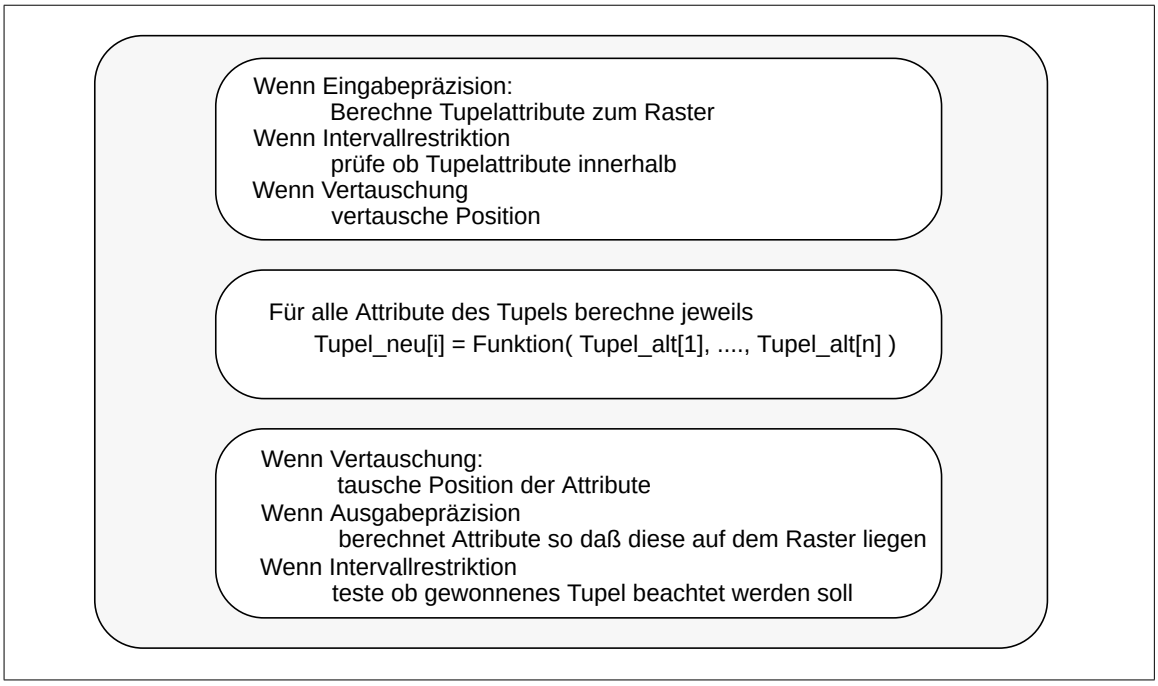

Abbildung 7-16: Vorgehensweise Projektion

Zusätzlich zur reinen Projektion kann in der Projektionsvorschrift noch eine Einschränkung des interessierenden Wertebereiches für Ein- und Ausgabe definiert werden. Diese Restriktion besteht in der Festlegung eines zulässigen Intervallbereiches jedes Attributes sowie einer möglichen Präzision. Dies ist hier aber nicht im Sinne der reinen mathematischen Auslegung zu verstehen, sondern in der Bedeutung eines Darstellungsrasters. Alle Attributwerte sollen so nahen Rasterwerten zugeordnet werden.

Es lassen sich natürlich auch die Indexpositionen der Attribute vertauschen. Diese Umsetzung der Projektion wird sich im Falle von unterschiedlichen Attributen innerhalb eines Tupels als besonders schwierig erweisen.

Der Vorgang der Projektion gliedert sich in drei, aufeinander folgende Schritte:

- Ausführungen auf dem Eingabetupel
  - Wird eine Präzision definiert, so sind die Attributwerte vorher auf das Raster vorgegebener Genauigkeit abzubilden.
  - Restriktion, ist das Tupel mit allen Attributen innerhalb der intressierenden Intervalles.
  - Vertauschungen der Indizes von Attributen
- Projektion der Werte
  - Festlegen der Basisfunktion für jede zu erzeugende Dimension. Bisher werden nur einfache Funktionen unterstützt, aber das Konzept läßt sich auf freidefinierende Funktionen erweitern.
- Ausführungen auf dem Ausgabetupel
  - Vertauschen der Indizes
  - Wird eine Präzision definiert so sind die Attributwerte des erzeugten Tupels auf das Raster abzubilden.
  - Test ob erzeugtes Tupel innerhalb des Intervalles ist.

7.6 Verwaltung und IO Handling

Zusätzlich werden für die Administration der Mengen noch Verbände benötigt, die sich mit der Verwaltung einer Ansammlung von Wertemengen befassen. Hierfür werden sogenannte Beschreibungen der Wertemengen zwischengeschaltet. In dieser werden eine Bezeichnung, ein Referenzzähler und ein Speicherungsname festgelegt.

Die applikationsweit verwendeten Wertemengen und die dazugehörigen Beschreibungen werden im Controller verwaltet. Dieser bildet die Schnittstelle zwischen Menüeinbindungen und interner Verwaltungsstruktur. Die IO-Handler dienen demselben Zweck, nur daß sie in diesem Fall die externe Ein- und Ausgabe bereitstellen und ein temporäres externes Speicherformat festlegen. Die Handler dienen der Erzeugung oder Speicherung interner Darstellungen aus extern abgelegten Wertelisten. Diese Repräsentation ist nur temporärer Natur, da eine Speicherung in einer Datenbank sinnvoller ist.

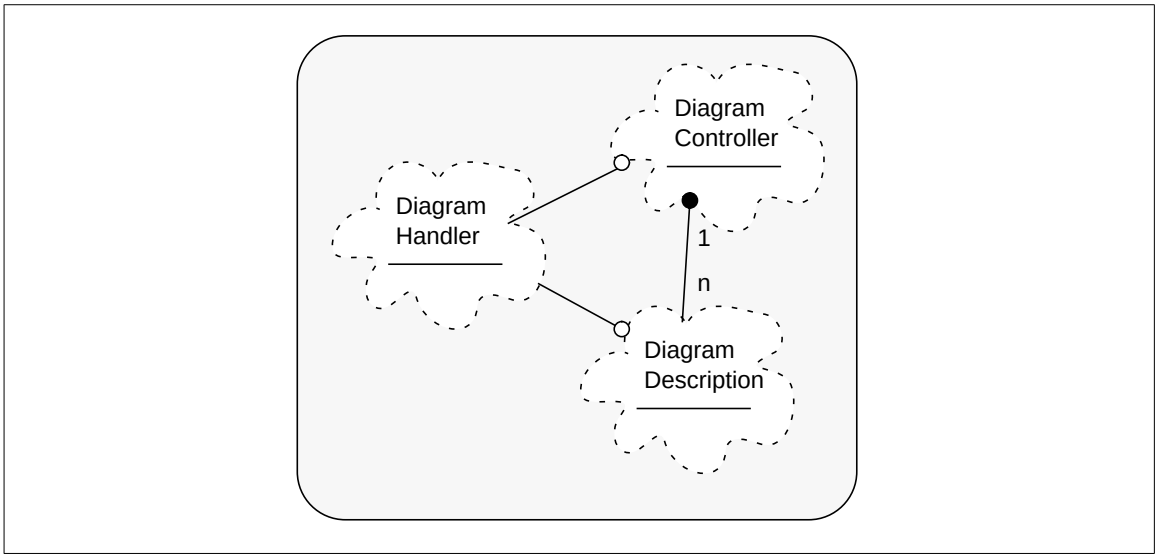

Abbildung 7-17: IO-Verwaltung

8
Analyse und Bewertung von Wertemengen

Das Kapitel stellt alle Mechanismen und Strukturen vor, die für die Analyse und Bewertung der n-dimensionalen Wertemengen notwendig sind. Es werden Methoden und Klassen bereitgestellt, die es ermöglichen, das Verhalten der Wertemenge zu analysieren und wenn notwendig auch zu prognostizieren. Eine Prognose setzt voraus, daß den Wertemengen ein systematisches Basisverhalten unterstellt werden kann. Diese These ist anhand von mathematischen Hilfsmitteln zu validieren. Die erforderlichen Erweiterungen für Betrachtung von Teilintervallen und Berücksichtigung der Tupelzähler ist in die Konzeption integriert.

8.1 Korrelations- und Regressionsanalyse

Für eine genauere mathematische Erläuterung der Regressions- und Korrelationsanalyse wird auf das entsprechende Kapiel 5, Anhang A und Anhang B verwiesen. Für die Analyse der Wertemenge, ausgehend von der Repräsentation, die für diese Aufgabe optimiert ist, sind zusätzliche Funktionalitäten erforderlich.

Die im folgenden Kontext noch weiter ausgeführten Funktionalitäten sind:

- Iteratoren, die auf einem Teilintervall der Wertemenge arbeiten und jeweils Berechnungen auf diesen Intervall durchführen. Da in der Analyse nur Iteratoren zur Berechnung von Summen über Teilintervallen der Wertemengen von Interresse sind, werden diese im weiteren Kontext als Summeniteratoren bezeichnet. Die Anwendung läßt sich aber leicht auf Produkte und andere Berechnungen wie Matrixiteratoren erweitern.
- Bereitstellen der Matrizen und deren Funktionalität. Bei der Berechnung werden nur 2-dimensionale Matrizen verwendet. Die Konzeption und Implementierung ist allgemein für n-dimensionale Matrizen ausgelegt und für 2-dimensionale Matrizen aus Gründen der Effizienz spezialisiert. Gemeinsam ist den Operatoren die iterative Arbeitsweise über der Wertemenge um nach vorgegebener Rechenvorschrift Werte zu berechnen.
- Für die allgemeine Regressionsanalyse ist eine einheitliche Abbildung, auch nicht linearer, funktionaler Abhängigkeiten zu wählen. Der Ansatz mittels vorgeschalteter Transformationen die Funktionen zu linearisieren und Polynome durch Aufbrechen in ihre jeweiligen Einzelterme zu vereinfachen, bietet diese

einheitliche Möglichkeit. In diesem Zusammenhang wird jeweils immer das Zusammenspiel aus Transformation und Rücktransformation als eine feste Einheit verstanden.

- Diese Transformation wandelt dadurch eine nicht lineare Abhängigkeit in eine lineare um. Bei der Transformation entsteht aus Abhängigkeiten die auf Polynomreihen beruhen, aus jedem Term in der Linearisierung jeweils eine neue Dimension.
  Anschaulich werden aus einem Regressionsansatz mit $x^5 + x^3$ für ein Attribut des Tupels zwei Dimensionen mit x_5 und x_3 durch die Transformationen x^{-5} und x^{-3} erzeugt. Analoges Verhalten gilt natürlich auch für Polynomreihenentwiclungen höherer Grade.

8.2 Operatoren für iterative Auswertungen über Wertemengen

Für die Auswertung ausgehend von den Wertemengen werden Operatoren verwendet, die Berechnungen über Teilintervalle dieser Mengen durchführen und dabei die Werte einzelner Attribute verwenden. Einfachstes Beispiel ist die Berechnung des arithmetischen Mittels über ein Teilintervall der Wertemenge. Dem Operator wird eine Wertemenge und bei Restriktion auf Teilintervalle ein Bitvektor, der zu berücksichtigenden Tupel übergeben. Dieser Vektor ist das Resultat einer vorher durchgeführten Bereichseinschränkung über der Wertemenge, auch als Intervallrestriktion bezeichnet.

Der Operator besitzt die Methode „*execute*“ und „*calculate*“, wobei nur die Methode „*execute*“ geerbt werden kann, da sich bei „*calculate*“ die Signatur der Rückgabewert ändern könnte. Zusätzlich wird noch eine Methode „*execute_region*“ angeboten, dieser wird anstelle des Bitvektors ein Intervallbereich mitgegeben. Werden nacheinander mehrere Operationen auf dem gleichen Teilintervall der Menge ausgeführt, ist aus Gründen der Geschwindigkeit der Abarbeitung die Verwendung des vorab ermittelten Bitvektors anzuraten.

Die Operatoren arbeiten basierend auf dem Konzept des Iterators. Dabei werden die Ausführungen über eine Schleifenkonstruktion FOREACH mit möglicher Einschränkung über Bedingungen ausgeführt. Die Ausführungen auf einer Tupelausprägung werden analog zu den Knotenkommandos in Kapitel 7.4 durchgeführt.

Werden bei der Berechnung Konstanten oder Indizes von Tupelattributen im Schleifenrumpf benötigt, so sind die Attribute des Operators vor der jeweiligen Ausführung explizit zu setzen. Es ist zwar in den meisten Fällen einfach die Konstante aus dem Schleifenrumpf und den darin enthaltenen Termen zu entfernen. Es kann aber aus Sicht der numerischen Genauigkeit manchmal sinnvoller sein, die Terme dort zu belassen. Dies ist abhängig vom jeweiligen Einzelfall und kann nicht allgemeingültig

angegeben werden. Im Fall der iterativen Operatoren über Wertemengen ist die Bezeichnung Tupelkommando, gegenüber der des Knotenkommandos bei baumartiger Struktur, anschaulicher.

8.3 Transformationen und Rücktransformationen

Die Transformation innerhalb der Ausarbeitung ist erforderlich, um die Analyse auf den Ansatz der multiplen, aber in diesem Fall linearen Regression abzubilden und so eine einheitliche Vorgehensweise zu ermöglichen. Der Vorteil dieses Ansatzes liegt zum einen in der allgemeinen Verwendbarkeit, zum anderen ist das Fundament der Regression, der Ausgangspunkt der Abschätzung, die minimale Summe aller Fehlerquadrate.

Unter den Transformationen und den damit gekoppelten Rücktransformationen werden nicht nur die Operationen basierend auf Basisfunktionen verstanden, sondern auch die zusätzlichen Operationen, die im Falle der Abbildungsvorschrift von Polynomen, hier in Form von Potenzreihenentwicklungen, bei der Linearisierung vorgenommen werden.

Dabei wird jedem Term der Potenzreihenentwicklung eine eigene Funktion zugeordnet. Somit entsteht aus x^n ein x_n. Die erforderliche Erhöhung des Dimensionsgrades wird im Austausch einer einheitlichen Vorgehensweise in Fall von Polynome billigend in Kauf genommen.

Es werden festvorgegebene Funktionen für die Transformation angeboten, zu denen bereits die Rücktransformationen definiert sind.

Diese Transformationen umfassen:

- Basis Funktionen
  - Potenzen von x: x^n
  - Logarithmus ln(x), $\log_a(x)$
  - Potenzfunktionen e^x, c^x
- Reihenentwicklung durch Polynomansätze aus Basisfunktionen

8.4 Regressions- und Korrelationsberechnungen

Die Regressions- und Korrelationsanalyse wird von zwei Ansatzpunkten her angegangen. Die einfachen Fälle der Regression wie zum Beispiel die lineare Regression für Wertemengen 2 bzw. 3 Dimensionen wird aus Effizenzgründen als Sonderfall angegangen. Der kompliziertere Ansatz soll den allgemeinen Fall lösen.

Beim einfachen Ansatz wird eine für diese Bedingung spezielle Lösung geschaffen. Diese Lösung wird aber nicht aus dem allgemeinen Ansatz abgeleitet. Bei den speziellen Lösungen werden nur deren festeingeschränkte Arbeitsbereiche gelöst.

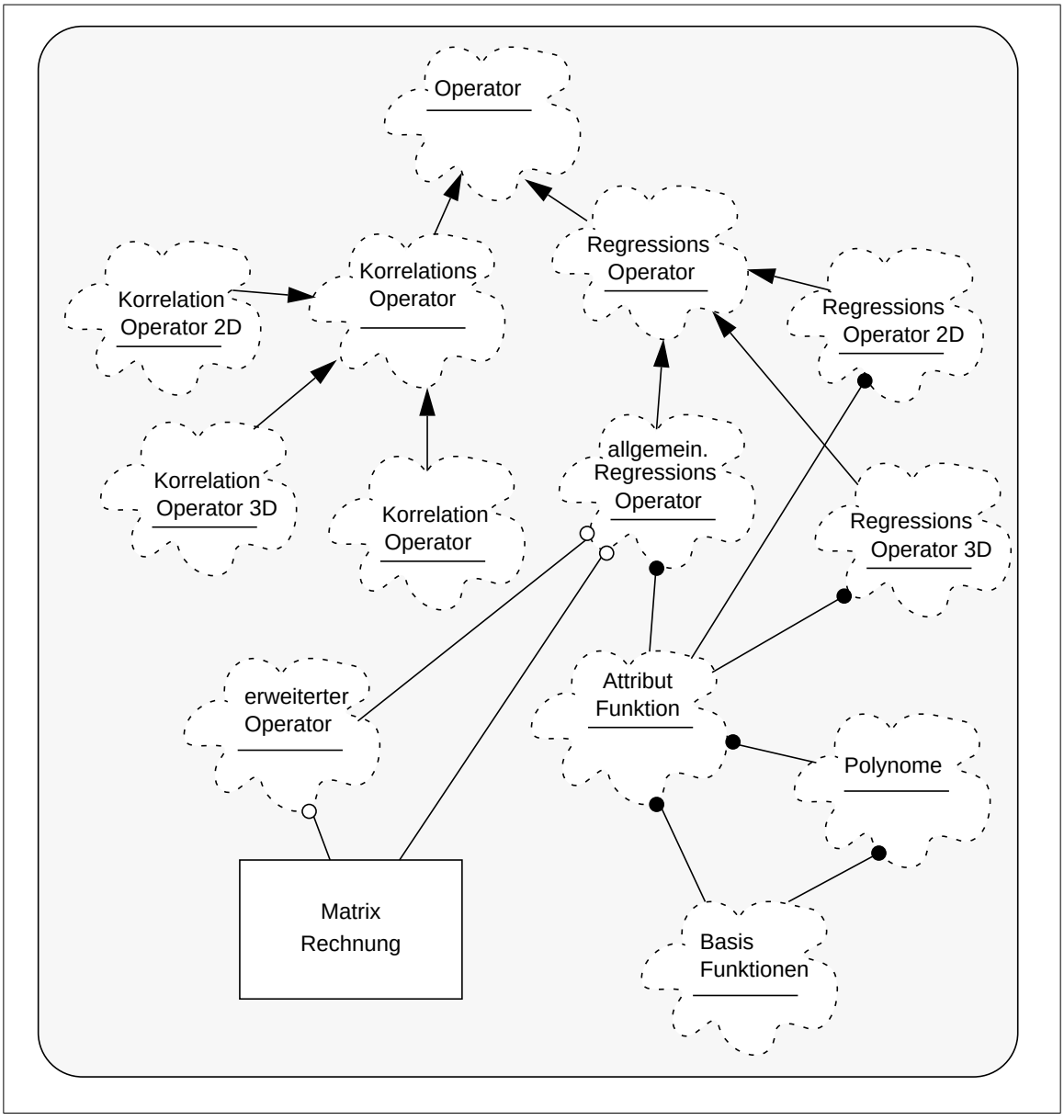

Abbildung 8-1: Vererbungsstruktur Operator

8.4.1 Korrelation

In der Korrelationsanalyse wird bei der Bestimmung der Pearson Korrelationkoeffizienten angesetzt. Im Fall von bi-partieller Korrelation läßt sich die Bestimmung auf Iterationen eines Operators über der Wertemenge zurückführen. Bei der Berechnung handelt es sich um die Ermittlung dreier Standardabweichungen.

```
kalkuliere_arithmetische Mittel
     -- Eingabe Wertemenge und Bitvektor über zu verwendende Tupel
do
     Allokiere Ergebnisvektor und setze Elemente 0

     Für alle Tupel T der Wertemenge W
          Zähler = W.Zähler[T]
          Tupel = W[Tupel]

          Für alle Dimensionen d jedes Tupels
               -- neu = alt + Tupelzähler * Attibutwert
               Avg[d] = Avg[d] + Zähler * Tupel[d]
          end; -- Schleife
     end; -- Schleife

     Für alle Dimensionen d der Menge
          Avg[d] = Avg[d] / Gesammtanzahl aller Tupel mit Zähler
     end; -- Schleife
end; -- kalkulate_arithmetic_average
```

Beispiele 8-1: Arithmetisches Mittel mit Tupelzähler

Bei der Berechnung der partiellen multiplen Korrelationskoeffizienten wird erheblich mehr Aufwand notwendig. Sie läßt sich zurückführen auf eine Matrizenrechnung aller Paarweise erzeugten bi-partiellen Korrelationskoeffizienten und nach Pearson berechnen, siehe Formel 5-10 auf Seite 45.

Zuerst wird die Berechnung aller Paarweise verschiedener Koeffizenten der Korrelationsrechnung nach Pearson durchgeführt. Dabei werden alle Dimensionen der Wertemenge miteinander korreliert. Diese Berechnung der n(n+1)/2 bi-partiellen Pearson Koeffizienten könnte nacheinander durch Ausnutzung des einfachen Operators der bi-partiellen Korrelation durchgeführt werden. Weil aber bei der Berechnung wiederholt gleichartige Entwicklungen von Reihensummen auftreten, wird für den Fall ein eigener Operator verwendet. Dieser Operator berechnet erforderliche Vektoren und Matrizen. Die für die weitere Berechnung erforderliche Matrizenrechnung wird in Kapitel 8.5 ausführlicher beschrieben.

Die Berechnung eines bi-partiellen Pearson Koeffizenten muß zwei Durchläufe über die Wertemenge durchführen. Im ersten wird das arithmetischen Mittel der betroffenen Tupel, bezogen auf alle Attribute ermittelt. Im zweiten Durchlauf wird der eigentliche Wert, daß heißt die zu bestimmende Korrelation errechnet. Bei der Berechnung von einer 2-dimensionalen Ausgangsmenge zur Resultatmenge werden schon 3 unterschiedliche Pearson Korrelationskoeffizienten benötigt, siehe Formel 5-9 auf Seite 44.

Im Falle der multiplen, partiellen Korrelation wird wie folgt vorgegangen. Ziel ist die Bestimmung der Paarweise verschiedenen bi-partiellen Pearson Koeffizienten und die Eintragung in eine quadratische Matrix, sowie die Erzeugung eines Vektors zur weite-

```
Fülle_Matrix_und_Vector
      Eingabe: setze vorher zu füllende Matrix, zu füllender Vektor
            Wertemenge, Bitvektor zu berücksichtigender Tupel
do
      allokiere
            Delta -- Differenz zwischen Wert und Mittelwert Avg
            QuadSum -- Differenz zwischen Wertquadrat und Anzahl*Avg

      setze 0: Delta[dim], QuadratSumme[dim], Matrix[d,d], Vector[d]
      setze 1: Diagonale von Matrix

      Avg = kalulate_arithmetisches_Mittel;

      -- Bestimmme wichtige Werte
      Für alle interessierenden Tupel T mit Zähler Z berechne
            -- berechne Quadratsumme und Summe aller Deltawerte
            Für alle Attribute A von Index 1 bis n + 1 ( auch Zielmenge )
                  QuadSum[A] = QuadSum[A] +
                        ( (Zähler * Tupel[A])^2 - Gesamtzahl * Avg[A] )
                  Delta[A] = Delta[A] + Zähler * (Tupel[A] - Avg[A])
            end;
            -- berechne Paarweise Korrelation der X-Werte
            Für alle Attribute A1 der Tupel von Dimension 1 bin n - 1
                  Für alle Attribute A2 der Tupel von (A1+1) bis n
                        Matrix[A1,A2] = Matrix[A1,A2] + (Delta[A1]*Delta[A2])
                  end;
            end;
            -- berechne Korrelation von Zielwerten Y und X Werten
            Für alle Attribute A1 der Tupel von Dimension 1 bin n
                  Vector[A1] = Vector[A1] + ( Delta[A] * Delta [n+1] )
            end;

            -- Normiere die Matrix
            Für alle Attribute A1 der Tupel von Dimension 1 bin n - 1
                  Für alle Attribute A2 der Tupel von (A1+1) bis n
                        Matrix[A1,A2] = Matrix[A1,A2] /
                              Wurzel(QuadSum[A1]*QuadSum [A2])
                  end;
            end;
            -- Normale Zielvektoren
            Für alle Attribute A1 der Tupel von Dimension 1 bin n
                  Vector[A1] = Vector[A1] / (QuadSum[A] * QuadSum[n+1])
            end;
      end;
end;
```

Beispiele 8-2: Fülle Matrix für multiple Korrelationsberechnung

ren Berechnung. Die Wertemenge sei durch n+1 Dimensionen beschrieben, dabei entfallen die ersten n Dimensionen auf Werte X_1,..., X_n der Ausgangsmenge und die letzte Dimension der Tupel repräsentiert die Zielwerte Y.

Im ersten Durchlauf werden die arithmetischen Mittel aller Tupelattribute unter Berücksichtigung der Tupelhäufigkeit ermittelt. Dies gilt auch für den Bereich der Zielwerte, also dem Tupelattribut mit dem höchsten Index.

Im zweiten Durchlauf werden mehrere Berechnungen durchgeführt. Die Berechnung ist so ausgelegt, daß mehrere Schleifenschachtelungen existieren. Das Resultat der Berechnung ist ein Vektor und eine Matrix mit denen weitergerechnet werden kann. Hierfür wird auf die Matrizenrechnung zurückgegriffen.

8.4.2 Regression

Soll eine Regression durchgeführt werden, wird analog vorgegangen. Die einfachen Berechnungen der Regression werden mit speziellen Operatoren durchgeführt. Bei der linearen Regression wird die Steigungsgröße b über die Formel 5-13 auf Seite 46 berechnet.

```
berechne_konstanten:Rüchgabewert
      Eingabe 2-dim Wertemenge und Bitvektor
do
      allokiere und setze 0: Avg_X, Avg_Y, ...

      Für alle intressierenden Tupel T der Wertemenge W
            Avg_X = Avg_X + Zähler[T] * X_Wert[T]
            Avg_Y = Avg_Y + Zähler[T] * Y_Wert[T]
      end; -- Schleife
      Avg_X = Avg_X / Anzahl aller Tupel
      Avg_Y = Avg_Y / Anzahl aller Tupel

      Für alle intressierenden Tupel T der Wertemenge W
            Summe_Delta_X = Summe_Delta_X + X_Wert[T] - Avg_X
            Summe_Delta_Y = Summe_Delta_Y + X_Wert[T] - Avg_X
            QuadSum_Delta_X = QuadSum_Delta_X + ( X_Wert[T]^2 - Axg_X^2 )
      end; -- Schleife

      Resultat_Steigung := ( Summe_Delta_X * Summe_Delta_Y ) / QuadSum_Delta_X
      Resultat_Verschiebung := Avg_Y - Resultat_Steigerung * Avg_X
end; -- berechne_steigung
```

Beispiele 8-3: Konstantenberechnung der Regression

Die Verschiebung der Approximationsgeraden im Ursprung läßt sich leicht ermitteln, da garantiert ist, daß die Gerade immer die Koordinate, die dem arithmetischem Mittelwert beider Dimensionen entspricht, schneidet.

Analog wird für den Fall von drei Dimensionen, das heißt zwei Wertparametern und einem Resultatwert ein spezieller Operator bereitgestellt. Dieser Operator geht auf die Formel A-18 auf Seite 162 zurück und verwendet die dafür erforderlichen Hilfskonstanten.

Im allgemeinen Ansatz wird nur multiple Regressionen mit linearer Abhängigkeit verwendet. Sind Polynome mit Potenzentwicklungen anderer Ordnungen oder Terme anderer Art vorhanden, muß vorab transformiert werden. Die multiple Regression wird dabei auf den allgemeinen Ansatz über die Matrizenrechnung, mit Erweiterung die die Häufigkeiten der Tupel berücksichtigen, zurückgeführt.

Die Berechnung mit der Verwendung der Zähler ist aber nur korrekt, wenn diese bei der Gesamtberechnung wieder eliminiert werden und der „normalen“ Matrixmultiplikation entsprechen, die entsteht, wenn alle Tupel mit Zählern größer 1 einfach n fach in die entsprechende Matrixrepäsentation eingetragen werden.

8.4.3 Kompletter Vorgang bei der Analyse

Der komplette Vorgang der Regressions- und Korrelationanalyse wird wegen der möglicherweise vorgeschalteten Transformationen und Linearisierungen in mehreren Ausführungsschritten berechnet. Zuerst erfolgt die Wahl der Basisfunktionen, die das Verhalten der funktionalen Abhängigkeit der Tupelattribute zu dem Ergebnisattribut am besten beschreiben. Durch die Wahl der Transformation wird implizit die Wahl der Rücktransformationen festgelegt. Bei der Wahl von Polynomen lassen sich Anfangskonstanten festlegen, mit denen die Regression ihre Startanalyse ansetzt. Durch die Transformation und die darin festgelegten Basisfunktionen wird die Anzahl der Dimensionen für die Rechenmatrix und die Vektoren bestimmt, die der multiplen Regression zugrunde gelegt werden. Werden keine Polynome verwendet entspricht die Anzahl der Dimensionen denen der Tupel der Wertemenge.

Danach wird mit dem erweiterten Operator für Matrixmultiplikation, der auf die Verwendung von Zählern der Tupel optimiert ist, die Regression berechnet. Die Berechnung ermittelt den approximierenden Steigungsvektor. Dieser Steigungsvektor ist bei den Basisfunktionen in der anschließenden Berechnungen zu berücksichtigen. Die konstanten Terme müssen dazu noch transformiert werden.

Die jetzt resultierenden Basisterme bilden den Ansatzpunkt der Regressionsabschätzungen basierend auf der funktionalen Abschätzung der Kurve. Deren Prognosequalität in Bezug auf das reale Verhalten der Wertemenge läßt sich mit der multiplen Korrelationsanalyse testen.

Die Ausführungsreihenfolge ist also:

- Wahl der Basisfunktionen, mit der jede Dimension des Tupels in der Wertemenge funktional abgeschätzt wird.
- Transformation der Werte
- Berechnen des Steigungsvektors in der erweiterten Matrixmultiplikation und ermitteln des Restbetrages.
- Transformiere den Steigungsvektor zurück in die Ausgangsmenge und setze diese Terme in die ausgehenden Basisfunktionen.

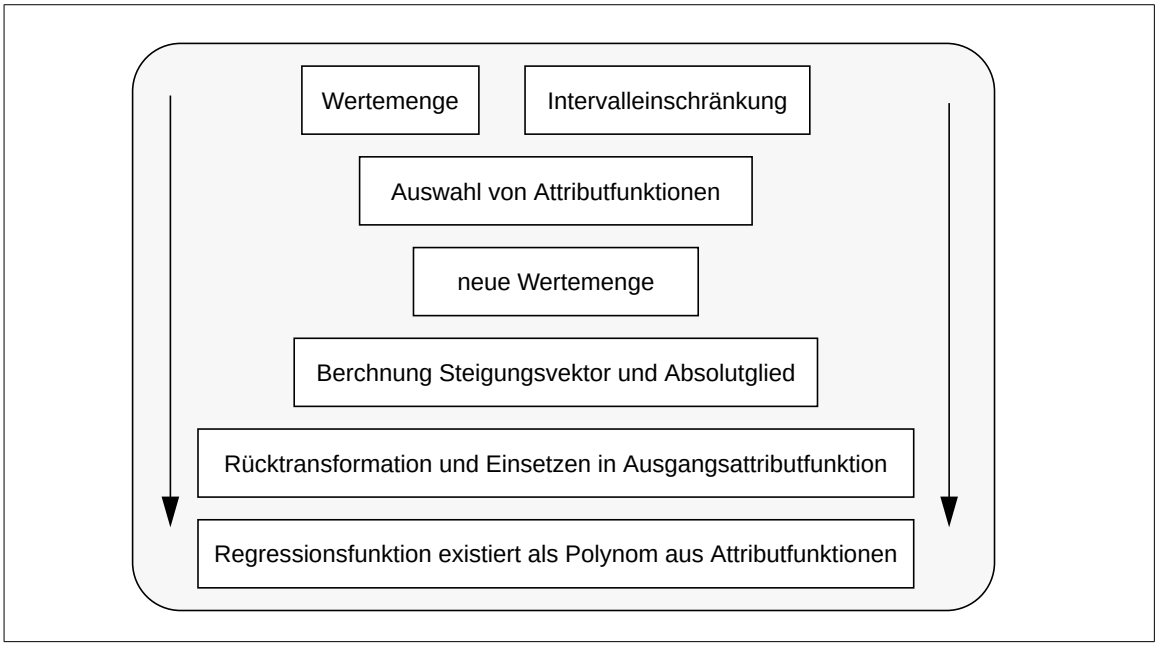

Abbildung 8-2: Abarbeitungsschritte der Analyse

8.5 Matrizenrechnung

Die Matrizenrechnung ist so ausgelegt, daß die Struktur sich in Bereiche von möglichen Verwendungen und Eigenschaften ihrer generischen Basistypen, den einzelnen Zellenattributen, gliedert und der Ansatz wird so allgemein wie möglich durchgeführt. Es wird von der n-dimensionalen Matrix ausgegangen, bei der die Dimensionsanzahl sowie der Größe der einzelnen Dimension, frei gewählt werden kann. Da dieser Ansatz in Bezug auf die Geschwindigkeitsaspekte ein deutliches Defizit aufweist, wird für den Fall von zwei Dimensionen eine Spezialisierung durchgeführt.

```
berechne_position
      Eingabe Vektor aus Indexpositionen, Position von 0..(max - 1)
do
      Wertigkeit := 1;
      Result := 0

      Für alle Dimensionen D von hinten nach vorn berechne
            Result := Result + Wertigkeit * ( Position[D] - min[D] )
            Wertigkeit : = Wertigkeit * ( max[D] - min[D] )
      end; -- loop
end; -- berechne_steigung
```

Beispiele 8-4: Positionsberechnung bei n Dimensionen

Die n-dimensionalen Matrizen operieren mit Hilfe von einfache Operatoren, die sich auf die Verknüpfung von Einzelattributen beziehen und somit keine Iteratoren über alle Indexe verwenden müssen, um die Attribute an Index i mit denen an Index j zu verknüpfen. Es werden somit nur Attribute des gleichen Index verknüpft oder Konstanten addiert.

Für die Bearbeitung wird eine eigene Berechnung des Index ausgeführt, die die Position des Eintrages aus den Einzelindizes jeder Dimension ermittelt. Iterativ ausformuliert lautet die Berechnung wie in Abb. 8-4 dargestellt und im Falle der Spezialisierung bei Matrizen mit zwei Dimensionen wird der Rang der ersten Dimension fest an die Indexberechnung einbezogen.

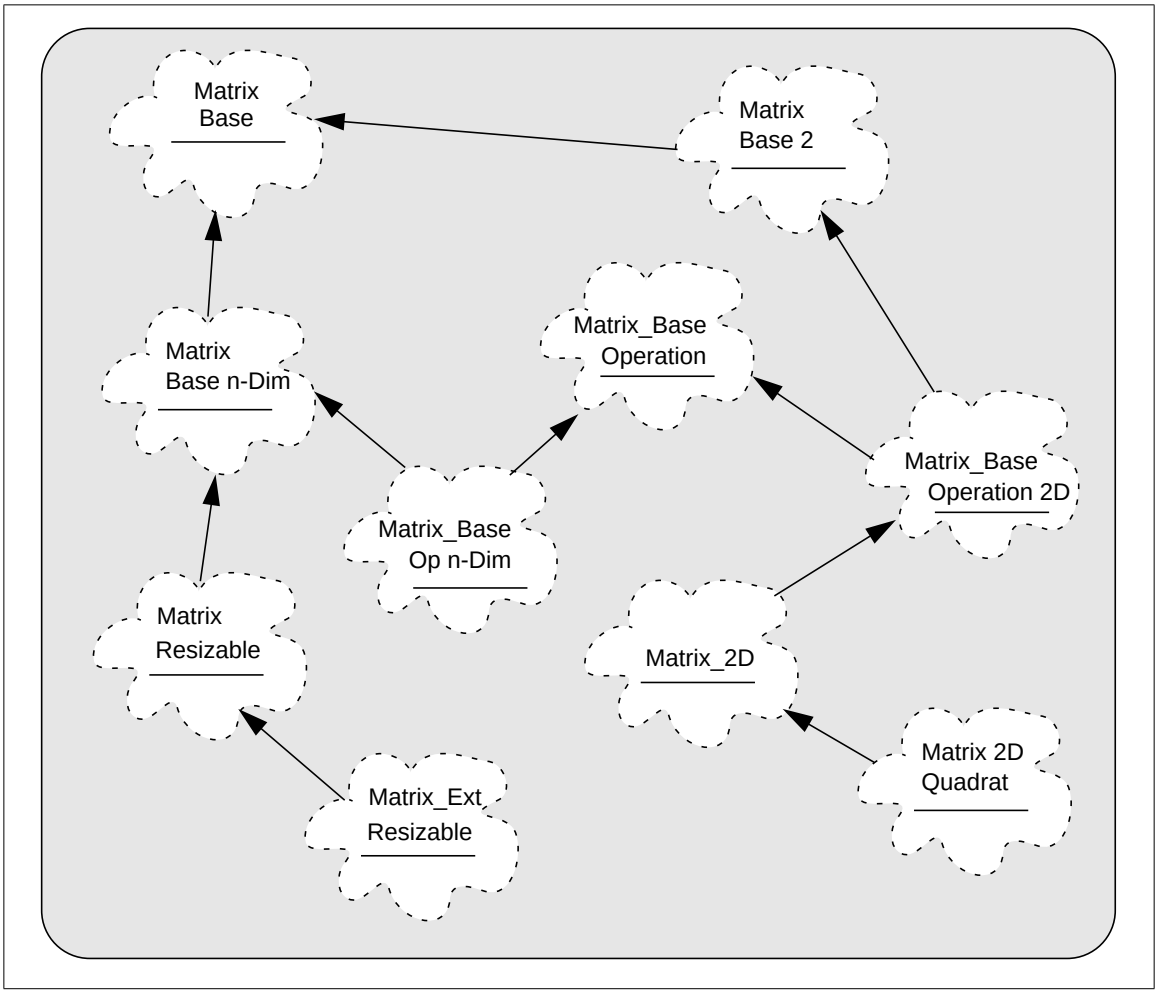

Abbildung 8-3: Verband zur Durchführung der Matrizenrechnung

Zulässige Operatoren für Matrizen mit zwei Dimensionen sind Matrixmultiplikation und das Bilden von transponierten Matrizen. Zusätzlich werden noch die Operatoren X^TX und XX^T sowie Verknüpfungen mit einem Vektor entsprechender Größe durch Y^TX und XY^T angeboten. Der für diese Berechnung erforderliche Operator zur Berechnung des Steigungsvektors $b = (X^TX)^{-1}(X^TY)$ ist vorhanden.

Ist die Matrix von quadratischer Gestalt lassen sich weitere Operatoren anwenden. Von einer Trennung zwischen quadratischen und anderen 2-dimensionalen Matrizen wird abgesehen, diese eingeschränkte Nutzung für bestimmte Operatoren wird über definierte Vorbedingungen eingefordert.

Mögliche Operatoren auf quadratische Matrizen sind:

- Test auf Invertierbarkeit
- Determinantenberechnung
- Berechnung der Inversen, bei nicht verschwindend kleiner Determinante
- Operator zur Variation der Matrix um die Invertierbarkeit zu garantieren

Der Test bezüglich der Invertierbarkeit beruht auf dem Gauss'schen Normalisierungsverfahren und berechnet eine Diagonalmatrix. Bei linearer Abhängigkeit bilden sich Zeilen aus Nullenvektoren.

```
Test_Lineare_Abhänigkeit:Wahr oder Falsch
      -- test ob eine Lineare Abhängigkeit zwischen den Spalten und Zeilen besteht
do
      Resultat := True

      Für alle Zeilen Z der Matrix Index i von 1 bis n oder (Result = False)
            maximal_Zeile := i;
            -- suche maximalen Eintrag
            Für alle Zeilen ZF der Matrix Index j von i+1 bis n
                  wenn Betrag(Matrix[j][i]) > Betrag(Matrix[i,i]) dann
                        maximale_Zeile := j

            -- vertausche Zeile i und j
            Wenn die maximale_Zeile geändert ist
                  Für alle Spalten der Matrix Index j von i bis n
                        vertausche(Matrix[i][k],(Matrix[max,k])
                              maximale_Zeile := j
            Wenn Matrix[i,i] = 0 dann
                  Result := False
            ansonsten
                  Für alle Zeilen ZF der der Matrix Index j von i+1 bis n
                        Für alle Spalten k von n bis j setze
                              Matrix[j,k] = Matrix[j,k] -
                                    Matrix[i,k]*Matrix[j,i]/Matrix[i,i]
                        Matrix[j,i] := 0
                  end; -- loop
end; -- Test_Lineare_Abhänigkeit
```

Beispiele 8-5: Test auf lineare Abhängigkeit

Die Berechnung der invertierten Matrix erfolgt über rekursives Anwenden der Cramer'schen Regel, das heißt die Determinante läßt sich über Streichung entsprechender Zeilen- und Spaltenindizes rekursiv auf Determinanten von Matrizen zurückführen, deren Rang um 1 kleiner ist, als der Rang der Ausgangsmatrix.

Es gilt:

$$\mathrm{Det}(A) = \sum_i (-1)^i \cdot \mathrm{Det}(A_{ij}) = \sum_j (-1)^j \cdot \mathrm{Det}(A_{ij})$$

Formel 8-1: Crame'rsche Regel zur Berechnung der Determinanten

wobei A_{ij} die resultierende Matrix nach Streichung der i. Zeile und der j. Spalte ist. Zur rekursiven Berechnung wird ein Bitvektor verwendet, der die jeweils gestrichenen Zeilen markiert.

```
Determinante_Intern( Bitvektor bereits gestrichener Zeilen Z, Spalten Sp)
      -- test ob eine Lineare Abhängigkeit zwischen den Spalten und Zeilen
      -- besteht
do
      Anzahl := Anzahl_gestrichener_Zeilen( Vektor )

      Wenn ( dim - Anzahl = 2 )
            Result := Matrix[Sp.low,Z.low]*Matrix[Sp.high,Z.high] -
                        Matrix[Sp.high,Z.low]*Matrix[Sp.low,Z.high]
      ansonsten
            Resultat auf 0
            streiche nächste Zeile im Zeilenvektor
            setze Cursor von Spaltenvektor auf Anfang

            toggle_falg := True;
            Nehme solange den nächsen freien Index i des Spaltenvektors bis Ende
                  Setze Spaltenvektor von Index i auf gelöscht
                  wenn toggel_flag dann
                        Result := Result + Determinante_Intern( Z, Sp );
                  ansonsten
                        Result := Result - Determinante_Intern( Z, Sp );
                  toggle_falg := not toggle_fla
                  Setze Spaltenvektor von Index i auf nicht gelöscht
            end; -- loop
end; -- Determinante
```

Beispiele 8-6: Berechnung der Determinanten

Die Variation der Matrix zur Berechnung wird nur über einfachste Operatoren durchgeführt. Die Matrizenelemente werden über eine iterativ Berechnung in festgelegten Intervallschritten um Deltawerte variiert. Das bedeutet zum Beispiel, jeder dritte Eintrag wird um etwas erhöht und jeder sechste um den gleichen Wert erniedrigt.

8.5.1 Die Erweiterungen für Tupel mit Zählern

Die hier vorgestellte Erweiterung der Operatoren auf Verwendung von Tupeln mit Zählern sind von ihrer Auslegungsweise eng an die Matrizenrechnung gekoppelt. Bei der Berechnung soll auf ein Hin- und Herkopieren zwischen den unterschiedlichen Matrizen und Wertemengen aus Gründen der Effizienz verzichtet werden. Daher werden die Operatoren für die Matrixoperationen zwischen einer Matrix und ihrer Transponierten oder der Matrix und einem Vektoren dahingehend optimiert, daß bei Operationen die implizite Tupeldarstellung samt ihrer Häufigkeitszähler Berücksichtigung findet. Bei der Berechnung von XX^Tentsteht dabei eine erheblich größer Resultatmatrix

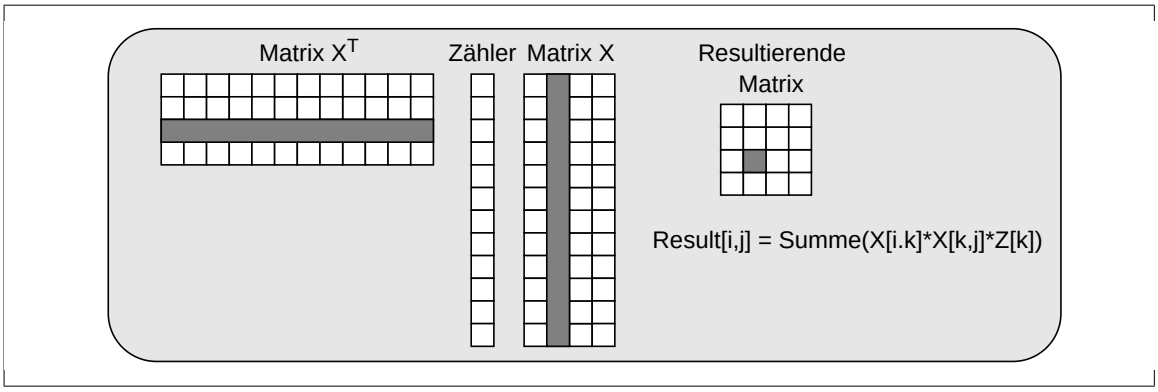

Abbildung 8-4: Funktionsweise des erweiterten Operators

Wie berechnet sich jetzt X^TX oder X^TY bei gegebener Wertemenge und Ergebnisvektor. Die resultierende Matrix ist quadratisch vom Grad um eins kleiner als die Dimension der Tupel, da die Werte der Zielmenge nicht verwendet werden. Y ist der Vektor in den alle Werte der Resultatsmenge eingetragen werden. Bei der Multiplikation werden die Zähler an die ursprüngliche Matrix, bzw. den Vektor gekoppelt und bei jeder einzelnen Matrizenmultiplikation berücksichtigt, das heißt a_ib_i wird jetzt $z_ia_ib_i$, mit z_i als Zähler des Index i.

```
Berechne_transponiert_mal_normal
      -- liefere Matrix zurück
do
      Für alle Wertedimensionen i
            Für alle Wertedimensionen j
                  Result[i,j] := 0
                  Für alle Tupel T der Wertemenge mit Index t
                        Result[i,j] = Result + T.Attribute[i] * T.Attribute[j]* Zähler
end; -- Test_Lineare_Abhängigkeit
```

Beispiele 8-7: Erweiterte Matrixmultiplikation von X^TX

9
Visualisierung von Wertemengen

In diesem Kapitel werden die Mechanismen zur Visualisierung von Wertemengen aufgezeigt. Im Zusammenhang mit dem nachfolgenden Kapitel 10 werden unter Wertemengen, jetzt diejenigen verstanden, welche ohne weitere Transformationsschritte direkt darstellbar sind.

Es wird zuerst auf den allgemeinen Aufbau der Mechanismen eingegangen. Anschliessend werden die unterstützenden Strukturen zur Visualisierung von Mengen und der dazu erforderlichen Darstellungsmodelle, welche die jeweils interessierende Interpretation über der Menge beschreiben, skizziert. Am Schluß wird auf das Zusammenspiel der einzelnen Bestandteile eingegangen und wie sich aus diesen Bestandteilen ein Diagramm aufbauen läßt. Der Baukasten und die Vorgehensweise zur Erzeugung neuer Diagrammarten wird in Kapitel 10 vertieft.

Alle hier beschriebenen Sachverhalte beziehen sich bei nicht ausdrücklicher Festlegung auf die Repräsentation des Modelles und nicht die visuellen Repräsentationen, sogenannte „Views“ .

9.1 Grundlegende Gedanken zur Visualisierung

Zu Beginn des Kapitels werden Grundkonzepte der Visualisierung besprochen. Bei der Visualisierung haben sich die vorgestellten Konzepte, unter den Gesichtspunkten Allgemeingültigkeit, Übersichtlichkeit und Nachvollziehbarkeit, als geeignet erwiesen.

Der wesentliche Gedanke ist: „Wie läßt sich der Bereich der Visualisierung und die darin befindlichen visuellen Repräsentationen gliedern und modellieren, so daß ein anschauliches und überschaubares Entwurfsszenario für den Entwickler entsteht“.

Diese Bedingung führt zur Forderung der Verwendung von geschachtelten Strukturen in der Verwaltung und impliziert dadurch eine einfache, überschaubare Hierarchie, die es sukzessive zu verfeinern gilt.

9.1.1 Verwaltung durch Verwendung geschachtelter Rahmen

Der Ansatzpunkt der Visualisierung von Dokumenten und den darin enthaltenen Diagrammen ist die Verwendung von Rahmen als abstrakte Oberklasse aller zu definierenden Strukturen der Verwaltung. Hierunter werden Objekte verstanden, die ihrerseits Methoden für Aufgaben der Verwaltung ausführen können.

Diese Rahmen verwalten Objekte, die ihrem Darstellungbereich zugeordnet sind. Die Objekte dürfen den Bereich der Darstellungsfläche der zugeordneten Rahmen nicht überschreiten. Einer Erweiterung hinsichtlich der Funktionalität des „Clippings“ oder des „Abschneidens“ dargestellter Objekte, die aus dem Rahmen herausfallen findet keine Berücksichtigung.

Bei den Rahmen wird zwischen zwei unterschiedlichen Arten, abhänigig von ihrer Funktionsweise unterschieden. Der eine Rahmen verwaltet nur einen „besetzten“ Bereich innerhalb der Darstellungsfläche, unabhängig davon ob der Rahmen samt Inhalt angezeigt wird oder nicht. Zusätzlich gibt es eine Spezialisierung, die Methoden der Verwaltung, wie Koordinatentransformation bereitstellen.

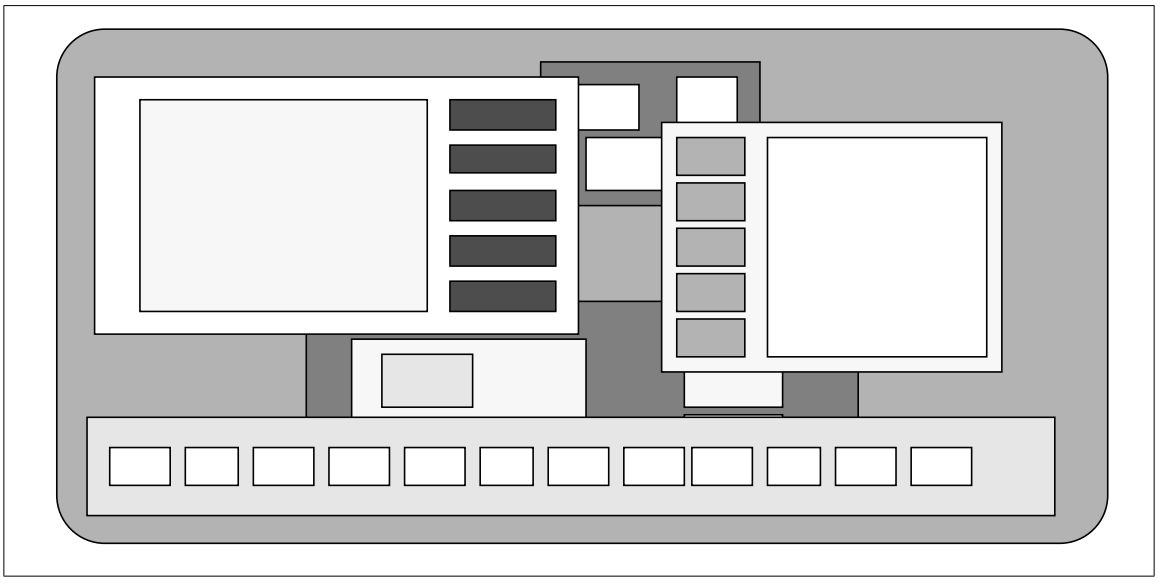

Abbildung 9-1: Graphik über Hierarchie von Rahmen

Die von den Rahmen belegte Fläche wird durch Angaben der linken unteren Koordinaten und ihrer Ausmaße in X- und Y- Richtung, also der Höhe und Breite, beschrieben. Eine Beschreibung anhand von zwei Koordinaten ist eine andere Möglichkeit. Dabei wirkt eine Verschiebung auf die Koordinate und eine Grössenänderung auf die Ausmaße. Dies bezieht sich auf Änderungen am Modell und nicht dessen unterschiedliche Ansichten. Es können noch Vergrößerungsfaktoren und unterschiedliche Ausschnitte der Ansicht hinzukommen.

9.1.2 Wahl der Koordinaten

Es erscheint dem Autor erforderlich die Koordinatenwahl bezüglich des Diskussionspunktes der Wahl von lokalen und globalen Koordinaten darzulegen. Unter den globalen Koordinaten werden diejenigen verstanden, welche sich auf den Ursprung (0,0) des Dokumentes beziehen. Der Richtungsinn ist dabei mathematisch definiert, daß heißt im kartesischen Koordinatensystemen von links nach rechts und von unten nach oben.

Die Wahl des Koordinatensystems ist in Systemen zur Steuerung von Oberflächen nicht immer eindeutig. Häufig wird mit Positionierungen ausgehend von einem Ursprung in der linken oberen Ecke gearbeitet. Von diesem Koordinatensystem wird wegen des nicht intuitiven Charakters abgesehen.

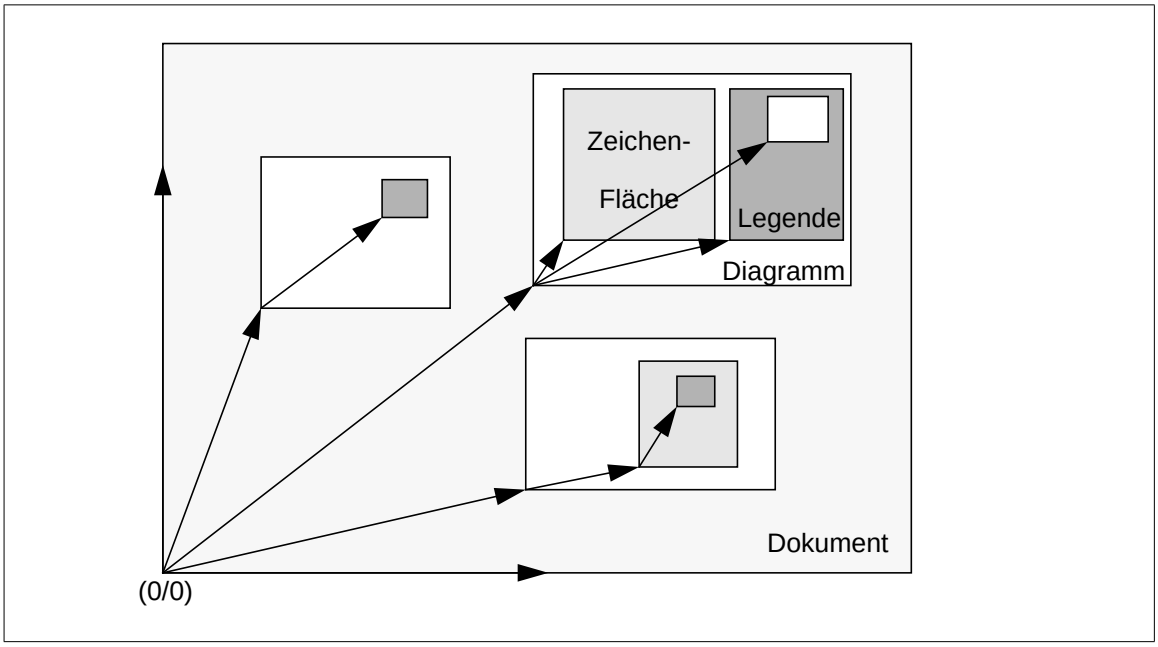

Abbildung 9-2: Lokale und globale Koordinaten

Bei interaktiver Positionierung, zum Beispiel über einen Manipulator, ist es leichter eine relative Positionierung bezüglich des umschließenden Rahmens der übergeordneten Struktur auszuführen. Zusätzlich besitzt jedes Modell eine Koordinate in Relation zum globalen Ursprung des Modelles. Alle übrigen Koordinaten werden relativ zum Ursprung der darüberliegenden Verwaltungstruktur gespeichert.

9.1.3 Geschachtelte Verwaltungsstrukturen: Container

Eine höhere Verwaltungsstruktur ist der sogenannte „Container“. In die Struktur lassen sich beliebige zu ihr passende Objekte „hineinwerfen“ und positionieren. Ein Container ist somit ein Rahmen mit Verwaltungsaufgaben über seine untergeordneten Objekte. Die zu einem Container passenden Objekte werden als seine Primitive oder

Basiskomponenten bezeichnet. Bei diesen kann es sich wiederum um Container einer untergeordneten Hierarchie handeln. Das heißt, es entsteht eine Hierarchie aus Containern. Zur Hierarchiebildung innerhalb eines Containers werden zusammenhängende Basisprimitive, auch „Composite“ genannt, eingesetzt.

9.1.4 Ansammlungen von Basiskomponenten: Composite

Aus den Basiskomponenten eines Containers lassen sich zusammengesetzten Ansammlungen oder Gruppen aufbauen, wobei diese Gruppen ihrerseits sich wiederum als Basiskomponenten darstellen. Die lokale Positionierung erfolgt immer bezüglich der übergeordneten Gruppe, und falls es sich bei der Gruppe um diejenige handelt, die dem Container zugeordnet ist, ist die Zuordnung bezüglich des Rahmen des Containers vorzunehmen. Jede Komponente kennt ihren zugeordneten Container und kann genau einem zur gleichen Zeit zugeordnet sein.

9.1.5 Operationen: Verschieben, Plazieren, Größenänderung

Für das Verständnis der Arbeitsweise von Visualisierungen werden jetzt die Basisoperationen auf Rahmen anhand der vorgestellten Struktur durchgespielt und dadurch veranschaulicht. Getroffene Entscheidungen werden durch die beschriebenen Operationen verständlicher, aber historisch gesehen entstanden sie im Rahmen der Arbeit in umgekehrter Reihenfolge. Oftmals wurde erst einmal vereinfacht mit anderen Ansätzen gearbeitet, wobei deren Erweiterung und Ausbau sich später unweigerlich als erforderlich erwies.

Verschieben von Basiskomponenten

Beim Verschieben von Primitiven werden zwei Fälle unterschieden. Ist das Objekt der Ankerpunkt der Verschiebung, bedeutet dies, das Objekt ist in der Hierarchie der größte Rahmen, der alle Objekte umfaßt, die es zu verschieben gilt. In diesem Fall werden sowohl die globalen als auch die lokalen Koordinaten angepaßt. Wird hingegen nur eine Basiskomponente als ein Bestandteil einer Gruppe oder eines gesamten Teilbaumes einer Hierarchie verschoben, wird in diesem Objekt nur die globale Koordinate angepaßt. Ein Anpassung seiner lokalen Koordinaten ist, da sich diese Relativkoordinate nicht ändern, nicht erforderlich. Wegen der dabei kurzzeitig innerhalb der Operation entstehenden Inkonsistenz zwischen internen Koordinaten in Bezug auf ihre lokalen und globalen Referenzen, muß die Operation atomar durchgeführt werden.

```
move ( delta:VECTOR )
     -- verschieben des aktuellen Objektes um Delta
     -- Objekt Composite ist Ansatzpunkt der Verschiebung
local
     a_composite:COMPOSITE
do
     move_global( delta );
     move_local( delta );

     from elements.start;
     until elements.forth;
     loop
          elements.item.move_global(delta)
     end;
end; -- move
```

Beispiele 9-1: Verschiebeoperation bei der Gruppe

Plazieren von Primitiven

Das Plazieren einer Komponente erfolgt über das Anpassen sowohl ihrer globalen als auch ihrer lokalen Koordinaten. Vorher muß das Primitiv der zugehörigen übergeordneten Gruppe zugeordnet und bei Wechsel der Gruppe aus der ausgehenden Gruppe entfernt werden. Ansonsten ist die Hierarchie der Verwaltung Inkonsistent da die Komponente dann mehreren Gruppen angehört.

Manipulation der Größe

Die Änderung der Größe von Basiskomponenten läßt sich im einfachsten Fall auf eine Änderung der Ausmaße zurückführen. Handelt es sich bei der Komponente um eine Gruppe oder gar um einen Teilbaum ausgehend von einer Gruppe muß anders vorgegangen werden, da dann zwischen Vergrößerung und Verkleinerung unterschieden werden muß.,

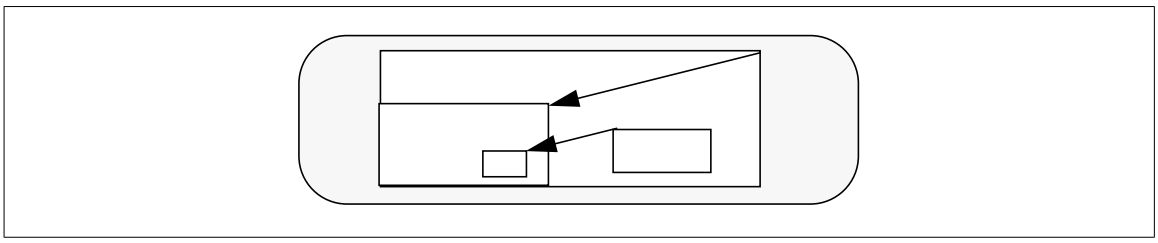

Abbildung 9-3: Verkleinerung von Objekten, zum Beispiel als Gruppe

Der Änderungsfaktor muß in diesem Fall durch den ganzen Baum propagiert werden. Da kein Element in seinen Ausmaßen die Ausmaße des Gruppenrahmens überschneiden darf, muß bei Vergrösserung immer zuerst die Gruppe und dann erst die zugeordneten Komponenten geändert werden. Wird eine Verkleinerung durchgeführt ist genau

in umgekehrter Reihenfolge vorzugehen, also zuerst verkleinern der Basiskomponenten und ihrer Koordinatenanteile mit anschließender Verkleinerung der Gruppenrahmen an sich.

Operation „Resize“

Die Operation „Resize“ verlangt sowohl Änderung der Größe als auch eine neue Positionierung der Koordinaten. Bei Vergrösserung wird zuerst der Rahmen vergrößert, danach werden alle Basiskomponenten vergrößert und anschließend positioniert. Bei der Verkleinerung wird genau umgekehrt vorgegangen, zuerst werden alle Posititionen und Größen der Basiskomponenten angepaßt und danach Rahmen verkleinert. Am Schluß erfolgt eine Reduzierung der Größe der Rahmengruppe. Da sich die Änderungen in jeder Komponentenrichtung des Koordinatensystemes unterschiedlich auswirken können, werden die Operationen komponentenweise durchgeführt.

Bei der Operation wird entweder eine Koordinate oder eine Gerade zwischen Rahmenkoordinaten, die eine gemeinsame Kante besitzen, festgehalten. Alle Koordinaten und Ausmaße ändern sich in Abhängigkeit der Dehnungsfaktoren in X- und Y- Richtung.

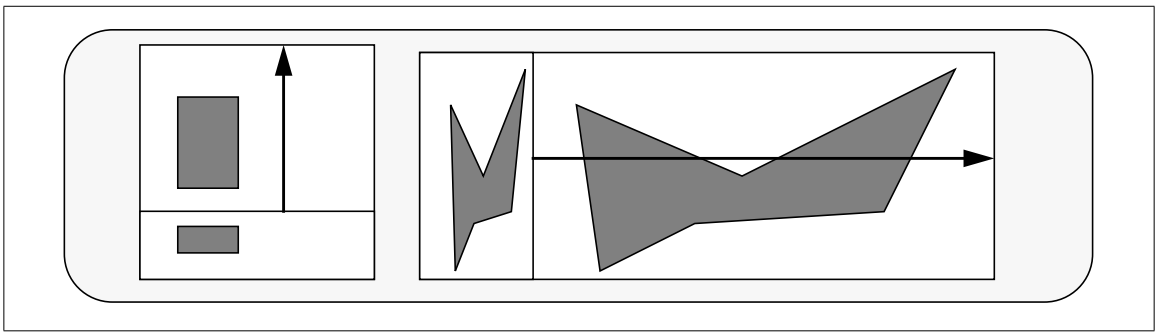

Abbildung 9-4: Operation Resize auf Gruppe

Für alle Koordinaten, die sich nicht auf dem festgehaltenen Punkt oder der Achse befinden, und bei der Positionierung verwendert werden, muß eine Umrechnung der Koordinaten mit den Skalierungsfaktoren durchgeführt werden. Die Ausmaße der untergeordneten Basiskomponenten werden entsprechend der Streckungfaktoren angepaßt. Mit den anderen Koordinaten wird analog vorgegangen. Die Dehnungsfaktoren müssen positiv sein, sind sie kleiner 1 wird verkleinert, sind sie grösser wird das Objekt gedehnt. Die Faktoren in X- und Y- Richtung sind disjunkt und von daher voneinander entkoppelt.

Der Spezialfall einer festgehaltenen Gerade läßt sich auf einen Dehnungsfaktor 1 in der korrespondierenden Richtung zurückführen. In diesem Fall bleiben alle Koordinaten und Ausmaße in Richtung der Senkrechten erhalten, in der anderen Richtung wird wie beschrieben vorgegangen.

```
resize( fix:COORDINATE; factor_x, factor_y: INTEGER )
     -- Ändert die Größe der Objekte
local
     a_composite:COMPOSITE
do
     -- ändert Koordinaten nur bei Komponente die Vergrößert
     change_global( 0, fix, factor_x, factor_y );
     change_local( 0, fix, factor_x, factor_y );
     change_other_attributes( 0, fix, factor_x, factor_y );

     from elements.start;
     until elements.forth;
     loop
          elements.item.resize( fix, factor_x, factor_y )
     end;

     -- ändert Attribute die Verkleinern.
     change_global( 1, fix, factor_x, factor_y );
     change_locals( 1, fix, factor_x, factor_y );
     change_other_attributes( 1, fix, factor_x, factor_y );
end; -- resize
```

Beispiele 9-2: Größenänderung bei Gruppe

Abhängig von der relativen Position der Achse oder des Fixpunktes muß die Komponente einer Koordinate entweder vergrößert oder verkleinert werden. Dies ist nicht zu verwechseln mit dem Verhalten, das bei Verkleinerung oder Vergrößerung analog in der Ansicht auftreten kann.

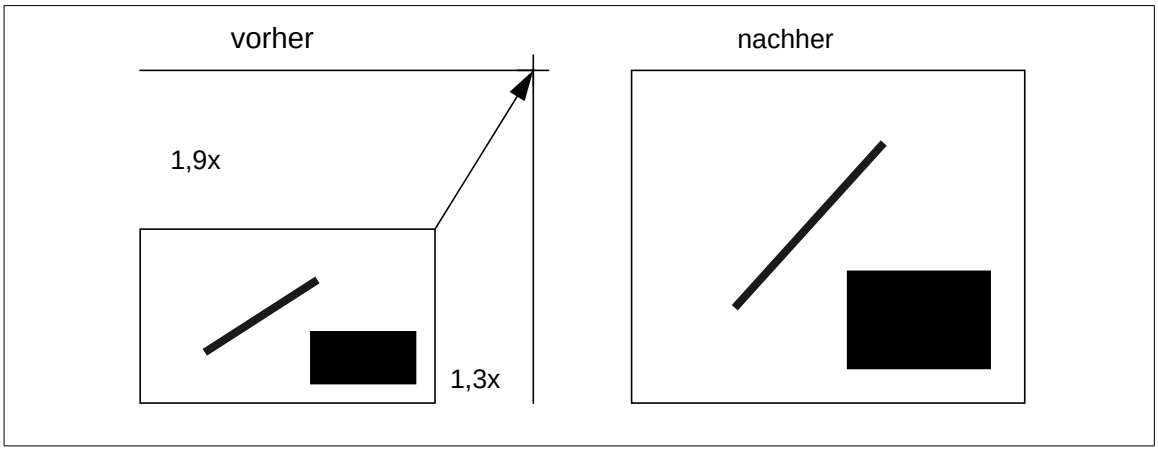

Abbildung 9-5: Vergrößerung einer Gruppe

Wird eine Längenangabe verwendet, die Anteile in beiden Richtungen aufweist, muß diese komponentenweise in Achsenrichtung zerlegt werden und anschließend einzelnen Komponenten angepaßt werden. In diesem Fall sollte die Angabe über die Beschreibung in Form von zwei Koordinaten durchgeführt werden.

Bei Komponenten, die in ihren Modellen auf Angaben mit Radien beruhen, tritt das Problem ungleicher Skalierungsfaktoren in beiden Achsenrichtungen des Koordinatensystemes auf. In diesem Fall sollte auf die elliptische Beschreibung ausgewichen werden. Wenn eine solche Darstellung zu umständlich oder kompliziert sein sollte, wird in diesem Bereich der kleinere Skalierungsfaktor der Operation verwendet.

9.2 Hierarchischer Aufbau

Aus Gründen der Übersichtlichkeit und einfachen Strukturierung wird eine Hierarchie aus mehreren Verwaltungsebenen aufgebaut. Diese Hierarchie findet sich sowohl in dem zugrundegelegten Modell als auch in den korrespondierenden, visualisierten Ausprägungen. Unterschiedliche Container bilden hierbei eine Hierarchie aus verschiedenen Containerklassen, innerhalb derer die Basiskomponenten ihrerseits wieder eine Hierarchie über „Composites“ aufbauen können.

9.2.1 Mechanismen zur Hierarchiebildung Composite - Container

Für die Erzeugung der Hierarchie werden zwei Ansätze gewählt. Zum einen die Containerstruktur, und zum anderen das zusammengesetzte Objekt aus Basiskomponenten. Der Container ist die Verwaltungsstruktur für seine Primitive, die Basiskomponenten. Der Container bedient sich als Ankerpunkt für seine Basiskomponenten eines „Composite“ für Primitive, dieses wird im folgenden Kontext auch als Hauptgruppe eines Containers bezeichnet. Die Primitive schachteln sich in Gruppen aus weiteren Primitiven und elementaren Primitiven. Das „Composite“ als Ansammlung von Basiskomponenten ist selbst eine Basiskomponente, dadurch entsteht eine Hierarchie der Basiskomponenten innnerhalb eines Containers.

Spezielle Komponenten können ihrerseits wieder die Funktion von Containerstrukturen übernehmen. Somit entstehen zwei ineinander verzahnte Hierarchien, die jeweils an spezielle Aufgaben gekoppelt werden. Es entsteht dadurch eine Hierarchie für die administrativen Aufgaben parallel zu einer Hierarchie, die der strukturellen Gliederung des Modelles für den visualisierten Aufbau entspricht.

Aufgabenteilung

Das „Composite“ wird verwendet um zusammengesetzte Objekte gleicher Basiskomponenten aufbauen zu können. So lassen sich zusammengesetzte Objekte, zum Beispiel Legenden bestehend aus Symbol, Text, Farbe aufbauen und als eine Komponente des Containers verstehen.

9.2.2 Prinzip der Hierarchie aus Verwaltungscontainern

In der vorliegenden Arbeit und dem verwendeten „Application Framework“ [Grh96][Grh96] wird die folgende Hierarchie aus Containerklassen verwendet. Jede dieser Containerverbände wird anhand seiner Aufgaben kurz skizziert.

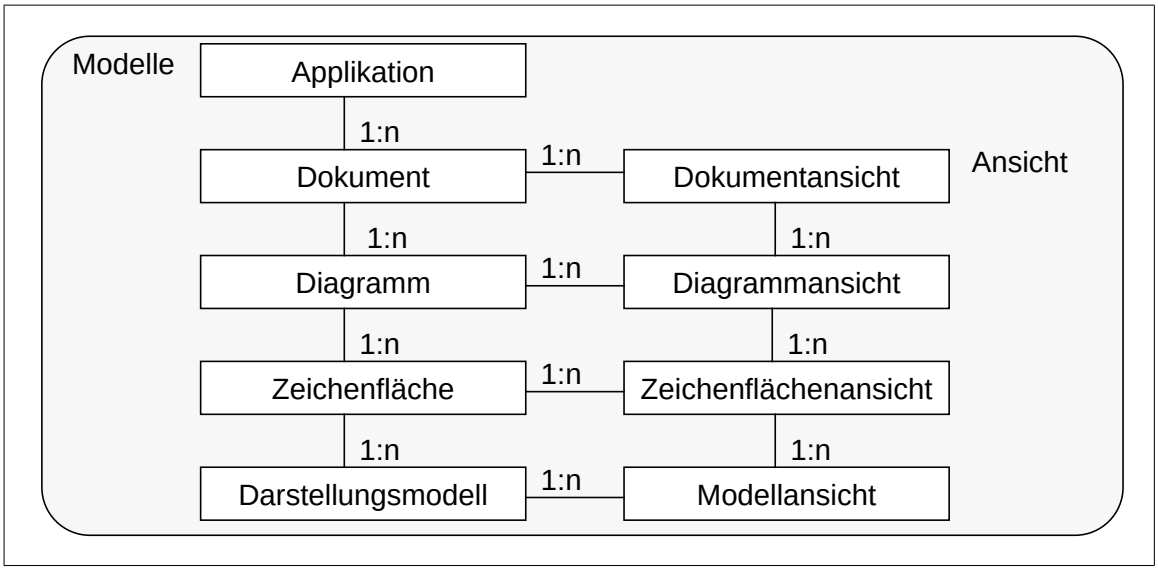

Abbildung 9-6: Containerhierarchie der Visualisierung

Applikation

Die oberste Containerklasse bildet die Applikation. Sie ist im Gegensatz zu den anderen Containern ohne korrespondierende Anteile, welche die Visualisierung realisieren. Die Applikation verwaltet eine beliebige Anzahl Dokumente und ist für die Voreinstellungen von Parametern für Basiskomponenten verantwortlich. Hier werden die Voreinstellungen von Dokumenten, ihre Größe und Anfangsausmaße, bei der Erzeugung einer neuen Ausprägung festgelegt.

Dokument

Das Dokument besteht aus einer beliebigen Anzahl zugeordneter Basiskomponenten. Alle spezialisierten Basiskomponenten erben daher von der verallgemeinerten Komponente SELECTABLE. Jede Basiskomponente des Dokumentes kann daher selektiert, verschoben und in ihrer Größe verändert werden. Genauso läßt sich auch eine Operation „Resize“ auf den Basiskomponenten durchführen. Die entsprechenden Methoden werden aus SELECTABLE geerbt und passend überlagert.

Zu jedem Dokument gehören mehrere Ansichten, die sogenannten „document views“. In den Dokumenten werden die aktuelle Größe und die Einheiten der Bemaßung abgelegt, wogegen in der visuellen Ausprägung der Dokumente die Skalierungsfaktoren und das aktuell genutzte Sichtfenster verwaltet werden.

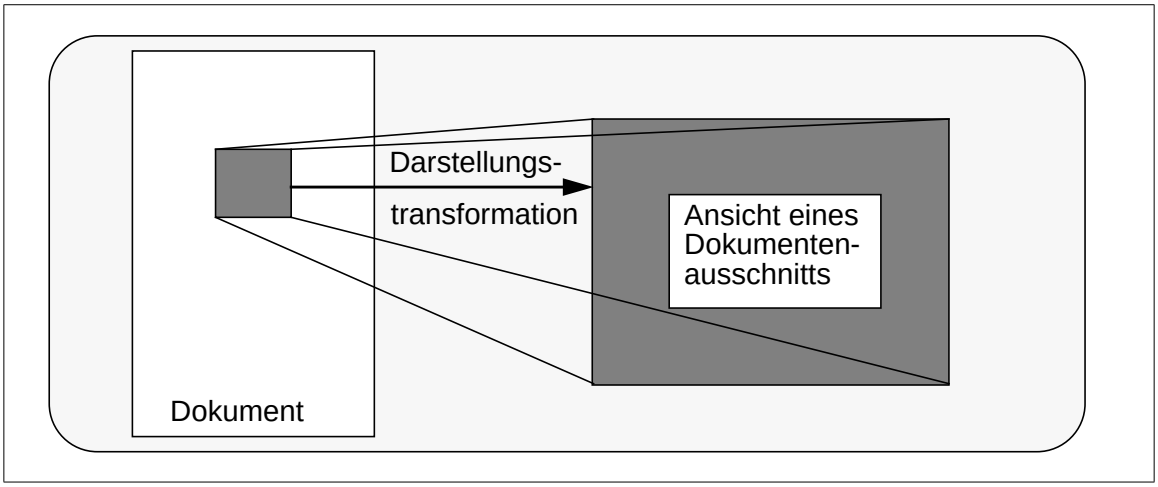

Abbildung 9-7: Das Dokument und seine Ansicht

Diagramm

Das Diagramm ist ein eigener Verwaltungscontainer, der aus der Basiskomponente des Dokumentes erbt. Das Modell des Diagrammcontainers ist gekoppelt mit seinen visuellen Repräsentationen der Abbildung als Diagramm.

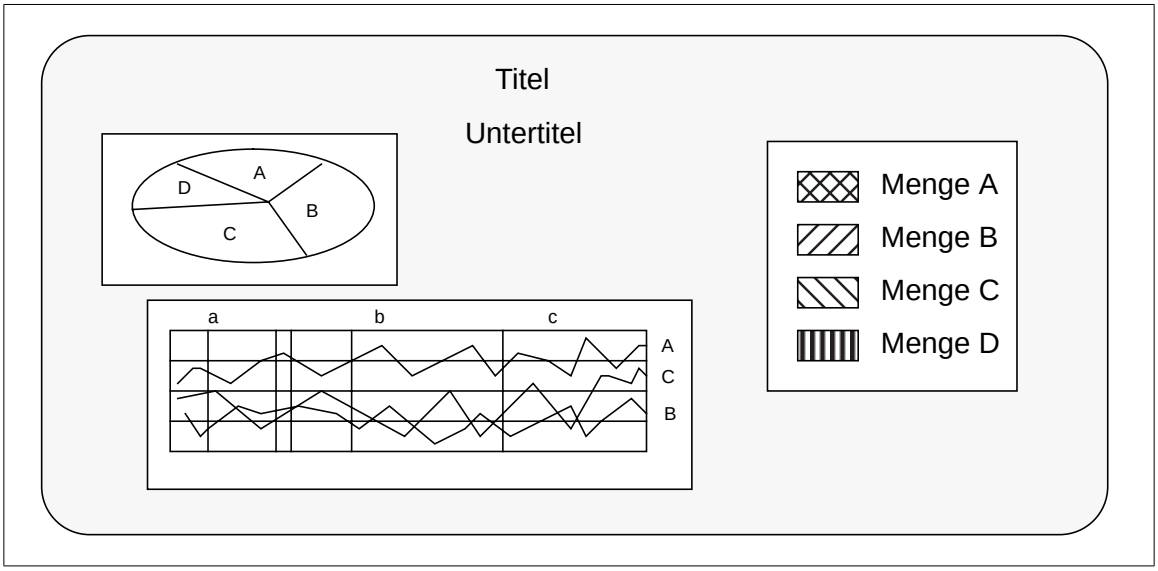

Abbildung 9-8: Aufbau von Diagrammen

Das *„Diagramm“* verwaltet Basiskomponenten des Diagrammcontainers, die zur Darstellung von Diagrammen oder Geschäftsgraphiken erforderlich sind, aber keine komplexeren Strukturen wie Achsen und Punktgraphiken aufnehmen können. Diese werden in einem eigenen Container, der sogenannten *„Zeichenfläche“* getrennt behandelt. So lassen sich mehrere graphische Auswertungen in ein und demselben Diagramm

durchführen, wie es zum Beispiel bei der Verwendung von Kuchengraphiken mit gleichzeitiger Darstellung von sechs Kuchendiagrammen in einem Diagramm der Fall ist .

Bei den „normalen“ Basiskomponenten handelt es sich um Texte, einfache Symbole wie Kreise, Sterne und regelmäßige Polygone und Legenden. Diese werden im Fall von Symbolen, die in *„Zeichenflächen“* auch als Marker eine Verwendung finden und aus gemeinsam genutzten Klassen abgeleitet sind, eingesetzt. Werden spezialisiertere Diagramme benötigt, wird von *„Diagramm“* geerbt und dessen Methoden überlagert.

Zeichenfläche

Die Zeichenfläche als Basiskomponente des Containers für Diagramme, bietet die Möglichkeit komplexere Strukturen wie Wertemengen zu modellieren und zu visualisieren. Ein Zeichenfläche bietet in ihrer Darstellungsfläche die Möglichkeit Graphiken, die auch aus einzelnen Punkten bestehen zu verwalten.

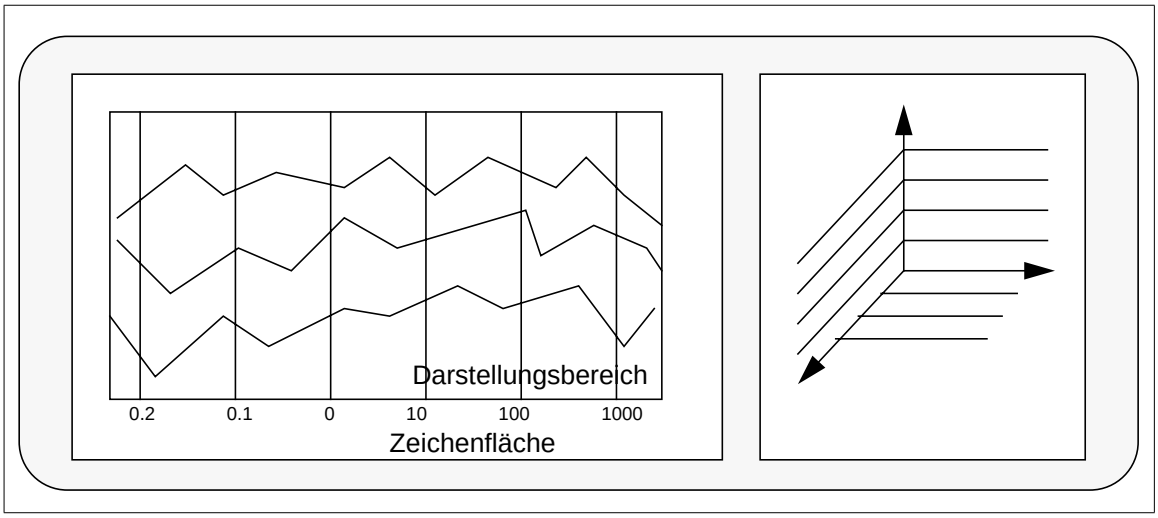

Abbildung 9-9: Zeichenfläche

Zusätzliche Basiskomponenten der Zeichenfläche sind der Darstellungsbereich, das darübergelegte Raster mit umschließender Beschriftung, die Darstellung von Achsen und deren Beschriftung, Ticks und Höhenlinien sowie Darstellungsmodelle der zugrundegelegten Wertemengen.

Darstellungsmodell

Als eine Art der Basiskomponente von Zeichenflächen fungieren sogenannte Darstellungsmodelle. Diese sind mit einer Ansammlung von Wertemengen verbunden. Die Darstellungsmodelle beschreiben die Art und Weise, wie mit einer Wertemenge umgegangen werden soll um diese zu interpretieren.

Hierunter wird zum Beispiel im Falle von Streudiagrammen verstanden, daß alle Werte des Intervalles [a,b] zusammengefaßt werden und jeweils Min/Max/Avg Werte bestimmt werden. Die Darstellungsmodelle sind also die Beschreibung zur Verwaltung der auferlegten Interpretation über den Wertemengen.

Diese Aufspaltung hat einen Vorteil gegenüber der Möglichkeit zweier unterschiedlicher Darstellungen verschiedener Diagrammart: Erstens lassen sich so zwei unterschiedliche Interpretationen der gleichen Wertemenge in der gleichen Darstellung anzeigen. Zum anderen wird bei unterschiedlicher Interpretation über einer Wertmenge einfach nur ein weiteres Darstellungsmodell geschrieben oder ein bestehendes erweitert.

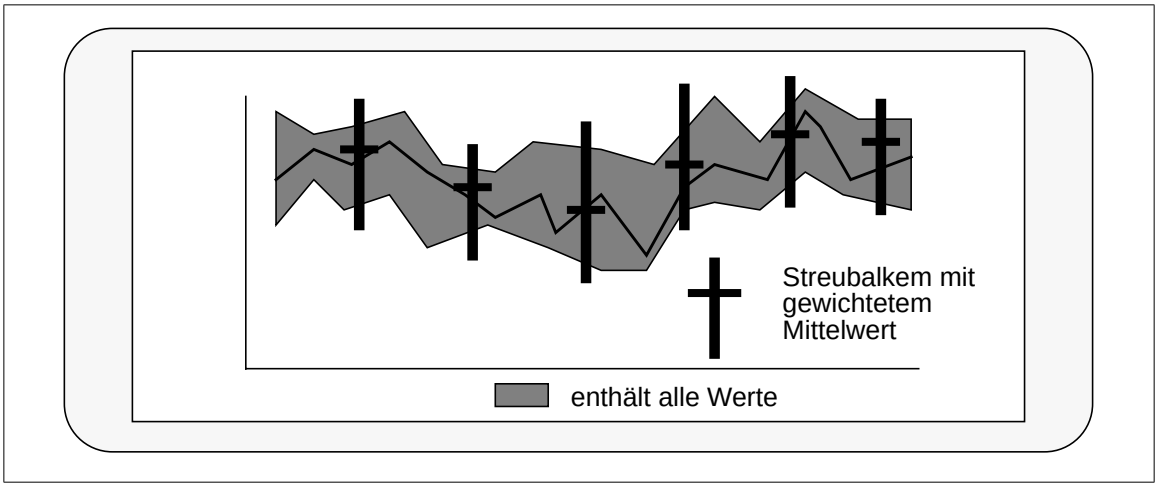

Abbildung 9-10: Verschiedene Darstellungsmodelle

9.2.3 Primitive, Composite und Container

Die Modelle für die Visualisierung basieren in erster Linie auf der Gliederung in Containerstrukturen, zusammengesetzte Gruppen und Basiskomponenten, sowie dem Zusammenspiel der einzelnen Bestandteile.

Primitive des Containers

Die Basiskomponente eines Containers ist in einer dem Container untergeordneten Gruppe aus zusammengesetzten Komponenten eingegliedert. Der Container hat die Aufgabe, die seine untergeordneten Basiskomponenten betreffen, zu synchronisieren und zu koordinieren. Der Container selbst benutzt als Verwaltungsstruktur die Gruppe aus Basiskomponenten als Einstiegspunkt für die Verwaltung der Basiskomponenten.

Zusammenspiel: Primitive - Gruppe

Unter Basiskomponenten, die aus nicht zusammengesetzten Komponenten bestehen werden sehr einfache Modelle verstanden. Dieses sind zum Beispiel einfache Symbole wie Sterne, Spiralen oder Texte. Um ein Verhalten wie bei einem Baukasten zu bekommen, müssen sich diese trivialen Objekte zu neuen Komponenten zusammenbauen lassen und die entstandenen Objekte dadurch ihrerseits wieder als Komponente des Baukastens zur allgemeinen Verfügung stehen.

Beispiele für die Bildung von zusammenhängenden Objekten sind einzelne Legenden bestehend aus Texten und Symbolen, aber auch alle Legenden einer Zeichenfläche zusammen.

Geschachtelte Gruppen

Die Möglichkeit der Bildung von Hierarchie innerhalb eines Containers über geschachtelten Gruppen erlaubt die Gestaltung übersichtlicher „Bäume“ mit einer überschaubaren Anzahl von „Knoten“ als Basiskomponenten dieser Ebene. Die Positionierung erfolgt immer relativ zur übergeordneten Gruppe. Wenn es sich bei der Gruppe um die Hauptgruppe des Containers handelt ist die globale Position der Gruppe identich zu derjenigen der Containerstruktur, die ja ihrerseits eine Basiskomponente der darüberliegenden Containerstruktur ist.

Bei geschachtelten Gruppen ist eine Verschiebung einfach zu realisieren. Die lokale Relativkoordinate zur darüberliegenden Gruppe wird angepaßt und die notwendigen Änderungen der globalen Koordinate werden durchgeführt und danach in der eigenen Hierarchie nach unten propagiert.

Verbindung Basiskomponenten - Container

Der Bezug zwischen den Basiskomponenten und ihrem direkt übergeordneten Verwaltungscontainer, besteht in der Notwendigkeit, Methoden die von ihrem Container bereitgestellt werden zu verwenden und Aktionen auf der Hierarchie aus Komponenten auszuführen. Die Beziehung ausgehend vom Container, besteht in einem Durchlaufen seiner untergeordneten Komponenten. Hierbei kann es es erforderlich sein, diese Operation sowohl in der Vorgehensweise der Tiefensuche als auch Breitensuche durchzuführen.

Aus dem Blickwinkel der Komponenten bieten sich zwei Möglichkeiten an, die Methoden des Containers auszuführen, zum einen über die Verwendung des Musters *„Chain of Responsibility“* zum anderen dadurch, daß die Eigenschaften des Containers als Referenzen miteingebunden sind. Um die dabei entstehende Redundanz etwas geringer zu halten, kann die reale Containerkennung auf die Basiskomponente des Types „Gruppe“ eingeschränkt werden. Will eine Basiskomponente Methoden ihrer Verwal-

tung verwenden, fragt sie in diesem Fall bei ihrer übergeordneten Gruppe an. Von dieser Möglichkeit wurde in der vorliegenden Arbeit wegen der dann entstehenden Uneinheitlichkeit der Basiskomponenten Abstand genommen.

9.3 Abbildung auf das Modell-View-Controller Prinzip

Die Verwendung eines Aufbaues nach dem Modell-View-Controller Ansatz erscheint auf den ersten Blick naheliegend und einfach. Aber wie in viele Bereichen steckt auch hier der Teufel im Detail. Nicht immer läßt sich die Frage beantworten in welchem Teil - ob Modell oder Ansicht -, der zu modellierende Aufgabenbereich, genau gehört. Vielmehr wandern mit steigender Spezialisierung der Diagrammdarstellung Anteile der Ansicht in das korrespondierende Modell. Sei es um Voreinstellungen der Ansicht festzulegen, oder weil die zugrundegelegte Interpretation einer Visualisierung durch ihren Spezialisierunggrad dem Modell sehr nahe kommt.

9.3.1 Problem der Aufgabenzuteilung

Die Schwierigkeit besteht darin genau zu trennen, welche Belange in das Modell gehören und welche den Ansichten zuzuordnen sind. Generell läßt sich sagen, daß die Belange, die für alle Ansichten gültig sind, Bestandteile des Modelles sein müssen. Wird dagegen die Frage gestellt, wie weit schlagen die Änderungen, die über Manipulatoren durchgeführt werden durch? Dann ist eine eindeutige Zuordnung einfach möglich. Das heißt, eine notwendige Trennung der Manipulatoren bedingt die entsprechende Aufspaltung der Ausprägung in Modell und View.

Das zweite Kriterium ist die Spezialisierung. Je spezialisierter ein Modell mit seinen Ansichten ist, desto mehr Anteile wandern von der Ansicht in das korrespondierende Modell. Das heißt, allgemein gilt je spezialisierter das Objekt, desto verwischter wird die Trennung zwischen Modell und Ansicht, da über die Spezialisierung eine engere Bindung zwischen beiden entsteht.

Eine anderer Ansatz für die Trennung zwischen Modell und „View“ ist die folgende Betrachtungsweise: Welche Interpretation soll mit der Wertemenge durchgeführt werden und welche Information wird dabei ausgelesen. Die Interpetation gehört in das Modell und die Fragestellung, wie läßt sich diese Interpretation unterschiedlich darstellen, gehört in die Ansicht.

Trennung der Manipulatoren bezogen auf Modell und Ansicht.

Prinzipiell ist bei der Möglichkeit von Manipulatoren anzuraten, die Trennung durch deren Definition festzulegen. Diese Vorgehensweise vereinfacht die Entscheidung, in welchen Bereichen zwischen Modell und View eine Trennung erfolgen soll, sehr.

Um ein Beispiel zu nennen: Bei der Darstellung von Streugraphen ist es erforderlich Intervallbereich festzulegen, die im Falle von Streubalken zu einem Darstellungsobjekt zusammengefaßt repräsentiert werden. In diesem Fall muß entschieden werden, ob das betrachtete Intervall in den unterschiedlich Ansichten verschieden sein darf oder nicht. Wenn nicht ist eine Speicherung im Modell angebracht, falls doch sollte die Verwaltung in der Ansicht erfolgen. Zu Zwecken der Initialisierung neuer Ansichten kann zusätzlich eine redundante Speicherung im Modell erforderlich sein.

Das Beispiel läßt sich analog auch auf andere Basiskomponenten wie Raster und Achsen ausdehnen. Da in der Ausarbeitung keine Manipulatoren angeboten werden, wird vom Ansatz her in Zweifelsfall sowohl im Modell als auch im View eine Verwaltung durchgeführt. Hierbei wird die Trennung der Attribute im Falle der Operation *„redraw“* am deutlichsten herauskristallisiert. Alle aus dem Modell ausgelesenen Attribute werden dort verwaltet, alle anderen in den korrespondierenden Ansichten.

9.4 Verwaltungsaufbau der Visualisierung

Der Verwaltungsaufbau der Visualisierung geschieht im wesentlichen über die Containerstrukturen, bisher wurde aber nur auf die Modelle eingegangen. Analog zur Hierarchie im Modell existiert natürlich auch eine Hierarchie der korrespondierenden Ansichten.

Die einfache Struktur des Nachrichtenkonzeptes im Observer Pattern ist nicht mehr eindeutig sobald eine Einbindung in Verbindung mit einer Hierarchie, sei es in Form geschachtelter, zusammenhängender Objekte oder als Hierarchie von Containerklassen, zustande kommt.

9.4.1 Zusammenspiel Modell - View

Das Zusammenspiel aus Modell und View wird durch die Hierarchie und das Observer Pattern abgebildet, zur genaueren Betrachtung ist der exakte Nachrichtenfluß der „Update“, „Notify“ und „Changed“ Nachrichten festzulegen. Wann und wie werden die Strukturen in den gekoppelten Hierarchien verständigt? Das heißt, wie sieht der Nachrichtenfluß in dem System genau aus?

Propagieren innerhalb der Verwaltungseinheit

Werden Änderungen innerhalb eines Teilbaumes des Verwaltungscontainers des Modelles durchgeführt, so werden diese von den Hauptgruppen oder den Containern selbst weitergeleitet. Zum Beispiel wird die Größe des Diagrammes geändert, werden die Änderungen im Modellbaum der Komponenten des Diagrammes propagiert.

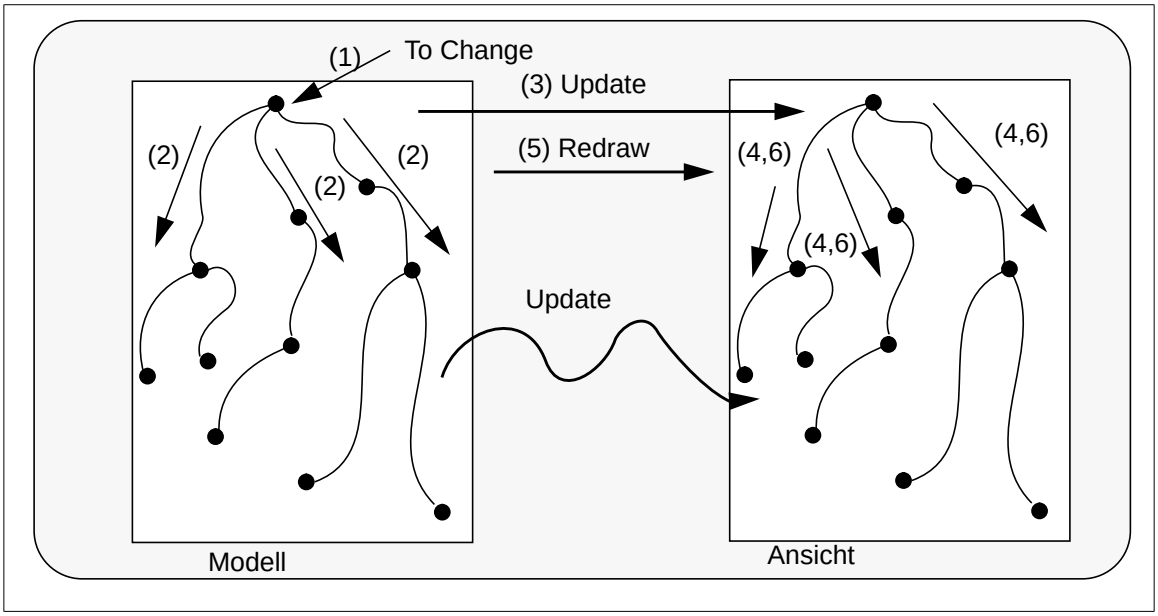

Abbildung 9-11: Nachrichten und Ausführungsfluß im System

Wurden alle untergeordneten Modelle angepaßt, sendet der höchste betroffene Container, in diesem Fall das Diagramm, allen angekoppelten Ansichten die Nachricht, das Änderungen im Modell vorgekommen sind. Diese Aufforderung zur Operation „Update“ wird in der Hierarchie für jede Ansicht innerhalb deren Teilbäume propagiert. Sind alle Änderungen in den Ansichten durchgeführt, wird eine zweite Welle von Nachrichten durch die Bäume der Ansichten propagiert, diese fordert das Neuzeichnen der Objekte.

Die Hierarchie, bestehend aus Komponenten im „View“ oder dem Modell, kann als Baum betrachtet werden. Hierbei sind alle Knoten „Composite“ Strukturen und alle Blätter des Baumes einfache Basiskomponenten.

```
redraw
     -- Zeichnen einer Komponete
do
     redraw_previous;
     redraw_self;
     redraw_post;
end; -- redraw
```

Beispiele 9-3: Redrawoperation einer Komponente

Die Durchlaufrichtung innerhalb der Hierarchie auf Seiten der Ansichten kann im Falle des „Redraws“ nicht als in-, prä- oder postfix bezeichnet werden. Zuerst wird der gesamte Hintergrund der Hauptgruppe gelöscht und der entsprechende Hintergrund gezeichnet. Als nächstes wird wie folgt vorgegangen: wenn die Komponente eine Gruppe ist werden vorab durchzuführende Zeichenoperationen ausgeführt, und

anschließend alle Einzelkomponenten der Gruppe neugezeichnet, am Schluß werden abschließende Operationen auf der Gruppe durchgeführt. Wenn es sich um einfache Komponenten handelt werden diese direkt gezeichnet. Das heißt im Falle der Basiskomponente „Gruppe“ werden Pre- uind Postoperationen auf dem Gruppenrahmen durchgeführt .

In einigen Gruppen des Teilbaumes kann es vorkommen, daß die Operationen die beim „redraw“ vor- und nachgeschaltet werden teilweise oder komplett entfallen. Aber es ist für jedes zusammengesetzte Objekt der Ansicht, beim Einbinden von eigenen Komponenten erforderlich, sich über diese Operationen Gedanken zu machen.

Die Reihenfolge der Basiselemente innerhalb der Gruppe auf Seiten der Ansicht repräsentiert die Umkehrung der Reihenfolge, in der die Elemente neu zu zeichnen sind. Die Reihenfolge innerhalb der Gruppe gibt implizit eine Zeichenreihenfolge der Bestandteile der Gruppe an, das heißt in der Verwaltung werden die Elemente von hinten nach vorne angeordnet und werden in dieser Reihenfolge aufgefordert sich neu zuzeichnen.

Analog läßt sich die Abbildung im Modell durchführen. Hier legt die Reihenfolge fest, in der die untergeordneten Komponenten bei Änderungen am Modell zu verständigen sind.

9.4.2 Diagramm, Zeichenfläche und Darstellungsmodelle

Als Verwaltungseinheiten in der Ansicht werden die Container der Strukturen Diagramme, Zeichenflächen und Darstellungsmodelle angeboten. Der Container für Diagramme setzt sich im wesentliche aus Titel, Untertitel, Legenden, Legendengruppen und Zeichenflächen zusammen. Bei der Zeichenfläche kommt zusätzlich noch die Achsen, deren Beschriftung und Raster sowie eine Anzahl von Darstellungsmodellen hinzu. Die Zeichenfläche muß sicherstellen, daß ihre Darstellungsmodelle und ihre Achsen samt Beschriftungen konsistent zueinander sind.

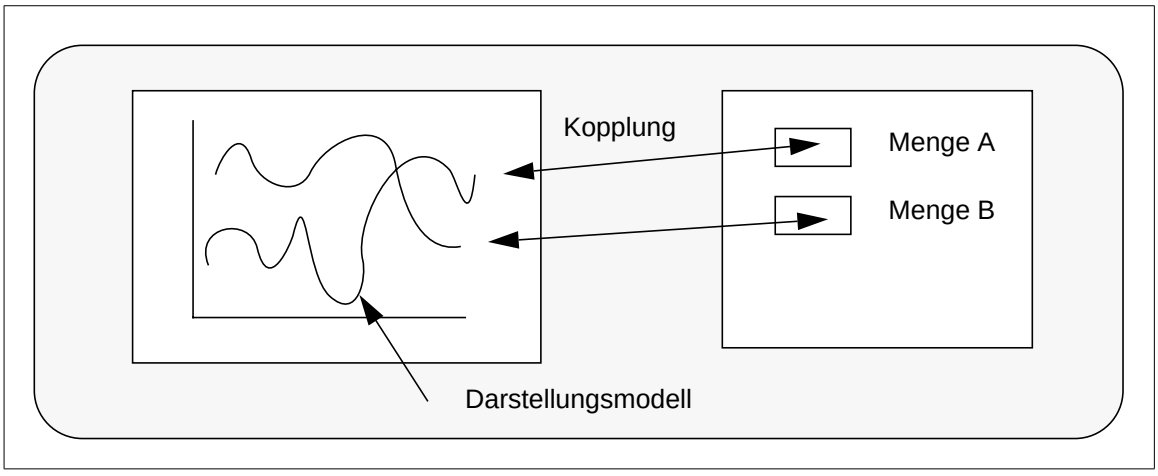

Abbildung 9-12: Kopplung zwischen Containern und Primitiven

Es ist unzulässig, daß drei Achsen bei Darstellungsmodellen von 2-dimensionale Wertemengen verwendet werden. Dies wird als Koordinierung der Belange zwischen den Basiskomponenten verstanden.

9.4.3 Gemeinsame Primitive

Zwischen den Objekten unterschiedlicher Container kann es im Bereich der Basiskomponenten Überschneidungen bei gemeinsam genutzten Basiskomponenten geben. Unter den Basiskomponenten deren gemeinsame Nutzung sowohl im Container *„Diagramm“* als auch *„Zeichenflächen“* bereitgestellt sind, werden die folgenden Darstellungen verstanden:

- Texte, zur Beschriftung mit Farbe, Font und Größe
- Graphische Basiselemente
  - Kreise, Kreisesegmente
  - Ellipsen und Ellipsensegmente
  - Regelmäßige Polygone und Rechtecke
- einfache Basissymbole
  - Sterne mit beliebiger Anzahl von Zacken
  - Kreuze und Linien

Wie aus dieser Aufzählung ersichtlich, werden diese Komponenten als Beschriftung, Symbole der Legende und als Marker in der Verwaltungsstruktur für Zeichenflächen eingesetzt. Eine Verwendung in anderen Bereichen könnte sich noch ergeben, soll aber hier nicht weiter betrachtet werden.

Eine gemeinsame Verwendung setzt voraus, daß die speziellen Basiskomponenten in diesen Fall die der Container „Diagramm“ und „Zeichenfläche“ von der gemeinsamen Basiskomponente erben.

9.5 Das Diagramm und seine Primitive

Der Container für Diagramme als Ausgangsklasse oder Ankerpunkt der Visualisierung von Wertemengen soll das Rahmengerüst für den Aufbau von Diagrammen bereitstellen.

Der allgemeine Aufbau ist wie folgt: Der Rahmen oder Container eines Diagrammes kann alle Komponenten enthalten, die für den Aufbau eines Diagrammes sinnvoll erscheinen können, also Legenden, Titel, Untertitel, beschreibende Texte und komplexere Basiskomponenten, die ihrerseits einen Container darstellen, wie Zeichenflächen.

In diesem Fall handelt es sich um die verallgemeinerte Struktur. Da es sich um eine Bibliothek handelt sind die jeweiligen Kopplungen zwischen den Komponenten nicht explizit definiert. Das konkrete Zusammenspiel zwischen den Basiskomponenten ist innerhalb der Bibliothek noch nicht festgelegt. Die hier geforderte Konkretisierung kann erst in der spezialisierten Form zum Beispiel bei Streudiagrammen erfolgen.

Man sollte sich immer vor Augen halten, daß es sich in diesem Fall um den Bau einer allgemein gültigen Biblioliothek handelt. Darauf aufsetzend werden spezialisiertere Bibliotheken wie zum Beispiel die für Streugraphiken eingeklinkt.

9.5.1 Aufbau der Verwaltung der Container für Diagramme

Der Aufbau ist eine Vereinigung des Entwurfsmusters „Composite“ für die Basiskomponenten des Containers „Diagramm“ und dem gleichzeitig eingebundenen „Observer-Pattern“ zwischen dem Modell und den Ansichten.

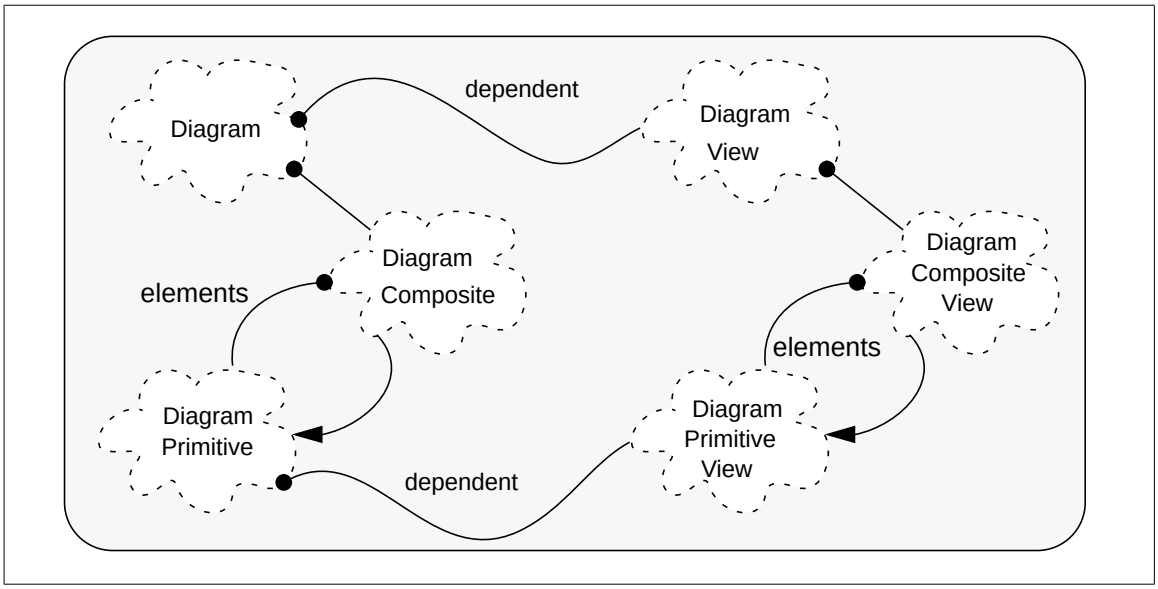

Abbildung 9-13: Diagramm und Basiskomponenten

Die Basiskomponenten, die gemeinsam mit anderen Containern benutzt werden erben natürlich aus den gemeinsam genutzten Klassen. Eine gemeinsame Klasse aus der alle Klassen erben ist der Rahmen. Die durch den Rahmen festgelegte Überdeckungsfläche impliziert die Zeichenreihenfolge in Z-Ordnung der Darstellungsobjekte.

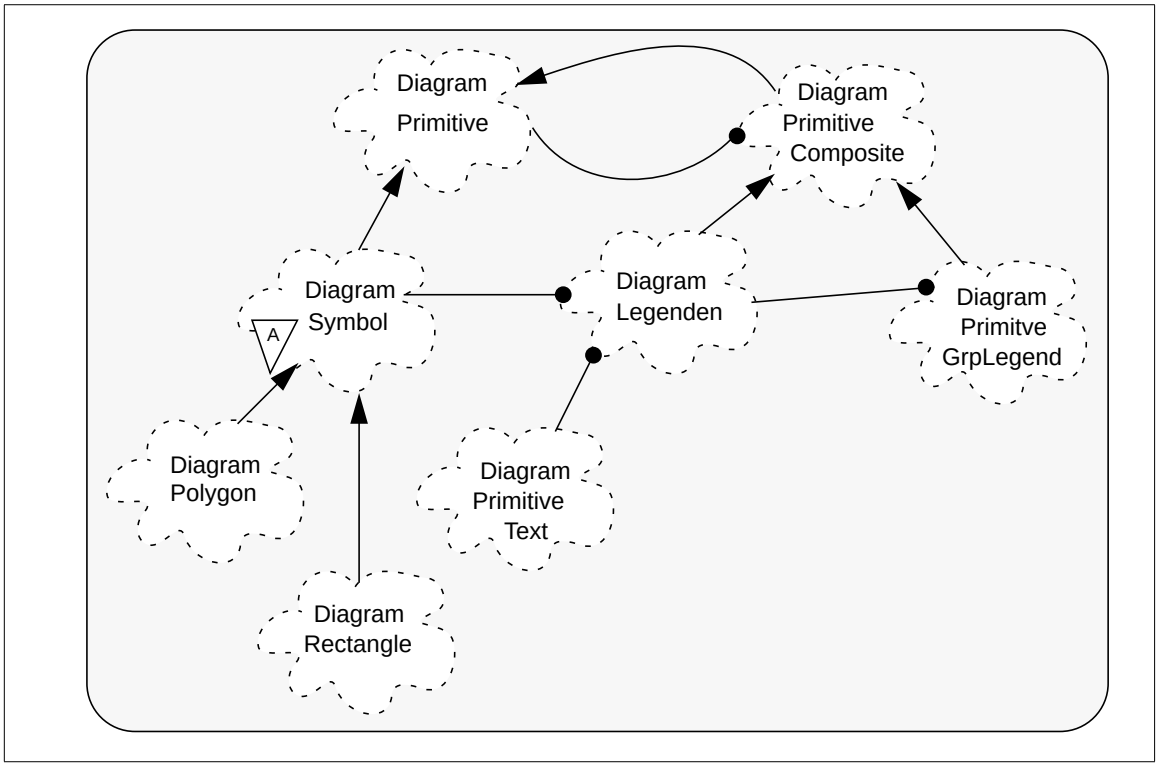

Abbildung 9-14: Basiskomponenten von Diagrammen

9.5.2 Aufgaben des Diagrammcontainers

Die Hauptaufgabe der Klasse „Diagramm“ besteht in der Verwendung als Ausgangspunkt für die Erzeugung neuer Diagramme. Wird zum Beispiel eine Ansicht des Diagrammes als Summendiagramm erzeugt, ist hierfür nur die erforderliche Zeichenfläche zu definieren, und in diese das Darstellungsmodell zur Auswertung zu integrieren.

9.5.3 Die Basiskomponenten und geschachtelte Gruppen

Die Kopplung zwischen Basiskomponenten und geschachtelten Gruppen basiert auf dem Entwurfsmuster „Composite“. In der Bibliothek wird hierbei die sehr verallgemeinerte Betrachtungsweise angenommen, daß ein Ausgangsobjekt eine rechtwinkelige Fläche besetzt und auf dieser dargestellt wird.

Ansonsten erfolgt keine Koordination zwischen den Komponenten. Auf der Fläche können weitere Objekte, beziehungweise deren zugrundegelegte Modelle, Darstellungsflächen belegen. Diese hierarchisch tiefer anzusiedelnden Objekte können ihrerseits wieder andere Darstellungsobjekte enthalten.

Einschränkend wird verlangt, daß keines der Objekte den Rahmen seiner übergeordneten Verwaltungstruktur oder Gruppe überschreitet. Das heißt, alle Komponenten einer Gruppe oder des Containers sind darstellungstechnisch komplett in dessen besetzten Fläche enthalten. Ein Löschen oder Überschreiben der Ansichten der Gruppe betrifft also automatisch alle ihr zugeordneten Teilbäume aus Basiskomponenten.

9.5.4 Zeichenflächen, Wertemengen und Darstellungsmodelle

In den folgenden Abschnitten und Kapiteln wird die Verwendung und Kopplung der einzelnen Container beschrieben, um die Arbeitsweise der Bibliothek zu veranschaulichen. Diese Container werden im Falle einer Spezialisierung natürlich den gewünschten Verhaltensweisen angepaßt. Im Gegensatz zur Verwaltungstruktur „Diagramm“ ist die „Zeichenfläche“, der Ansatzpunkt um eine Ansammlung von Wertemengen darzustellen.

Es wird nicht festgeschrieben, daß ein Diagramm nur eine Zeichenfläche besitzen darf und es ist möglich, daß ein Diagramm zwei Darstellungen verschiedener Wertemengen in unterschiedlichen Zeichenflächen visualisieren will. Wird die gleiche Art der Interpretation über den Wertemengen verwendet, kann es in der Spezialisierung zu Mehrdeutigkeiten kommen. Zum Beispiel lassen sich in einem Diagramm das zwei Kuchendiagramme darstellt, beide als einzelne Zeichenfläche mit ihren Darstellungsmodellen definieren oder in einer Fläche mit neuen Darstellungsmodell erzeugen. Eine allgemeine Regel und ein einheitlicher Ansatz um diese Mehrdeutigkeit zu vermeiden konnte im Rahmen der Arbeit nicht gefunden werden.

Jeder Zeichenfläche lassen sich unterschiedliche Interpretationen in Form von Darstellungsmodellen für Wertemengen zuordnen. Unter einem Darstellungmodell über Wertemengen, wird eine Interpretationsvorschrift verstanden, die festlegt was aus den zugeordneten Wertemengen heraus interpretiert werden soll. Hier wird also die Vorschrift festgelegt, was mit der Wertemenge getan werden soll, um eine Darstellung mit der auferlegten Bedeutung zu erhalten.

Ein Beispiel für eine solche Darstellungsvorschrift wäre, für jedes Intervall der Breite B die folgenden Werte zu bestimmen: Den Mittelwert und die Anzahl enthaltener Wertepaare der Wertemenge. Das heißt, das Modell enthält nur die Angaben Startwert, Endwert, Wert des arithmetischen Mittels und die Anzahl enthaltener Wertepaare. Diese berechneten Werte lassen sich zum Beispiel in Form von Punkten oder Kreisen unterschiedlicher Radien darstellen. Analog läßt sich die Vorgehensweise auf die Regressionsrechnung ausweiten, das heißt welche Intervallbreite soll bei jedem Regressionsansatz angesetzt werden oder wieviele Wertepaare sollen jeweils in einer Analyse zusammengefaßt werden. Jeder Wertemenge kann eine Legende zugeordnet sein, da zum Beispiel ein Marker Verwendung finden kann. Eine genauere Beschreibung des Darstellungsmodelles erfolgt in Kapitel 9.7 .

9.6 Die Zeichenfläche

Die Zeichenfläche und deren Basiskomponenten stellen die Bereiche der Modellierung von Interpretationsarten über Wertemengen dar. Der Container wird als Zeichenfläche bezeichnet, da im einfachsten Fall die Darstellung einer Punktewolke als Repräsentation der Wertemenge erfolgen kann. Das heißt, es können Vorschriften zur Erzeugung von Modellen existieren, die sich im Punkteraster der Modellauflösung bewegen, nicht zu verwechseln mit dem Auflösungsraster der aktuellen Ansicht.

Abbildung 9-15: Zeichenfläche und Darstellungsbereich

Genauso läßt sich ein Bereich definieren für den jeweils ein Bandbreite bestimmt wird, in der sich alle Wertepaare des betrachteten Intervalles befinden. Prinzipiell ist der Container im Diagramm als Verwaltungselement erforderlich um die Möglichkeit zu schaffen, mehrere Interpretationsarten über Wertemengen in einem Diagramm zu integrieren, ohne die dafür zugrundegelegte Bibliothek größtenteils wieder umschreiben zu müssen. Eine häufige Anwendung hierfür sind unterschiedliche Interpretationsarten über der gleichen Wertemenge um verschiedene Darstellungsarten im gleichen Diagramm zu visualisieren.

9.6.1 Unterschiede: Diagramm und Zeichenfläche

Die Container „Zeichenfläche“ und „Diagramm“ unterscheiden sich im wesentlichen durch ihren Kopplungsgrad. Während bei der Verwaltungsstruktur für Diagramme zwischen den Basiskomponenten des Containers eine relativ lose Kopplung in der Spezialisierung herzustellen ist, zum Beispiel zwischen Legenden und Bestandteilen von Zeichenflächen, besteht in der Zeichenfläche und den darin enthaltenen Darstellungsmodellen, also zwischen Wertebereich, Raster und der Verwendung von Achsen eine sehr enge Kopplung. Die Schwierigkeiten eine Abbildung zu erzeugen sind in diesem Fall erheblich größer, als im Falle der Verwaltungstruktur Diagramm.

9.6.2 Aufgabe des Containers der Zeichenfläche

Der Container für die Bestandteile der Zeichenfläche legt die Rahmenbedingungen für die Darstellung der Wertemengen fest. Hier wird die Größe des Darstellungsbereiches sowie dessen Raster, das heißt, wie genau das Modellraster ist, nicht zu Verwechseln mit der Rastergenauigkeit der eigentlichen Ansicht, festgelegt. Als Darstellungsbereich der Zeichenfläche wird derjenige Bereich verstanden, der von einer Zeichenfläche nach Abzug der durch Beschriftung und Rahmenachse verbleibenden Restfläche des kompletten Containers übrigbleibt.

9.6.3 Basiskomponenten der Zeichenfläche

Die Basiskomponenten der Zeichenfläche sind die gemeinsam genutzten Komponenten zur Beschriftung der Darstellungsfläche und die erweiterten Basiskomponenten.

9.6.4 Erweiterte Basiskomponenten

Bei den erweiterten Basiskomponenten handelt es sich um solche Modelle, die erstens eng gekoppelt sein können, und zum anderen eine aufwendige Ansicht und visuelles Verhalten aufweisen können und in ihrer Vielfaltigkeit größere Spielräume aufweisen können. Der Vorwurf bei diesen Objekten handele es sich doch um Visualisierung ist nicht von der Hand zuweisen, aber für diese Visualisierung müssen Modelle existieren, die das Verhalten grob modellieren.

Achsen, Ticks und Beschriftung

Die Modellierung von Achsen erfolgt in mehreren Stufen, jede Zeichenfläche kann maximal drei Achsen für die drei Dimensionen besitzen. Ist die Zeichenfläche nur für 2-dimensionale Wertemengen definiert, so reduziert sich dieses Verhalten um eine weitere Dimension. Die Achsen selber werden durch die Angabe ihres Ursprunges, ihrer Richtung, einer Gesamtlänge und eine Längenangabe in Bezug zu ihrem Ursprung, beschrieben.

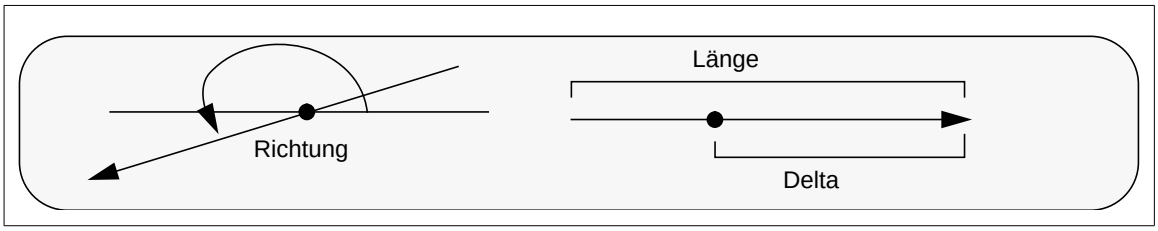

Abbildung 9-16: Modell der Achsen

Desweiteren ist zu definieren ob Ticks, Subticks und Mikroticks verwendet werden und wieviele dieser Sub- und/oder Mikroticks zwischen den Ticks existieren sollen, samt einer Berechnungsvorschrift wie ausgehend von einem Startwert im Ursprung die nächsten Ticks zu berechnen sind. Die Berechnungsvorschriften sind dabei linear, logarithmisch oder expotential.

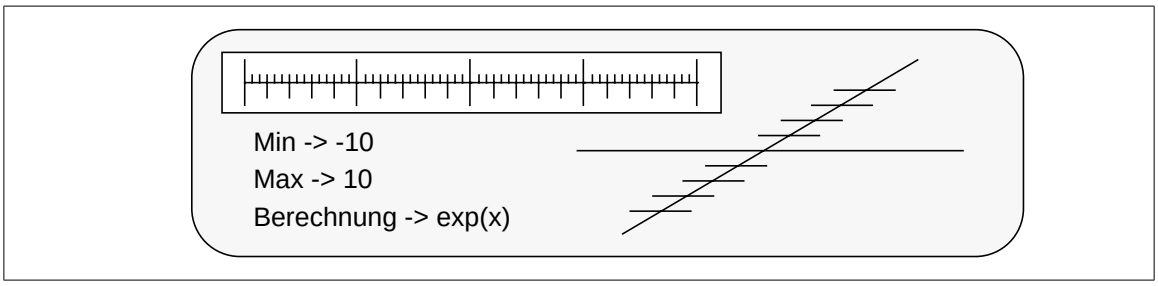

Abbildung 9-17: Modell der Ticks

Da die Achsen zueinander auch „schief“ liegen können und bei Projektion von 3 Dimensional auf eine 2-dimensionale Zeichenfläche sogar sein müssen, wird für alle Tickarten die Angabe einer Richtung sowie eines Teilungsfaktors möglich. Eine Beschriftung ist nur für die Ticks vorgesehen, hier werden Font, Farbe, Größe und Schriftrichtung samt Bezugspunkt festgelegt.

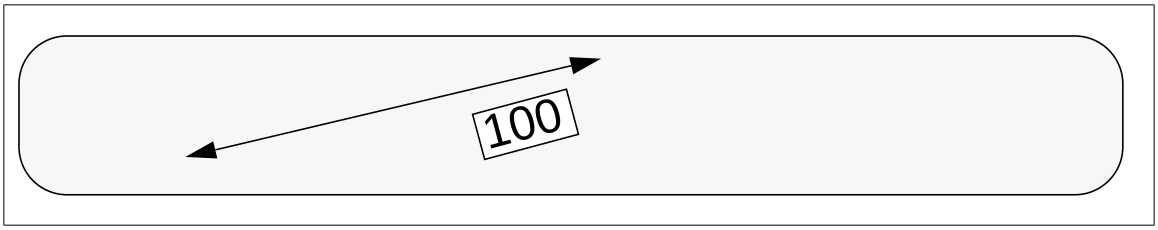

Abbildung 9-18: Achsenbeschriftungen und Richtungen

Rasterlinien

Bei den Rasterlinien wird zwischen zwei Arten von Rastern unterschieden. Die eine Form basiert auf der Darstellung von Achsen. Bei dieser Rastervariante handelt es sich um die „erweiterte“ Form eines Ticks. Hier werden ausgehend vom Ursprung in alle Achsenrichtungen jeweils nach Intervallen, die durch Berechnungsfunktionen ermittelt werden, in Richtung der korrespondierenden Achsen, Vektoren mit festgelegter maximaler Länge erzeugt.

Ob dieses Raster in einer Richtung nur an einer Stelle erfolgt, zum Beispiel in Höhe des Ursprunges, ist explizit zu definieren. Per Voreinstellung wird das Raster nur auf der Achse durch den Ursprung durchgeführt.

Bei 3-dimensionalen Wertemengen kann zusätzlich noch eine Schrittweite in der anderen Dimension definiert werden. Das Raster einer Achse wird also immer definiert über die Achsenrichtung, eine Gesamt- und Deltalänge in der die Achse geschnitten wird, sowie ob und wenn ja welche der beiden anderen Achsen mit welchem Inkrement verwendet werden soll, oder ob beide Achsen verwendet werden sollen.

Im Fall von drei Dimensionen wird bei Verwendung aller Achsen samt Raster, ein 3-dimensionales Raster aus Gitterlinien erzeugt. Dieses Raster bildet dabei einen schiefen Quader.

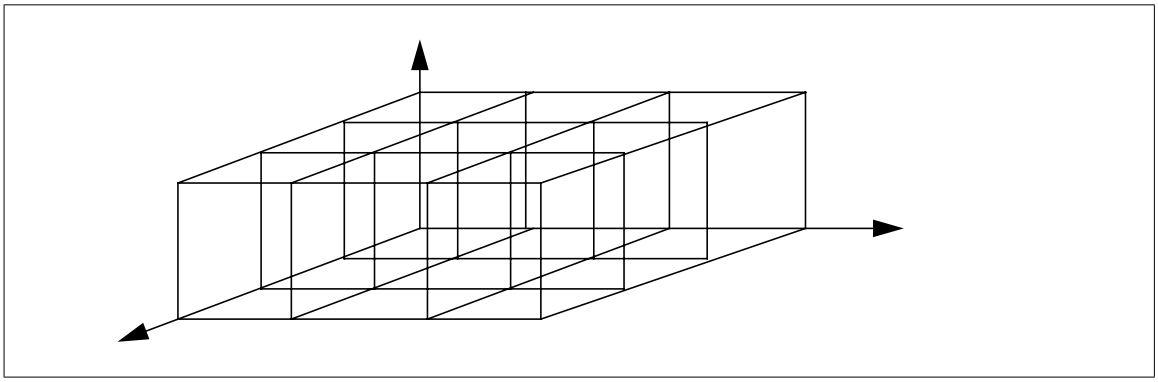

Abbildung 9-19: Rasterlinien

Die zweite Art Raster kann nur in der 2-dimensionale Darstellung verwendet werden, da in diesem Fall ein Raster an der Umrandung festgelegt werden kann. Dies geht aber nur, wenn die Rahmenachsen der gegenüberliegenden Seite die gleiche Beschriftung aufweist.

Für die Rahmenachsen gelten die gleichen Bedingungen wie für die Achsen an sich, auch hier lassen sich unterschiedliche Tickarten und Beschriftungen definieren. Zusätzlich gilt, daß Rahmenachsen immer senkrecht zueinander stehen müssen.

Darstellungsbereich

Der Darstellungsbereich der Zeichenfläche entspricht in vielen Fällen der Fläche, die der Container der Zeichenfläche in Anspruch nimmt. Nur wenn die Zeichenfläche zusätzlich Rahmenachsen samt Beschriftung verwendet, schrumpft der Darstellungsbereich um den entsprechenden Anteil.

Der Darstellungsbereich umfaßt die beliebig verlängerbaren Darstellungsanteile wie Achsen, Raster und andere Anteile, und beschränkt diese auf dessen maximale Ausdehnungsfläche. Punktewolken, Regressionsgraden und Streubalken dürfen außerhalb dieses Bereiches nicht dargestellt werden.

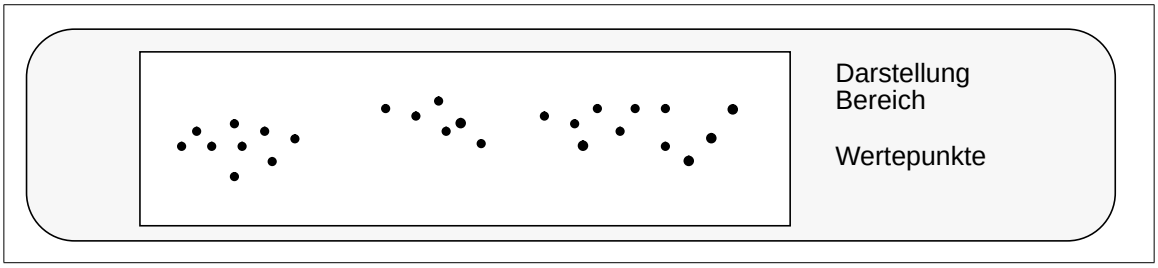

Abbildung 9-20: Darstellungsbereich

Darstellungsmodelle

Der Kernpunkt der Darstellung sind die Darstellungsmodelle unterschiedlicher Interpretationsarten. Hier wird ausgehend von den Wertemengen – ein Darstellungsmodell kann auch mehreren Wertemengen angekoppelt werden – festgelegt, wie aus der Wertemenge relevante Anteile der Interpretation extrahiert werden. Aus diesen wird dann durch Angabe des Ursprungs der Achsenrichtung und des entsprechenden Inkrementvektors eine neue Repräsentation gewonnen. Genaueres siehe unter Kapitel 9.7.

9.6.5 Zeichenfläche: Primitiven und Gruppen

Das Zusammenspiel zwischen den einzelnen „normalen“ Basiskomponenten und deren Gruppen ist analog zu den Basiskomponenten des Containers für Diagramme. Es lassen sich darin auch beliebige zusammengesetzte Objekte definieren. Die Schachtelung von Rahmenrastern und Achsen in einer untergeordneten Gruppe ist zwar möglich, aber semantisch nicht sinnvoll. Da es sich aber um eine allgemeine Bibliothek handelt lassen sich diese Unzulänglichkeiten nicht einfach abstellen. Verknüpfung dieser Art werden also nicht unterbunden, aber semantisch unsinnige Modelle erzeugen entsprechende Ansichten. Von einem Verbot über Invarianten indirekter Art wurde wegen der nicht auszuschließenden Möglichkeit einer sinnvollen Kopplung abgesehen und die Möglichkeit daher offen gehalten.

9.6.6 Arbeitsweise der Zeichenfläche

Werden in einer Zeichenfläche mehrere Darstellungsmodelle mit einer Anzahl von Wertemengen verwaltet und befinden sich Raster und Achsen in der Darstellung, so wächst das Problem der Zeichenreihenfolge.

Wenn die Modelle nur auf Funktionen und nicht auf Wertemengen, mit einzelnen Punkten als Wertepaare, eingeschränkt werden, dann läßt sich die Zeichenreihenfolge in der jeweiligen Ansicht aus der verwendeten Achsenansicht ermitteln. Danach wird von hinten nach vorne mit festen Inkrementwerten und den sich daraus ergebenden

Komponentenanteilen der jeweiligen Ausgangswerte, ein Zielwert ermittelt und dieser eingezeichnet. Die darunterliegende Fläche ist entsprechend zu löschen, da alle dort sichtbaren Anteile jetzt natürlich überdeckt werden.

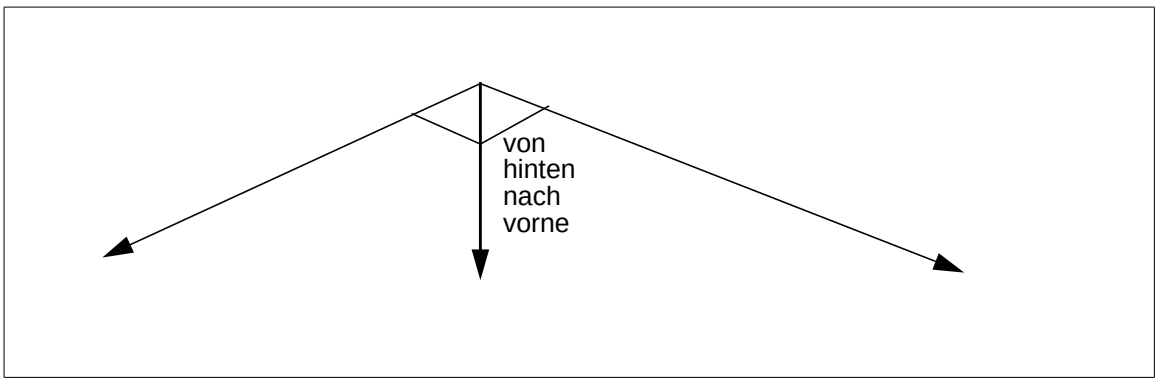

Abbildung 9-21: Zeichenreihenfolge

Sind die Wertepaare aber diskret als Einzelangabe vorhanden, so ist der Weg über die inkrementelle Generierung in der Vorzugsrichtung der jeweiligen Darstellung leider nicht möglich. Daher muß für jede Ansicht auf Seiten des Modelles ein abstrakter „View“ aus ausgewerteten Wertepaaren erzeugt werden, der in den Ansichten ein korrespondierendes Pendant besitzt, in welchem die Z-Ordnung Berücksichtigung findet.

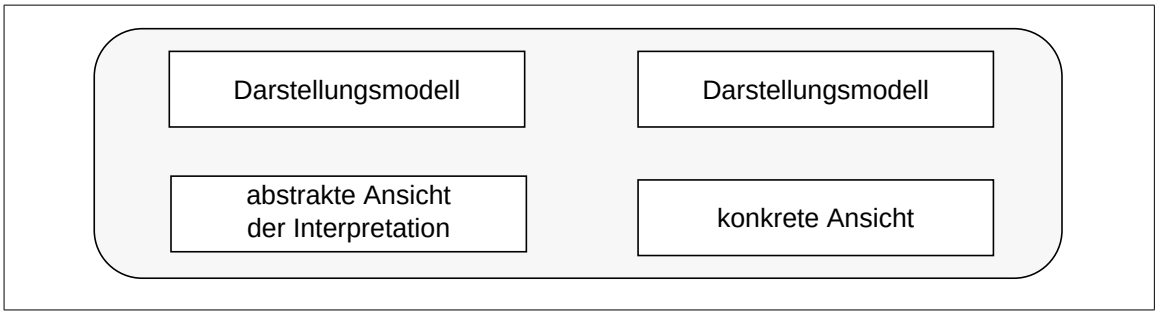

Abbildung 9-22: Abstraktes Modell des konkreten Views

9.7 Darstellungsmodelle für Wertemengen

Ein Hauptanliegen ist das Modell der Visualisierung möglichst ausbaufähig und offen zu gestalten. Es soll die Möglichkeit offengehalten werden, daß gleiche Wertemengen in einer Zeichenfläche auf unterschiedliche Weise dargestellt und verglichen werden, ohne explizit für jedes Diagramm oder Darstellungsmodell eine eigene Spezialisierung zu schreiben.

Dies ergibt bei n Modellen n!/((n-k)!k!) Möglichkeiten, diese miteinander zu verknüpfen. Selbst wenn nur ein Bruchteil dieser Verknüpfungen sinnvoll ist, so ist die Lösung nicht gerade elegant und ein Ausbau, um ein weiteres Modell erfordert die Kombinationsfähigkeit mit den bisher vorhandenen Darstellungsmodellen.

Bei einer Spezialisierung der Diagrammdarstellung wird es dem Anwender überlassen zu entscheiden, welche Kombinationen von Darstellungsmodellen sinnvoll sind und welche nicht. Werden mehrere Darstellungsmodelle miteinander in einer Zeichenfläche des Diagrammes kombiniert, so wird impliziert, daß die daraus resultierenden Visualisierungen untereinander verträglich sind.

9.7.1 Was ist ein Darstellungsmodell

Ein Darstellungsmodell ist eine Vorschrift wie die resultierenden Wertemengen zu interpretieren sind. Diese Vorschrift liegt in zwei Arten vor, solche die ihre Analyse genau über eine Wertemenge durchführen, und Modelle, die sich auf eine Ansammlung von Wertemengen beziehen.

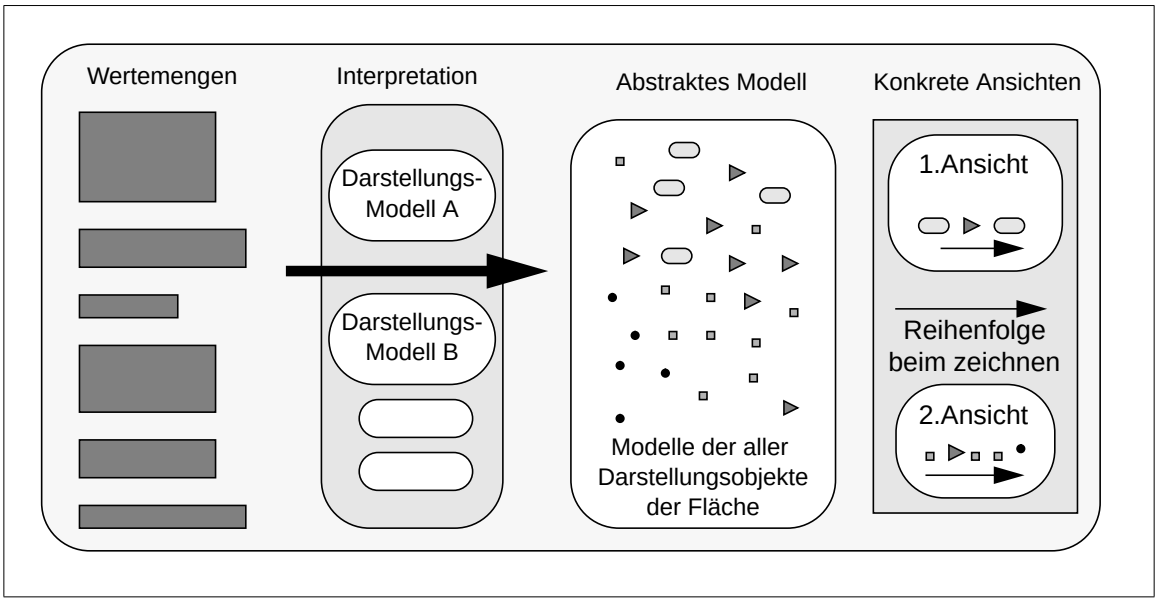

Abbildung 9-23: Zusammenspiel von Wertemenge bishin zu den Ansichten

Zum Beispiel ist dies der Fall, wenn Ertragswerte aus unterschiedlichen Abteilungen für Zeiträume eines Jahres erfaßt werden und mit denjenigen des Vorjahres verglichen werden. Natürlich lassen sich diese Darstellungen auch in eine neue gemeinsame Wertemenge überführen und auf dieser Menge dann ein zusätzliches Darstellungsmodell definieren. Da diese Vorgehensweise aber im allgemeinen relativ schwierig zu überschauen ist, werden Darstellungsmodelle, die auf Ansammlungen von Wertemengen arbeiten und solche die auf einzelnen Wertemengen operieren mit eingebunden.

9.7.2 Beispiele für Darstellungsmodelle

Einige Beispiele für Darstellungsmodelle sind Streudiagramme oder Intervallanwendung der Regression. Bei Streudiagrammen wird der Versatz zwischen den Balken oder die betrachtete Intervallbreite festgelegt. Es lassen sich die Intervalle aber auch über die Anzahl zu verwendender Tupel festlegen. In diesem Fall ist eine Bearbeitungsvorschrift festgelegt mit der die Werte des Intervalles zu ermitteln sind.

Oft handelt es sich dabei um Werte die das Maximum, Minimum und einen gewichteten Mittelwert bestimmen. Bei der Regression wird ein Bereich festgelegt für welchen jeweils die Regressionskonstanten ermittelt werden sollen, dann werden daraus die Darstellungspunkte anhand der Basisfunktionen ermittelt.

9.7.3 Darstellung der Modelle

Die Darstellung der Modelle geht jetzt wie folgt vor: Im 2-dimensionalen Fall werden die Ansichten anhand der Darstellungsmodelle berechnet, dabei werden die Berechnungen über Iteratoren auf den Wertemengen durchgeführt.

Da im Fall der 2-dimensionalen Darstellung auf eine Sortierung in Z-Ordnung der Ansichten verzichtet werden kann, spielt die Bearbeitungsreihenfolge der Wertemengen in ihren Darstellungsmodellen keine Rolle. Sukzessive wird zu jedem Wertepunkt der Wertemenge die Berechnung durchgeführt.

Da keine Überdeckung zwischen den Wertemengen aufgrund von „Tiefenwerten“ erfolgen kann, ist eine erweiterte Behandlung zur Berücksichtigung der Z-Ordnung nur im Falle von Diagrammdarstellungen für 3-dimensionale Darstellungen in Zeichenflächen erforderlich.

Darstellungskriterien bei 3-dimensionaler Darstellung

Bei der Darstellung von 3-dimensionale Wertemengen, ist zusätzlich zum einfachen Darstellungsmodell eine Überdeckung, die sich bei der Projektion vom 3-dimensionalen in den 2-dimensionalen Raum der Zeichenfläche ergeben kann, die sogenannte Z-Ordnung, zu berücksichtigen.

Um dies zu berücksichtigen ist bei der iterativen Auswertung über die Wertemengen eine Zwischenspeicherung aller durch die Dastellungsmodelle ermittelten Werte erforderlich. Diese Speicherung wird im folgenden Kontext als „abstraktes Modell“ der Ansicht bezeichnet. Zu jedem dieser Werte des „abstrakten Modells“ der Ansichten wird dann in jeder Ansicht ein „konkretes Modell“ ermittelt, das dessen Werte in der Z-Ordnung unter Verwendung der jeweiligen Betrachtungsrichtung festlegt. Zum Schluß wird die Ansicht von hinten nach vorne gezeichnet.

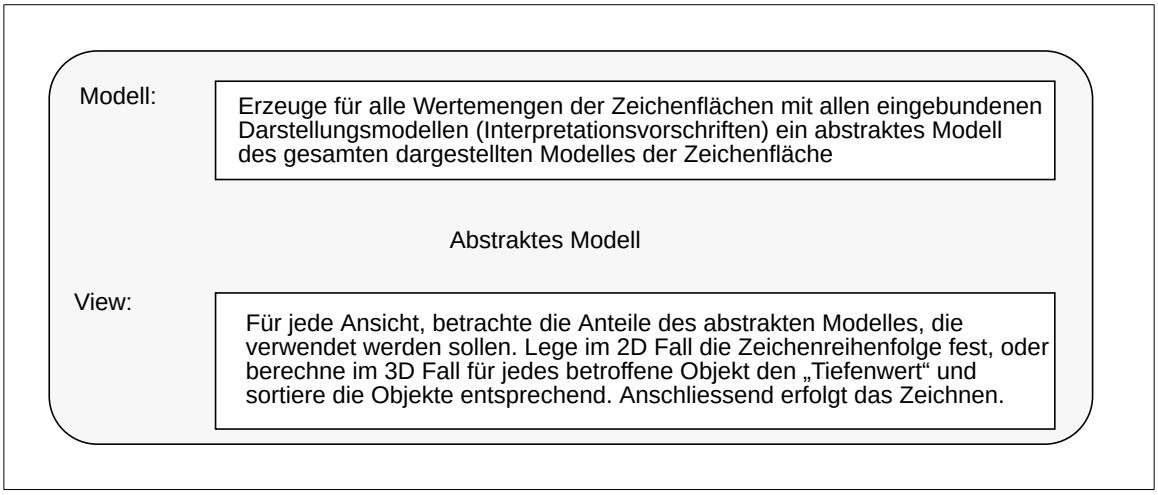

Abbildung 9-24: Zusätzliche Sortierung im 3-dimensionalen Raum

Im Falle der Darstellung von Funktionen läßt sich diese Vorgehen erheblich vereinfachen. In diesem Fall werden die Komponentenwerte von hinten nach vorne iterativ festgelegt. Dieser Bereich wird aus Gründen der Vollständigkeit und zur einheitlichen Gestaltung mit angeschnitten, eine Implementierung im Rahmen der Arbeit erfolgte nicht.

9.7.4 Abstraktes Modell der Ansichten

Das abstrakte Modell beschreibt was bei der Interpretation dargestellt werden soll und dient als Einstiegspunkt der unterschiedlichen Ansichten um ihre jeweilige visuelle Repräsentation zu generieren. Bei der Punktewolke kann es sich dabei um die gesamte Punktewolke handeln, im Falle von Streugraphen werden nur Minimum, Maximum und arithmetisches Mittel des betrachteten Intervalles berechnet und dort verfügbar gehalten.

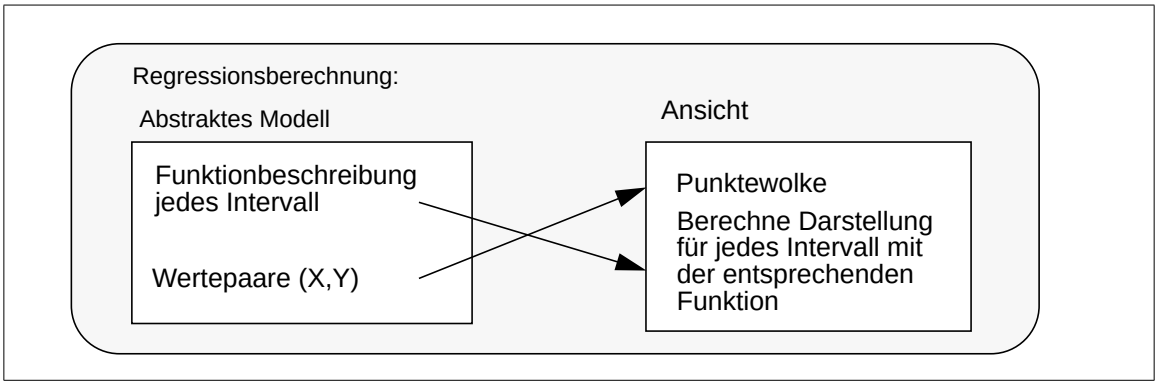

Abbildung 9-25: Beispiel Regressionsrechnung

Wichtig ist, daß hier alle Objekte auch aus unterschiedlichen Wertemengen gemeinsam betrachtet werden, um über allen jeweils in jeder Ansicht die Z-Ordnung zu berechnen. Diese weicht bei unterschiedlichen Betrachtungsrichtungen, die sich zukünftig über Manipulatoren verändern lassen werden, natürlich voneinander ab. Die jeweilige Ansicht definiert über ihren betrachteten Sichtbereich und den Zoomfaktor ihres Dokuments, welche Objekte des abstrakten Modelles überhaupt betrachtet werden müssen.

9.8 Baukastenprinzip

Wie aus der bisherigen Beschreibung ersichtlich ist, wird bei der Bibliothek großen Wert auf das Prinzip des Baukasten gelegt. Die Bibliothek soll so aufgebaut sein, daß sich damit eine evolutionär wachsende Bibliothek spezialisierter Diagramme bilden läßt, mit der ein Diagramm und Basiskomponenten zusammenfügt werden und deren Zusammenspiel untereinander festlegt wird. Man muß dazu nur ein Darstellungsmodell schreiben und die Methoden für das Hinzufügen neuer Wertemengen überlagern. Genauers wird in Kapitel 10 beschrieben.

9.8.1 Erstellung eines neuen Diagrammes

Bevor ein spezialisiertes Diagramm definiert wird, sind dessen Rahmenbedingungen festzulegen:

- Soll das Diagramm 2-dimensional oder 3-dimensional sein?
- Was soll dargestellt werden?
- Wie soll aus den Wertemengen anhand von Darstellungsmodellen eine Visualisierung erfolgen?

Man nehme ein Diagramm

Zuerst wird eine Spezialisierung der Diagrammklasse erzeugt. Diese erbt aus dem Container „Diagramm“. Als nächstes sollten Überlegungen zur Anzahl der zu verwendeten Zeichenflächen angestellt werden.

Bei Verwendung von mehreren Zeichenflächen bleibt die Frage offen, ob diese die gleiche Legendengruppe verwenden soll oder jeweils eine Gruppe für jede eingebundene Zeichenfläche verwendet werden soll. Dabei kann es bei gemeinsam genutzter Legendengruppe durchaus sein, daß einzelne Wertemengen nicht in allen Zeichenflächen vertreten sind.

9.9 Zusammenspiel zwischen Modell und Ansichten

Das wesentliche Kriterium im Zusammenspiel zwischen einem Modell und seinen korrespondierenden Ansichten ist die einfache Nachrichtenstruktur über das „Change-Update“ Protokoll und die strikte Trennung zwischen Modellbeschreibung und dessen Ansichten. Diese Trennung ist für Spezialisierungen mit möglichen Änderungen des Sichtwinkels vom Betrachter bei unterschiedlichen Ansichten nicht mehr zwingend erforderlich. Daher wird ein nicht unwesentlicher Anteil der Visualisierung, im Kontext auch als Interpretationsart bezeichnet, als Anteil des Modelles verstanden und in dieses integriert. Dieser Anteil wird als „abstrakte Ansicht“ bezeichnet und stellt quasi ein Zwischenglied zwischen den Bereichen dar.

Wird ein Modell verändert und sei es nur indem eine zusätzliche Wertemenge eingebunden wird, dann werden seine Anteile in der „abstrakten Ansicht“ berechnet und gespeichert. Danach werden alle Ansichten von der Änderung in Kenntnis gesetzt. Diese berechnen die jeweilige Z-Ordnung und erstellen daraus eine Reihenfolge aller zu zeichnenden Objekte der Ansicht. Am Schluß ruft die Zeichenfläche einen Operator auf, der über einen Iterator alle betroffenen Objekte im Vektor, der sich aus der Z-Ordnung ergibt, neu zeichnet.

9.10 Schnittstellen für Manipulatoren

Da in der Arbeit nicht die Implementierung von Manipulatoren erfolgte, sondern nur die zukünftige Verwendung solcher Beachtung findet, werden die Belange für die Erzeugung von Manipulatoren vorab berücksichtigt. Manipulatoren sind in diesem Falle Operatoren, die es ermöglichen, interaktive Änderungen an Objekten durchzuführen.

Bestes Beispiel ist eine Selektion mit anschließendem Dialog in Form eines „Popup“, so daß die entsprechenden Attribute mit eindeutigen Bezeichnern angezeigt und verändert werden können.

Eine Gestaltung für einen allgemeinen Entwurf von Manipulatoren erwies sich als sehr schwierig. Daher wird in diesem Fall, bei solch vielseitigen Objekten ein Ansatz über Hierarchiebildung zweckmäßiger.

Beispiel für einen solchen Ansatz ist die Verwendung von Notebooks zur Einstellung von Eigenschaften komplexer Objekte, wie dies unter Windows NT/95 und OS/2 Warp üblich ist. Wichtig ist in jedem Fall ein einheitliches Konzept, daß die Gestaltung solcher Manipulatoren schematisiert und einfach durchführbar gestaltet, so daß nicht in jedem Dialog andere Konventionen oder Namensgebungen verwendet werden.

9.10.1 Schnittstelle

Ein wichtiges Kriterium für die Gestaltung der Schnittstelle zwischen den Attributen und verwendeten Manipulatoren ist die einheitliche Namensgebung; das heißt, gleiche Bezeichner für Attribute und Methoden mit denen gearbeitet wird. Dies bedeutet nichts anderes, als Attribute, welche die Länge bezeichnen überall mit dem gleichen Bezeichner angesprochen werden und Methoden der gleichen Semantik verwenden. Dies sicher zu stellen, führt geradewegs in ein Design Dilemma, das sich nicht zufriedenstellend lösen läßt.

9.10.2 Konkurrierende Ziele: Objektorientierung und Schnittstelle

Eine einheitliche Schnittstelle für Manipulatoren führt zu einem Bruch im objektorientierten Design. Eigentlich sind die Attribute, Aspekte und Methoden so anzulegen, daß sie als Spezialisierung in eigene Klassen abgelegt werden. Alle Klassen, die diese Attribute verwenden, erben aus den spezialisierten Klassen der Attribute. Diesen wird zur Verdeutlichung ein „Has_“ vorgestellt, das heißt, beim Attribut „length“ der Fläche ergibt sich daraus ein „Has_Length“.

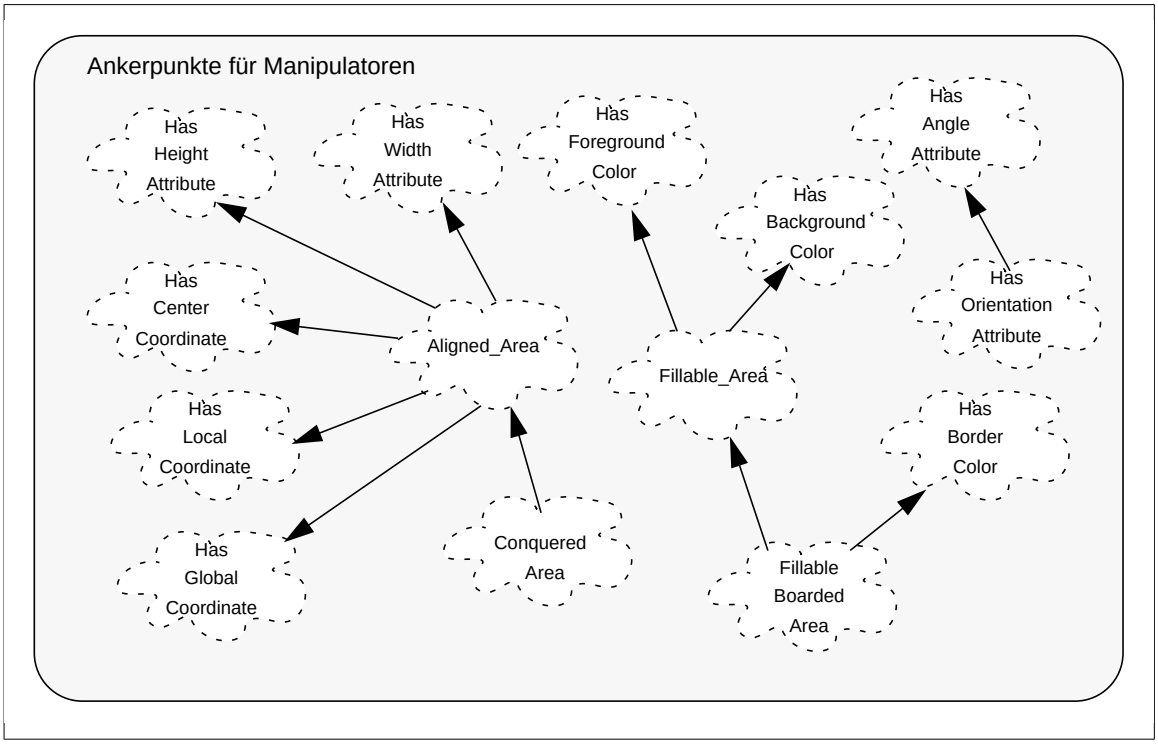

Abbildung 9-26: Verband zur Schnittstellenfestlegung für Manipulatoren

Der Bruch des Designs besteht darin, daß diese „Has_“-Beziehung in Klassen, die das entsprechende Attribut verwenden, geerbt und nicht einfach als „hat ein“ aus Sicht des objektorientierten Designs verwendte werden.

Dies ist aus Sicht der „reinen“ objektorientierten Lehre ein Bruch, da zum Beispiel „Fläche“ aus „Has_Length“ und „Has_Width“ erbt, keine „ist_ein“ Beziehung darstellt.

Der Vorteil liegt in der Wiederverwendung des Codes und des dadurch reduzierten Risikos von Fehlern durch immer wiederkehrende, einfache Attribute und deren Methoden. Der zweite Vorteil liegt natürlich darin, daß die Attribute und deren Methoden unabhängig voneinander nach außen sichtbar werden und so über Manipulatoren direkt abgreifbar sind. Jedem Attribut wird hierbei ein eindeutiger Bezeichner zugeordnet. Wird jetzt eine umrandete Fläche mit Vorder-, Hintergrund und Umrandungsfarbe benötigt so wird diese Fläche aus den fünf erforderlichen Klassen über erben der „Has-“ Struktur aufgebaut.

9.10.3 Konflikt zwischen der Manipulation von Modell und View

Ein bisher offengelassenes Problem ist die Art und Weise inwieweit Manipulatoren mit ihrer Durchschlagkraft wirken sollen. Dabei ist eine Trennung in Manipulatoren für das Modell und die jeweiligen Ansichten angebracht. Unter Manipulation des Modelles versteht der Autor die Festlegung abstrakter Koordinatenwerte, beschreibende Werte und Längen, ohne daß diese direkt mit der Ansicht etwas zu tun haben müssen. Bei der Verwendung von Achsen wird deren Anfangscharakteristik verstanden. Werden Drehungen auf einzelnen Achsen zugelassen trifft selbst das nicht mehr zu. Voraussetzung einer getrennten Betrachtung und Manipulation von Modell und View sind natürlich abhängig von der Bereitstellung zweier unterschiedlicher Arten der Selektion.

10
Kochrezept für den Diagrammbau

Wenn ausgehend vom Grundgerüst, das im Kapitel 9 vorgestellt worden ist, ein spezialisiertes Diagramm für eine spezielle Interpretationsart konstruiert werden soll, so muß die Vorgehensweise hierfür naheliegenderweise die im folgenden dargestellten Grundzüge aufweisen. Prinzipiell sollten alle Überlegungen zuerst am Modell ansetzen, anschließend sind alle Aspekte, die die unterschiedlichen Ansichten betreffen zu überdenken.

10.1 Grundüberlegungen

Der Grundgedanke bei der Erzeugung „neuer“ Diagramme sollte sein möglichst viele Bestandteile aus bereits bestehenden Diagrammen zu verwenden. Dabei sollen die Bestandteile des neuen Diagrammes, die noch nicht existieren ausgehend von den Grundkomponenten der allgemeinsten Bibliothek spezialisiert und erweitert werden. Dies bedeutet eine konsequente Anwendung des Aspektes der Wiederverwertung von Code.

Die Grundüberlegungen müssen daher immer von einem Fragekatalog ausgehen:

- Welche Informationen soll das Diagramm liefern ?
- Welche unterschiedlichen Arten der Visualisierung sollen angeboten werden.
    - Welche Informationen sollen aus welchen Ausgangsdaten extrahiert werden und wie lassen sich diese geeignet visualisieren? Überlegungen hierzu implizieren die Festlegung des zugrunde gelegten Darstellungsmodells.

Anschließend sollten Fragen über den strukturellen Aufbau des Diagrammes gestellt werden:

- Welche Hauptbestandteile in Form von Basiskomponenten werden dem Diagramm zugeordnet?
    - Wieviele Zeichenflächen sind vorgesehen und wieviele Legendengruppen sollen verwendet werden?
    - Werden von Zeichenflächen gemeinsame Legendengruppen verwendet ?

  - Die Relation zwischen den Basiskomponenten „Zeichenfläche“ und „Legendengruppe“ ist in vielen Diagrammen 1:1. Einige Diagrammarten verwenden bei vergleichender Darstellung 1:n. Die Exoten unter den Diagrammen verwenden sogar die Relation n:m, wobei natürlich die Anzahl der Legendengruppen geringer als die der Zeichenflächen sein muß.

- Welche einschränkenden Eigenschaften müssen die Wertemengen aufweisen und wie werden diese den Zeichenflächen zugeordnet?

  - Zu beachten ist, daß die Zeichenflächen Restriktionen in Bezug auf das Einfügen von neuen Wertemengen festlegen, weil Kombinationen zwischen diesen Wertemengen und damit gekoppelten Darstellungsmodellen mit den Eigenschaften der Zeichenfläche unverträglich sind.
    Zum Bespiel wenn die Darstellungsmodelle nur in einem Intervall eindeutig bestimmt sind wie von 0 bis 360.

  - Beispielweise ist das Einfügen von 2- und 3- dimensionalen Wertemengen in eine gemeinsame Zeichenfläche zulässig, wenn das resultierende abstrakte Modell der Ansicht typverträglich zur Zeichenfläche ist. Die Darstellungsmodelle über den Wertemengen beider Dimensionsgrade erzeugen also Anteile im abstrakten Modell, die sich mit fundamentalen Eigenschaften der Zeichenfläche vertragen.

  - Werden mehrere Zeichenflächen verwendet muß per Voreinstellung geregelt werden, welchen Zeichenflächen die Wertemenge zuzuordnen ist. Diese Zuordnung kann im einfachsten Fall über die Anzahl von Dimensionen erfolgen.

- Wieviele Darstellungsmodelle werden jeweils den Zeichenflächen angebunden und von welcher Art sind sie?

  - Wie wird die Kopplung zwischen Darstellungsmodell und Wertemenge bei neueingefügten Mengen festgelegt. Gilt die Verbindung zwischen neueingefügten Wertemengen für alle Darstellungsmodelle der Zeichenfläche oder nur für vereinzelte Darstellungsmodelle?

- Werden Titel und Untertitel des Diagrammes verwendet?

Steht die Entscheidung in Bezug auf die verlangte Interpretation und ihrer Visualisierung fest, sollte in der „Top down“ Strategie die Hierarchie auf Modellseite durchlaufen und dabei auf mögliche Wiederverwendung untersucht werden. Das Ziel des Vorgehens soll sein, bei der Untersuchung der allgemeinen und den aufsetzender spezialisierten Bibliotheken herauszufinden, welcher Anteil des Diagrammes geerbt, gekauft oder neu implementiert werden muß.

Angefangen wird die Suche bei den spezialisierten Bibliotheken. Hier sollte versucht werden verwendbare Komponenten aufzufinden. Der Ausgangspunkt ist der Verwaltungscontainer für Diagramme. Vielleicht existiert dort bereits eine Repräsentation

für das Diagramm oder zumindestens Anteile des Diagrammes lassen sich wiederverwenden. Mit viel Glück findet sich ein Diagramm, das genau die gewünschte Auswertung im Modell bereitstellt, so daß nur die angebotenen Ansichten modifiziert werden müssen. Das heißt, bei gefundener Übereinstimmung im Modell wird übergegangen zu den korrespondierenden Ansichten des Modells in der Bibliothek.

Nach erfolgter Untersuchung der Ansichten und der Feststellung, daß Änderungen im Bereich der Ansicht erforderlich und notwendig sein können, bieten sich hierfür unterschiedliche Strategien an. Erstens, die Ansichten sind typverträglich zu den neu zuerzeugenden Ansichten oder zweitens bestehender Ansichten sind nicht verträglich mit Ansichten des neuen Diagramm.

Bei Verwendung aus spezialisierten Bibliotheken wendet die Einbindung der neuen Anteile eher eine der beiden folgenden Strategien an. Im ersten Fall wird versucht bestehende Konkretisierungen auf Seiten der Ansicht zu erweitern. Das bestehende Modelle der Bibliothek wird verwendet und auf der Seite der Ansichten um weitere konkrete „visuelle“ Anteile erweitert.

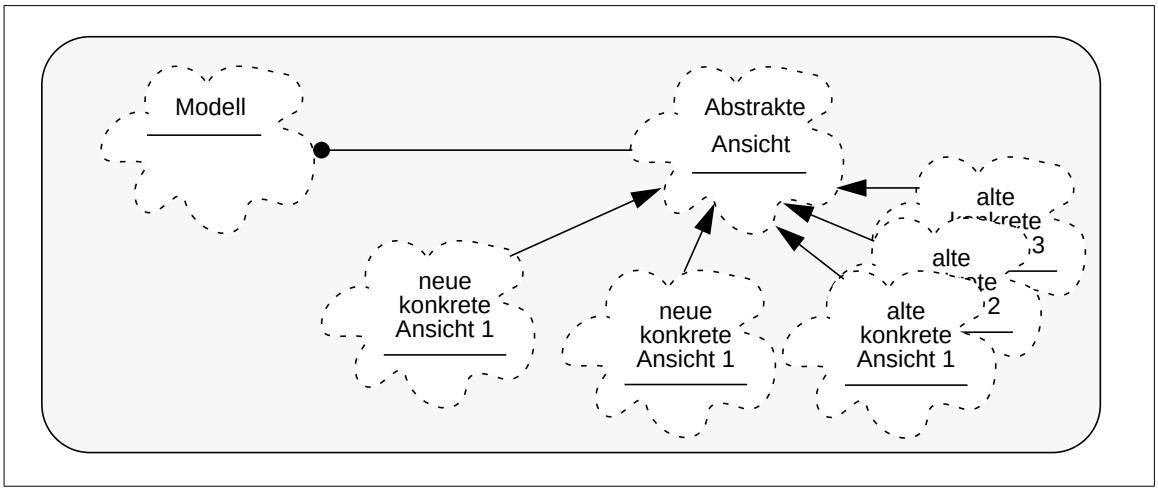

Abbildung 10-1: Modell ist um weitere Ansichten erweiterbar

Diese Möglichkeit birgt aber die Gefahr ungewollter Seiteneffekte, wenn die Ansichten nicht ganz verträglich sind und dieser Unterschied erst in Anwendungen basierend auf der Bibliothek auftritt.

Im zweiten Fall sind einige Ansichten des alten Modelles nicht verträglich mit Anforderungen des neuen Modelles für Diagramme. Daraus folgt, daß sowohl das Modell als auch die Ansicht des neuen Diagrammes ausgehend von existierenden Anteilen der Bibliothek spezialisiert werden. Im Zweifelfall ist der 2. Variante der Vorzug zu geben, da dieses Vorgehen typsicherer ist.

Der Aufwand für diese Art des Einbindens von neuen Komponenten ist natürlich höher und die Mechanismen des Vererbens sind in diesem Fall schwieriger auszuführen. Es besteht hier die Möglichkeit sogenannte „Rauten“ in der Vererbungshierarchie zu erzeugen, da sowohl von einer konkreten Klasse als ihrer abstrakten Vorgängerklasse geerbt werden kann.

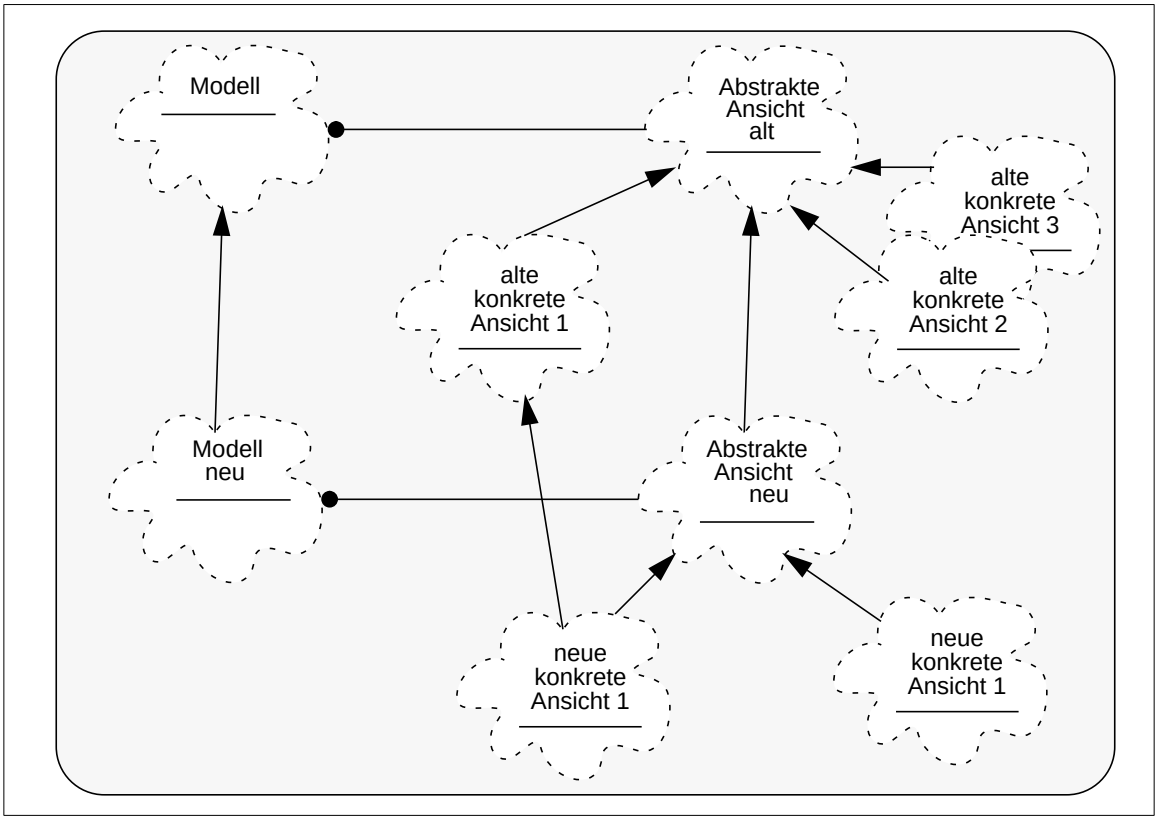

Abbildung 10-2: Erweiterung ausgehend von konkreter Bibliothek

Sind die Bestandteile neu zu implementieren wird von der allgemeinen Bibliothek ausgegangen und ausgehend von diesen Anteilen spezialisiert, siehe Abb. 10-3.

Wurden keine gemeinsamen Modelle für die Diagramme gefunden wird die Suche bei den Modellen untergeordneter Verwaltungsstrukturen fortgesetzt. Bisher ist dort nur die Zeichenfläche zu nennen. Es wird eine erneute Suche gestartet, diesmal in Bezug auf die Frage, ob bereits Zeichenflächen mit der gewünschten Anforderung existieren.

Diese Bestandteile werden dann in analoger Vorgehenweise wie bei Diagrammen erweitert. Bei Codeerweiterung ist zu beachten, daß sehr vorsichtig damit umgegangen werden muß, wenn nur die Ansichten erweitert werden. Bei den einzelnen Modellen der Zeichenflächen ist die primäre Frage, welche abstrakten Modelle der Ansicht im Bereich der Zeichenfläche integriert werden sollen. Dies legt die zugrundegelegte Interpretation und die sich daraus ergebenden Darstellungsmodelle fest.

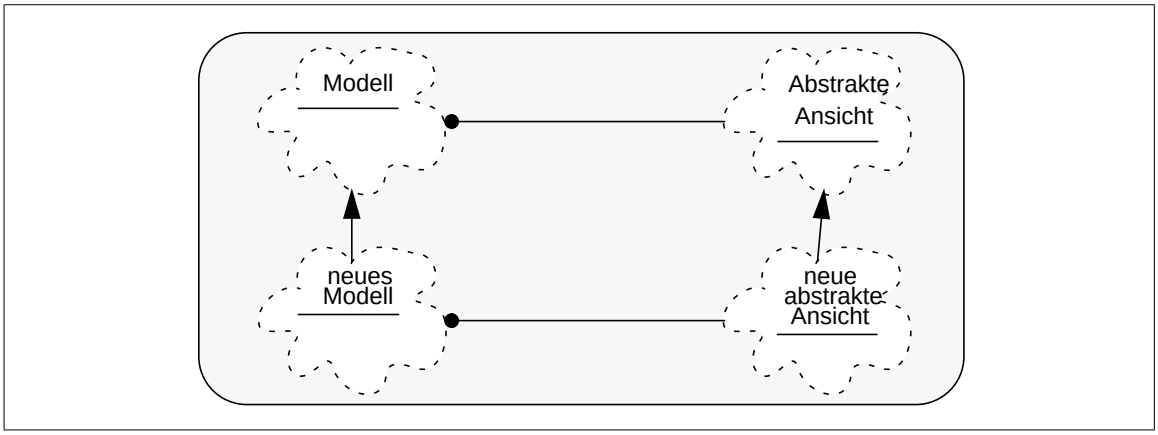

Abbildung 10-3: Erweiterung ausgehend von verallgemeinerter Bibliothek

Zuletzt gilt es zu überprüfen, ob die gewünschten Interpretationsvorschriften in bestehenden Bibliotheken bereits existieren oder ob diese neu zu implementieren sind. Bei den Darstellungsmodellen, also den Interpretationsvorschriften wird bisher davon ausgegangen, daß es sich um eine flache Hierarchie handelt. Dieses einfache Verhalten kann sich aber im fortschreitenden Ausbau ändern.

10.2 Verwaltungsstruktur für Diagramme

Wenn die grundlegenden Überlegungen beendet sind, sollte die wesentlichen Fragenfür ein konkretes Vorgehen sein:

- Welche Informationsanteile sollen für diese Struktur bereitgestellt werden?
- Werden Überschriften und Unterüberschriften benutzt?
- Wieviele Zeichenflächen und Legendengruppen sollen verwaltet werden?

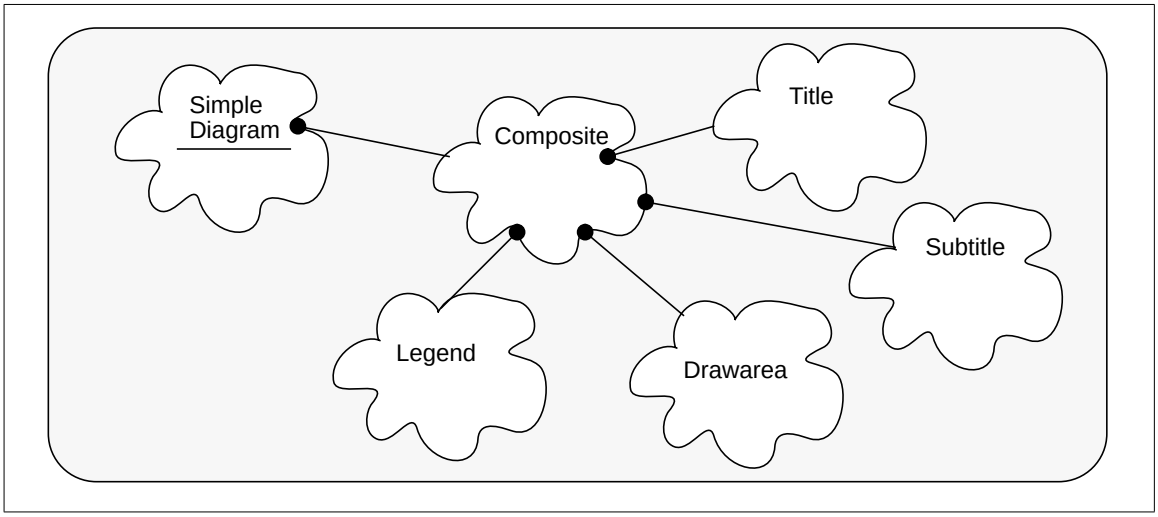

Abbildung 10-4: Objekt Diagramm für *Simple Diagram*

Diese Basiskomponenten für Diagramme sind entsprechend auszuwählen und aus Spezialisierungen aufzubauen. Für die Erstellung der Regressionsdiagramme wird vom einfachsten Aufbau für Diagramme ausgegangen. Ein einfaches Diagramm besitzt von allen Ausprägungen seiner Basiskomponenten genau eine, so daß genau eine Zeichenfläche und genau eine Legendengruppe enthalten sind.

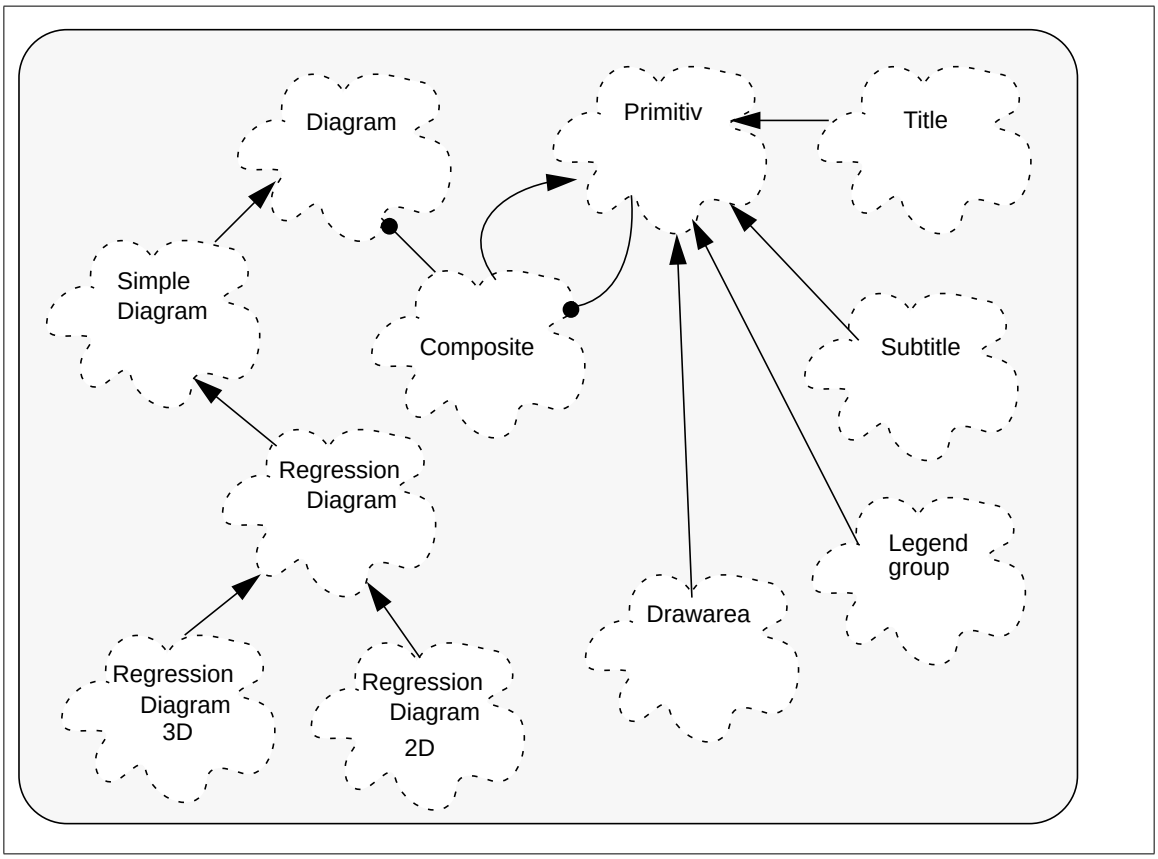

Abbildung 10-5: Verband auf Diagrammebene

10.2.1 Die Kopplungen zwischen Komponenten

Zu diesem Zeitpunkt sind die Basiskomponenten des neu erstellten Diagramms bereits festgelegt. Jetzt müssen Überlegungen stattfinden, welche Kopplungen zwischen den Basiskomponenten des Diagramms bestehen und wie bestimmte Methoden, zum Beispiel das Einfügen von Wertemengen, zu redefinieren sind. Das Vorgehen in diesem Bereich ist sehr stark abhängig von der konkreten Spezialisierung. Die für diesen Aufgabenbereich gestellten Fragen sind:

- Wie lassen sich neue Wertemengen einfügen?
- Welche Kopplungen bestehen zwischen Zeichenfläche und Legendengruppe?

- Wie sieht die Abhängigkeit zwischen Zeichenmodellen und den jeweils auferlegten Darstellungsmodellen, also Interpretationsvorschriften aus ?

Eine Wertemenge wird mit einem beschreibenden Text an die Zeichenflächen des Diagrammes angekoppelt. Beim Einfügen einer neuen Wertemenge wird also falls vorhanden, die Legendengruppe um einen Eintrag erweitert. Ist die Wertemenge in mehreren Zeichenflächen des Diagrammes eingebunden, gestaltet sich die Vorgehensweise etwas umfangreicher.

Im Fall der Regression kann hier sehr vereinfacht angesetzt werden. Beim Hinzufügen neuer Wertemengen wird die Legendengruppe um einen Eintrag erweitert. Die Reihenfolge der eingefügten Wertemengen ist seitens des Modelles unwichtig. Im Bereich der Ansichten stellt sie dagegen einen wesentlichen Gesichtspunkt dar. Prinzipiell entspricht die Reihenfolge der verwalteten Wertemenge im Modell denjenigen der Ansicht.

Im Modell wird unter Reihenfolge verstanden, in welcher Folge „change“ Operationen durchgeführt werden. In der Ansicht bezieht sich dies auf die Bearbeitungssequenzen für die „update“ und „redraw“ Nachrichten. In vielen Fällen werden die Reihenfolgen in Modell und Ansicht identisch sein, aber falls notwendig besteht die Möglichkeit die Reihenfolge in der Ansicht anders vorzunehmen. Die 3-dimensionalen Diagrammen ist dieser Fall die Regel, da die Bezugspunkte des Beobachters für jede Ansicht unterschiedlich gewählt werden können.

Bei der 2-dimensionalen Regression wird die Zeichenreihenfolge für die Ansichten, analog zu denen des Modells festgelegt. Werden zum Beispiel Bereichsbetrachtungen in Form eines dynamischen Intervalles über den Wertemengen durchgeführt, das heißt wird für feste Intervalle ein Bereich ermittelt, in dem Werte des Intervalls der betrachteten Wertemenge enthalten sind, so wird für jedes Intervall ein minimaler und ein maximaler Wert bestimmt, der das entsprechende Intervall im abstrakten Modell beschreibt. Abschließend werden diese Bestandteile in der Visualisierung für jede Wertemenge des Bereiches gezeichnet. Dabei kann die jeweils letzte durchgeführte Zeichenoperation die vorher durchgeführten Operationen überdecken.

10.3 Zeichenfläche

Nachdem die Überlegungen, die sich mit der Diagrammgestaltung und dessen Grobaufbau abgeschlossen sind, wendet sich die Betrachtung den einzelnen Zeichenflächen zu.

Für jede neue Zeichenfläche müssen die folgenden Fragen gestellt werden:

- Welche Darstellungsmodelle sollen verwendet werden?
- Wieviele verschiedene Darstellungsmodelle sind in der Zeichenfläche integriert?

- Welche Wertigkeit hat jedes Darstellungsmodell? Können von einem Modell gleichzeitig unterschiedliche Ausprägungen miteingebunden werden?
- Wie wird aus der Ansammlung von Darstellungsmodellen das abstrakte Modell berechnet? Wird hier jede Kopplung zwischen Darstellungsmodellen und Wertemenge nacheinander abgearbeitet oder nur inkrementell in bestimmten Schrittweiten eine Berechnung durchgeführt?

Desweiteren lassen sich hier Festlegungen bezüglich der Eigenschaften samt ihrer integrierten Anteile treffen. Welche Eigenschaften sollen Achsen, Beschriftung und Darstellungsbereich initial in den Ansichten aufweisen?

Im Fall der Regressionsdiagramme werden nur zwei Darstellungsmodelle an die Zeichenfläche gekoppelt. Zum einen das Darstellungsmodell für die Interpretationsvorschrift „Reduzierte Punktewolke“, zum anderen das Darstellungsmodell das funktionale Abschätzungen mittels Regression durchführt. Das Darstellungsmodell für die funktionale Abschätzung kann in mehreren Ausprägungen, in diesem Fall Funktionsarten, in die Zeichenfläche eingebunden werden. Neu eingefügte Wertemengen werden immer an das erste Darstellungsmodell gekoppelt. Ob bzw. welche Darstellungsmodelle im Bereich Regression angebunden werden sollen, muß für jede Wertemenge einzeln entschieden werden. Somit sind die Darstellungsmodelle, die die Regression durchführen, nur an einen begrenzten Anteil der Wertemengen angekoppelt.

Um die Abbildung des Modelles in die jeweiligen Ansichten einfach zu halten, ist die Bearbeitungsreihenfolge fest vorgegeben. Zuerst werden die Anteile der Punktewolke für jede Wertemenge berechnet. Danach wird für alle Darstellungsmodelle, die die Regression durchführen, der jeweilige Anteil im abstrakten Modell der Ansicht berechnet. Es werden zwei unterschiedliche Ausprägungen im Darstellungsmodell der Regression verwendet, diese werden auf allen Wertemengen der Zeichenmenge angewendet. So wird nacheinander die Berechnung über den Wertemengen ausgeführt und das abstrakte Modell der Ansicht ermittelt.

10.3.1 Arbeitsweise im Fall der Regression

Die Bearbeitungsschritte für den Fall der Regression werden kurz beschrieben um zu vermitteln, welche Vorgehensweise und welche Hintergedanken bei der Erstellung neuer Zeichenflächen durchzuführen sind.

Im Modell (Berechnungen sind unabhängig von der Reihenfolge):

- Berechne für jede Wertemenge eine zugrundegelegte Punktewolke für das abstrakte Modell.
    - Die resultierende Wertemenge im abstrakten Modell wird um die Anteile reduziert, die im Darstellungsbereich berücksichtigt werden können.

- Berechne in der im Modell festgelegten Reihenfolge für jede gewünschte Kombination aus Darstellungsmodellen und Wertemengen die entsprechenden Anteile des abstrakten Modelles der Ansichten.

Daraus ergeben sich im abstrakten Modell eine Vielzahl von „Punkten“ der verschiedenen Punktewolken und eine Ansammlung von funktionalen Beschreibungen in Form von Anfangs- und Endwert sowie die zugrundegelegte Bindefunktionen zwischen den Punkten. Aufsetzend auf diesen Bestandteilen des abstrakten Modells arbeiten die verschiedenen Ansichten.

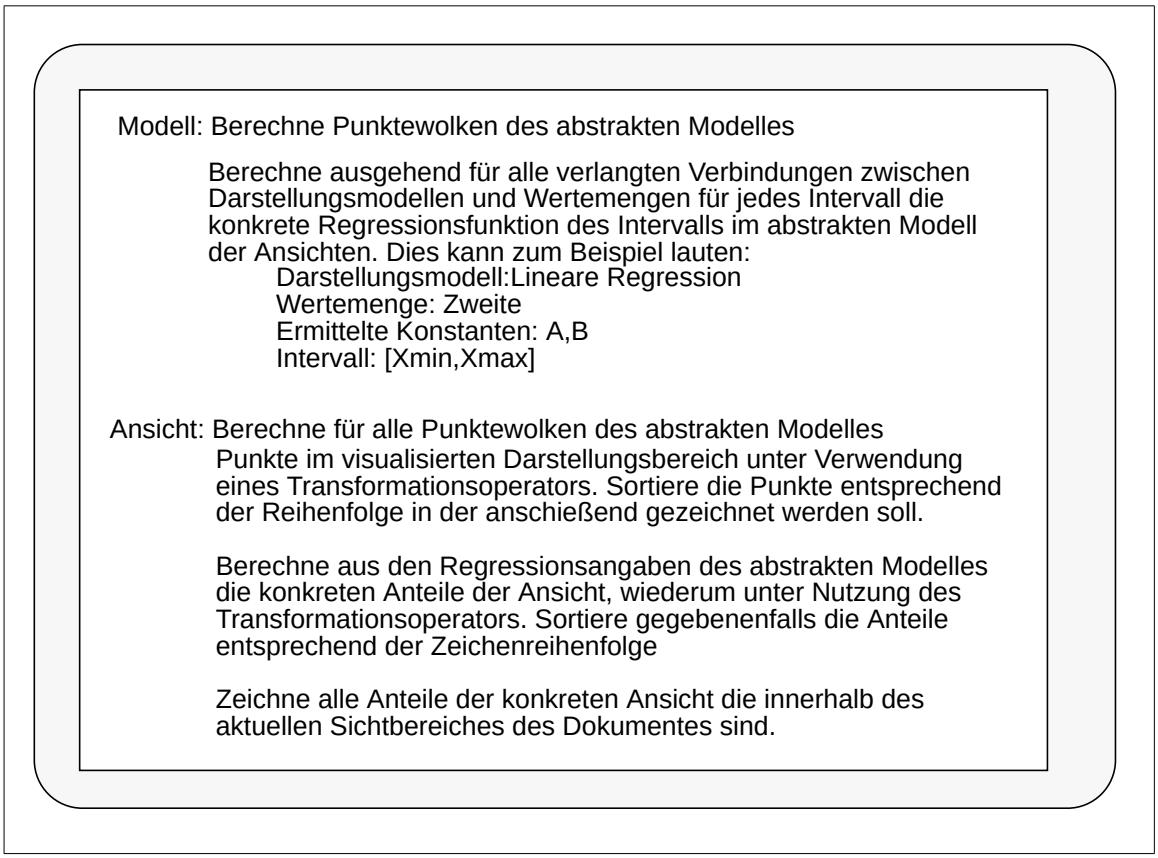

Abbildung 10-6: Arbeitsweise Regression

In der Ansicht werden folgende Bearbeitungen im Fall einer Aufforderung zum „Redraw“ durchgeführt:

- Werte den Anteile des abstrakten Modells aus, der für die Ansicht von Interesse ist. Die korrespondierenden Anteile in den Ansichten werden vorher gegebenenfalls umsortiert.
- Zwischen dem abstrakten Darstellungsmodell und der jeweiligen Ansicht ist ein Transformationsoperator zwischengeschaltet. Dieser rechnet die Anteile im abstrakten Modell in Koordinaten um, die in der visuellen Repräsentation verwendet werden können.

- Die Reihenfolge in der die Umwandlung von abstraktem Modell in die konkrete Ansicht durchgeführt wird ist die gleiche wie im Modell.

Bei der Aufforderung die Ansicht zu zeichnen wird zuerst ermittelt, welcher Bereich innerhalb des Sichtfensters des Dokumentes liegt und dieser wird dann neugezeichnet. Bei der Regressionsfunktion wird zusätzlich eine Abhängigkeit bestehen, ob es sich dabei um Geraden oder um über Punkte approximierte Funktionsverläufe handelt und diese dargestellt werden sollen.

10.4 Überlegungen zu den Darstellungsmodellen

Für die Auswertung innerhalb von Zeichenflächen müssen für die Wertemengen unterschiedliche Interpretationsvorschriften bereitgestellt werden. Die Interpretationsvorschrift ist nur eine Berechnungsvorschrift, die aus jeder Wertmenge die entsprechenden Anteile des abstrakten Modelles berechnet. Dieser Anteil wird anschließend in die konkrete Ansicht umgerechnet.

Die Vorgehensweise ist sehr einfach. Zuerst muß überlegt werden welche Werte und wie die für die Interpretation benötigten Werte aus der Wertemenge ermittelt werden sollen. Bei der Berechnung werden alle erzeugten „Objekte“ sequentiell in das abstrakte Modell eingetragen. Die Repräsentation der unterschiedlichen Darstellungsmodelle kann innerhalb des abstrakten Modelles sehr stark variieren.

Im Fall von Bereichsbetrachtungen werden jeweils für ein Intervall in X-Richtung, der minimale und maximale Y-Wert berechnet, so daß sich alle Y-Werte des Intervalles in dem berechneten Bereich befinden.

10.5 Manipulatoren

Zuletzt sollte man den Aufbau von Manipulatoren für die speziellen Diagramme berücksichtigen. Dabei sollte nach Ansicht des Autors analog zum hierarchischen Aufbau des Modelles vorgegangen werden. So lassen sich zumindest Anteile, die aus bestehenden Objekten der Bibliothek verwendet werden über bereits definierten Manipulatoren teilweise erledigen.

Das heißt, es sollte ein Manipulator für das Diagramm existieren. Bei diesem lassen sich alle Belange seiner direkt zugeordneten Basiskomponenten der Hauptgruppe einstellen. Dies betrifft in erster Linie die Größe und Position aber auch andere Attribute des Objektes.

11
Zusammenfassung

In der vorliegenden Arbeit wurde das Konzept und die Implementierung einer Bibliothek für die Visualisierung von Wertemengen vorgestellt. Die Ausarbeitung setzt am Beispiel der Regressionsanalyse an und zeigt ausgehend von dieser den Aufbau einer Bibliothek die diese Belange berücksichtigen kann. Die sich bei der Aufgabe herauskristallisierte Trennung zwischen spezialisierten Elementen und dem verallgemeinerten Ansatz, der für die Bildung einer Bibliothek erforderlich ist, wurde vorgestellt.

Es wurden zuerst die verschiedenen Möglichkeiten aufgezeigt, wie sich Wertemengen darstellen lassen. Dabei wurden die unterschiedlichen Schnittpunkte und Gemeinsamkeiten herausgearbeitet. Zusätzlich wurde zu Beginn die Entwurfsnotation nach Booch vorgestellt und die verwendete Implementierungsumgebung und die Sprache EIFFEL beschrieben. Eingehend wurden die mathematischen Mechanismen der Regressions- und Korrelationsanalyse beschrieben, eine weitergehende Vertiefung findet sich hierzu in Anhang A und B.

Die anschließende Beschreibung des groben Aufbaues der Bibliothek führte zur Dreiteilung der Aufgabe:

- Repräsentation von Wertemengen
- Regressions- und Korrelationsanalyse
- Visualisierung der Wertemengen

Diese Gliederung erwies sich für die Umsetzung als sehr hilfreich.

Die Repräsentation von Wertemengen wurde ausführlich erläutert und die dabei erforderlichen zwei unterschiedlichen Repräsentationen aufgezeigt. Die Notwendigkeit eines Konverters wurde beschrieben, genauso wie der Aufbau und die Verwendung von Projektionsverfahren um aus den n-dimensionalen Wertemengen darstellbare Wertemengen mit unter drei Dimensionen zu gewinnen.

Eine der vorgestellten Wertemengen ist optimiert auf Modifikationen über der Menge und dient daher dem Einlesen unsortierter Wertepaare. Die andere Darstellung für Wertemengen ist besser für iterative Berechnungen über Teilintervallen der Wertemenge geeignet.

Der Bereich der Regressions- und Korrelationsanalyse wurde eingehend beschrieben. Zuerst wurden alle Grundlagen über die mathematischen Hilfsmittel herausgearbeitet und dann die erforderliche Vorgehensweise erläutert. Wie gezeigt, arbeitet die Regression mit Hilfe von sogenannten Summeniteratoren. Der Berechnung liefert die Konstanten für die zugrundegelegte Regressionsfunktion. Für die Basisberechnungen werden im mehrdimensionalen Fall algebraische Ansätze aus dem Bereich der Matrizenrechnung benötigt. Die dafür notwendigen Strukturen und eine spezielle Erweiterung für die Regression wurden bereitgestellt.

Die beiden vorherigen Bereiche sind in Bezug auf die Trennung zwischen Spezialisierung und Verallgemeinerung eindeutig einem der beiden Bereiche zuzuordnen. Die Repräsentation der Wertemengen ist allgemeingültig und die Regressionsanalyse ist natürlich spezialisiert. Im 3. Aufgabenbereich der Visualisierung erwies es sich als besonders schwierig, die Verzahnung vom verallgemeinerten Ansatz und Spezialisierung geeignet zu trennen, da die Entwicklung der notwendigen Komponenten parallel lief und die Trennung nicht verwässert werden sollte.

Im allgemeinen Ansatz für die Visualisierung wurde eine geschachtelte Struktur aus Verwaltungscontainern vorgestellt, die eine grobe Verwaltungshierarchie aufbaut. Das Konzept wurde ausführlich beschrieben und es wurde auf Basisoperationen für graphische Objekte wie “Resize” und Positionieren eingegangen. Das Modell basierend auf geschachtelten Rahmen wurde beschrieben und auch das Auftreten der parallel verlaufenden Hierarchien aus Verwaltungscontainern und “Composites” dargelegt. Es wurde aufgezeigt wie einfach sich daraus komplexgekoppelte Strukturen visuellen Charakters aufbauen lassen. Es wurde veranschaulicht, daß die Kopplung zwischen den Komponenten erst in der Spezialisierung erfolgen kann.

Die Arbeit weist darauf hin das sich Probleme bei der eindeutigen Zuordnung auf Modell oder Ansicht nach dem Modell-View-Controller Ansatz für spezialisierte Diagramme ergeben können. Die vorliegende Arbeit versucht die Bibliothek so allgemein wie nur möglich zu halten und so eine zukünftige Nutzung basierend auf dem Ansatz zu ermöglichen. Die für die Implementierung oftmals notwendige Implementierung von Basiskomponenten samt ihrer Methoden war mühsam, aber da in den bereitgestellten Bibliotheken nicht vorhanden, notwendig.

Die Spezialisierung im Bereich der Regressionsanalyse und deren Abbildung wurde angeschnitten und beschrieben. Die hierfür erforderliche Abbildung geschieht über die Definition der entsprechenden Darstellungsmodelle und die Auswertung der Ansichten über deren abstraktem Modell. Im Falle der Spezialisierung ist die wesentliche Aufgabe die Bestandteile des Diagrammes festzulegen und die Kopplung zwischen den Bestandteilen herzustellen. Kriterien, die zusätzlich im Fall der Visualisierung von 3-dimensionalen Wertemengen zu beachten sind, wurden angeschnitten, eine Implementierung in diesem Bereich fand nicht statt.

Es wurde die Vorgehensweise beschrieben wie spezialisierte Diagramme zu erstellen sind, daß hierbei die Kopplung zwischen den einzelnen Anteilen der Visualisierung festzulegen ist und wie bei dem Entwurf der für diese Diagramme bereitgestellten Manipulatoren vorzugehen ist.

11.1 Ausblicke

Basierend auf der Arbeit können weitere Arbeiten aufsetzen. Wiederum erweist sich der dreigeteilte Ansatz der Teilaufgaben als geglückt. Ob die Arbeit in das „Application Framework“ miteingebunden werden soll oder nicht ist noch offen. Die möglichen Ausbaustufen für die einzelnen Bereiche:

- Die Repräsentation der Wertemengen kann auf allgemeine Tupel ausgebaut werden und dann als “Workspace”-Repräsentation im Fall von stattfindenden Datenbankanbindungen verwendet werden. Das heißt, es findet eine weitergehende Verallgemeinerung der Repräsentation statt. Die Repräsentation selbst sollte um eine Datenbankschnittstelle erweitert werden, so daß sich eine Speicherung in textueller Form erübrigt.
- Ein weiterer Ausbau der Wertemengen ist für den Fall von sehr großen Wertemengen, das heißt mit einer sehr großen Anzahl unterschiedlicher Tupel, sinnvoll. In diesem Fall ist eine Erweiterung in Form der Nutzung eines B*-Baumes wünschenswert. Dies ist unter dem Gesichtspunkt bereitzustellender Konverter und Iteratoren sicher eine schöne in sich geschlossene Arbeit. Ein Ausbau der Projektionen auf freidefinierte Funktionen und die dafür erforderliche Dialogstruktur ist wünschenswert.
- Die Auswertungsmethoden der Regressions- und Korrelationsanalyse lassen sich um weitere statistische Auswertungen erweitern. Dies umfaßt auch die Möglichkeiten, die sich aus den angesetzten Abschätzungsfunktionen und Konstanten ergeben.
- Die größten Möglichkeiten bestehen natürlich im visuellen Bereich. Hier können sukzessive alle Diagrammarten des Geschäftsbereiches integriert werden. Diese umfassen im wesentlichen 1-dimensionale Wertemengen, die entweder allein oder als Ansammlung über Darstellungsmodellen ausgewertet werden können.
- Genau so läßt sich parallel zur Verwaltungstruktur der Diagramme ein Verwaltungscontainer für feste Graphikobjekte aufbauen, deren Spezialisierung die zugrundegelegte Kodierung in Form von JPEG, GIF oder TIFF aufweist.
- Einem Ausbau im Bereich der Verwaltungsstrukturen Diagramm, Zeichenfläche und Darstellungsmodell sind nach Ansicht des Autors keine Grenzen gesetzt.
- Der Aufbau eines Klassenverbandes von Manipulatoren für komplexere Objekte sollte eine visuelle Repräsentation in Form eines „Notebooks“ bereitstellen.

11.2 Schlußbemerkung und Danksagung

Besonderer Dank gilt meinem Betreuer Dr. rer. nat. M. Ryba und seinem Kollegen Dipl.-Inform. P. Hofmann, die mit viel Eifer und nicht endender Geduld Ansätze diskutierten, neue Ideen einbrachten, die eigene Vorgehensweise kritisch hinterfragten und auch zur Seite standen wenn auf Seiten des Autors aus Mangel an Erfahrung Schwierigkeiten oder Mißverständnisse im Bereich der Abbildung auf die Sprache EIFFEL auftraten, sowie für den sehr kollegialen Umgangston untereinander.

Dank gilt auch meinen “Kollegen”: A. Grieshaber, F. Wohlgemuth, R. Brodbeck, J. Fasolt, R. Sakretz, P. Geretschläger und H. Reti, die im Diplomandenraum einigen bissigen Bemerkungen ausgesetzt waren, wenn der Autor sich über eigene Unzulänglichkeiten aufregte. Ich hoffe, daß ich meinen Kollegen einen Teil der durch sie geleisteten Unterstützung bei Motivation, Ideen, Konzepten und Ansätzen zurückgeben konnte und bedanke mich für die Professionalität in der Zusammenarbeit mit ihnen.

Außerordentlichen Dank schulde ich B. Meyer für den erstklassigen EIFFEL-Compiler, aber leider auch im negativen Sinne für das wiederholt auftretende Problem der unvollständigen Dokumentation von Bibliotheken der Entwicklungsumgebung. Die Online-Hilfe ist dem Autor als herausragendes Negativbeispiel für professionelle Entwicklungsumgebungen in bleibender Erinnerung. Es ist bedauerlich, daß sich ein so gutes Produkt durch fehlende Dokumentation selbst “das Wasser abgräbt”. Es bleibt zu hoffen, daß sich dies zukünftig ändert. Die Sprache EIFFEL ist aufgrund des klaren Aufbaues geeignet im Grundstudium als Vertreter objektorientierter Sprachen eingeführt zu werden.

12
Literatur

Beic95 Beichelt Frank
Statik für Ingenieure
B. G. Teubner Verlag, Stuttgart 1995
ISBN 3-519-02987-1

Boo94 Booch G.
Objectoriented Analysis & Design 2. Edition
Benjamin/Cummings Publishing Company, Readwood City, CA 1994
ISBN 0-8053-5340-2

Bort85 Prof. Dr. Bortz Jürgen
Statistik
Springer Verlag, Berlin 1985
ISBN 3-540-50736-1

CaRo91 Cameron Debra, Rosenblatt Bill
Learning GNU Emacs
O'Reilly Association Inc.
ISBN 0-937175-84-6

ChPr95 Chatterjee Samprit, Price Bertran
Praxis der Regressionsanalyse
Oldenburg Verlag, München 1995
ISBN 3-486-23302-5

EnSt86 Encarnacao, Strasser W.
Computer Graphics
Oldenburgverlage, München 1994
ISBN 3-486-34652-0

EnRe87 Engelins-Müller G., Reuter F.
Numerische Mathematik für Ingenieure
BI Wissenschaftsverlag, 1987
ISBN 3-4111-031561-4

Fal95 Falin Gerhard
Kurven und Flächen in Computer Aided Design, 2. Auflage
Vieweg Verlag, Braunschweig Wiesbaden 1994
ISDN 3-528-16542-1

Fel92 Fellner W. D.
Computergraphik Band 58, 2. Auflage
B. I. Wissenschaftsverlag, Mannheim 1992
ISBN 3-411-15122-6

FoDa90 Foyler, van Dam, Feiner, Hughes
Computer Graphics
Addison Wesley Publishing Company, 1990
ISBN 0-201-12110-7

Gam95 Gamma Erich et. al.
Design Patterns: elements of reusable object-oriented software
Addison Wesley, Reading, Massachusetts, 1995
ISBN 0-201-63361-2

GoRo89 Goldberg Adele, Robson David
Smalltalk 80 - The language
Addision Wesley Publishing Inc., 1989
ISBN 0-201-13688-0

Grh96 Grieshaber Armin
Entwicklung grundlegender objekt-orientierter Mechanismen zur Konstruktion von Diagrammeditoren.
Teil 2: Komponentenentwicklung
Studienarbeit 1544 am IPVR, Abt. ISE, Universität Stuttgart, 1996

Gri89 Dr. Grieger
Graphische Datenverarbeitung - Mathematische Modelle
Springer Verlag, Berlin 1987
ISBN 3-540-17895

GuOb94 Gulbins J., Obermayer K.
Desktop Publishing mit Framemaker
Springer Verlag, Berlin 1994
ISBN 3-540-36643-0

Hart85 Hartung Joachim
Statik
Oldenburg Verlag, München 1985
ISBN 3-486-26793-0

HaHe87 Hartung Joachim, Heine
Statistik Übungen
Oldenburg Verlag, München 1987
ISBN 3-486-20519-6

Hru94 Hruschka, Peter
Objektorientierte Analyse- und Designprinzipien
Objekt Spektrum 2/94
Seite 16 - 22

Ko85 Kohler Heinz
statistics for business and economics
Scott, Foresman and Company, USA Glenview Illinois, 1985
ISBN 0-673-15822-5

Krz88 Krazanowski W. J.
Principles of multivariant Analysis
Oxford Science Publications, Oxford 1988
ISBN 0-19-8522115-8

Mari95 Marinell Gerhard
Multivariante Verfahren
Oldenburg Verlag, München 1995
ISBN 3-486-23302-0

MeSi95 Mendihall William, SincichTerry
statistic for engineering and science, 4. ed
Prentice Hall Inc. Eaglewood Cliffs, New Jersey 1995
ISBN 0-02-380581-1

Mey90 Meyer Bertrand
Objektorientierte Software-Entwicklung
Hanser Verlag, München, 1990
ISBN 3-446-15773-5

Mey92 Meyer Bertrand
Eiffel: the language
Prentice Hall International (UK), Hertfordshire, 1992
ISBN 0-13-247925-7

Mey94 Meyer Bertrand
Reusable software: the base object oriented component libraries
Prentice Hall International (UK), Hertfordshire, 1994
ISBN 0-13-245499-8

MiVi95 M. Minas, G. Viehstaedt
DiaGen: A Generator for Diagram Editors Providing Direct Manipulation and Execution of Diagrams
Proceedings of the 11th IEEE Symposium on Visual Languages, 1995

Mon93 Monninger F.
Objektorientiertes Programmieren in der Praxis
Verlag Heinz Heise
ISBN 3-88229-028-3

MVB95 Meyer Bertrand , Valente Dino , Bezault Eric
Building graphical Applications with EiffelBuild
ISE Technical Report TR-EI-43/UL, Interactive Software Engineering, 1995

Ra93 Rauber, Thomas
Algorithmen der Computergraphic
B. G. Teubner Verlag, Stuttgart 1993
ISBN 3-519-02127-7

Ru86 Rutsch Martin
Statik 1
Birhauser Verlag, Basel 1986
ISBN 3-7643-1740-X

Ru87 Rutsch Martin
Statik 2
Birkhauser Verlag, Basel 1987
ISBN 3-7643-1813-9

SaSl87 Salomon Ros, Slater Mel
Computer Graphics - Systems & Concepts
Addision Wesley Publishing Company, 1987
ISBN 0-201-14656-8

Scha90 Schaller Thomas
Business Graphics
Sybex Verlag, Düsseldorf 1990
ISBN 3-88745-788-9

Sed90 Sedgewick Robert
Algorithmens in C
Addision Wesley Publishing Inc., 1990
ISBN 0-201-51425-7

Sed92 Sedgewick Robert
Algorithmen in C++
Addision Wesley GmbH., München 1992
ISBN 0-201-51425-7

StBa94 Stein W., Balzert H.
Worin unterscheiden sich objektorientierte Methoden
Objekt Spektrum 2/94
Seite 23 - 28

Str91 Stroustrup Bjarne
The C++ programming language, 2 ed.
Addison Wesley Publishing Inc., 1991
ISB 0-201-53992-6

Wien95 Wiener Richard
Software development using Eiffel: there can be life other than C++
Prentice Hall, , Englewood Cliffs, New Jersey, 1995
ISBN 0-13-100686-X

Woh96 Wohlgemuth Florian
Entwicklung grundlegender objekt-orientierter Mechanismen zur Konstruktion von Diagrammeditoren.
Teil 1: Komponentenverwaltung
Studienarbeit 1543 am IPVR, Abt. ISE, Universität Stuttgart, 1996

WoWo90 Wonnacott Thomas H. , Wonnacott Roland J.
Introductory statistics for business and economics, 4. Ed.
Wiley & Sons 1990
ISBN 0-471-61517-X

ZaKo95 Zavodnik R., Kopp H.
Graphische Datenverarbeitung, Grundzüge und Anwendungen
Hanser Verlag, München 1995
ISBN 3-446-17907-0

Anhang A
Regressionsanalyse

A.1 Einleitung

Die Regression versucht einen funktionalen Zusammenhang zu liefern. Die Ziele dabei sind unter anderem:

- Nachweis von bekannten Beziehungen
- Schätzen von Parametern einer bekannten funktionalen Beziehung
- Erkennen von funktionalen Zusammenhängen
- Repräsentation großer Datenmengen
- Interpolation fehlender bzw. Prognose zukünftiger Werte

Interessiert nur der Zusammenhang zweier Merkmale X und Y mit Ausprägungsvariablen x und y, Regressor und Regressand genannt so wird eine Beziehung $y = f(x)$ spezifiziert. Dies wird als Regression von Y auf X bezeichnet. Mittels einer Stichprobe wird dann der funktionale Zusammenhang geschätzt. Die einfachste, funktionale Abhängigkeit ist linear, von daher wird sie als lineare Regression bezeichnet. In der Residuenanalyse wird eine Überprüfung des Ansatzes vorgestellt. Viele Zusammenhänge lassen sich aber besser durch nicht lineare Regression modellieren, zu nennen sind die polynominale und multiple Regression. Für ein noch genaueres Studium der Regression wird auf die folgende Literatur verwiesen: [Beic95],[Bort85],[ChPr95], [Hart85],[Ko85],[Krz88],[Mari95],[MeSi95],[Ru86],[Ru87].

Die im folgenden Abschnitt auftretenden Quantilen sind:

- $t_{v,y}$ der t_v - Verteilung
- $\chi^2_{v,y}$ der χ - Verteilung
- $F_{v_1,v_2;y}$ der F_{v_1,v_2} - Verteilung

A.2 Lineare Regression

Vermutet man einen zumindest näherungsweisen, linearen Zusammenhang zwischen zwei Merkmalen in ihrer Gesamtheit, so kann mittels linerar Regression dieser Zusammenhang näher spezifiziert und untersucht werden. Es wird in diesem Fall davon aus-

gegangen, daß dann gilt: $y_i = \alpha + \beta \cdot x + e_i$ für $i = 1,...,n$ mit Verschiebung α, Steigung β und dem zufälligen Fehler e_i. Nachfolgend werden Punkt-, Intervallabschätzung und Tests für die Parameter α und β des Regressionsproblemes angegeben. Der dabei entstehende zufällige Fehler e_i wird genauer analysiert.

A.2.1 Methode der kleinsten Quadrate

Die Punktabschätzung von a und b für die Parameter α und β eines linearen Regressionsproblemes wird so bestimmt, daß durch die Gerade $\hat{y} = \alpha + \beta \cdot x$ eine möglichst gute Schätzung für die Ausprägung y geliefert wird. Als Kriterium der Güte dieser Abschätzung ist die Summe der Abweichungsquadrate (Residualquadrate) ein geeignetes Maß.

$$S^2 = \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 = \sum_{i=1}^{n} \left(y_i - (a + b \cdot x_i) \right)^2$$

Formel A-1: Methode der kleinsten Quadrate

Somit ergeben sich a und b als Lösung des Normalengleichungssystem

$$\frac{\partial S}{\partial a} = -2 \sum_{i=1}^{n} \left(y_i - (a + b \cdot x_i) \right) = 0$$

$$\frac{\partial S}{\partial b} = -2 \sum_{i=1}^{n} x_i \cdot \left(y_i - (a + b \cdot x_i) \right) = 0$$

Formel A-2: Normalengleichung bei linearer Regression

das heißt die Schätzer a und b sind:

$$b = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \quad \text{und} \quad a = \bar{y} - b \cdot \bar{x}$$

Formel A-3: Die kleinsten Quadratschätzer bei linearer Regression

Eine nach der Methode der kleinsten Quadrate geschätzte Regressionsgerade schneidet stets den Punkt $(\bar{x}, \bar{y})$ und bei fehlerfreier Messung sind die Fehlerterme $e_1,...,e_n$ unabhängig mit dem Erwartungswert 0 und der Varianz σ^2.

Es ergibt sich somit für die Schätzer a und b Varianzen der Art .

$$\sigma_a^2 = \left(\frac{1/n + \bar{x}^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \right) \cdot \sigma^2$$

$$\sigma_b^2 = \frac{\sigma^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

Formel A-4: Varianz der Schätzer bei linearer Regression

Gilt zusätzlich, daß die Fehlerterme normalverteilt sind, dann sind auch die Schätzer a und b normalverteilt und ihre Kovarianz beläuft sich auf:

$$COV(a, b) = \frac{-\bar{x}\sigma^2}{\sum_{i=1}^{n} (x_i \bar{x})^2}$$

Formel A-5: Kovarianz der Schätze a,b bei linearer Regression

und die Korrelation

$$\delta_{a,b} = \frac{-n\bar{x}}{\sqrt{n \sum x_i^2}}$$

Formel A-6: Korrelation der Schätzer bei linearer Regression

A.2.2 Schätzen der Fehlervarianz

Da die Varianzen σ_a^2 und σ_b^2 der kleinsten Quadratschätzer von der Varianz σ^2 der zufälligen Fehler $e_1,...,e_n$ abhängen, möchte man σ^2 erwartungstreu abschätzen. Schätze die theoretischen Residuen mit $e_i = y_i - (\alpha + \beta x_i)$ durch $\hat{e}_i = y_i - (a + b x_i)$ und nehmen für σ^2 die Schätzung,

$$s^2 = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n-2} = \frac{\sum_{i=1}^{n} (\hat{e}_i)^2}{n-2} = \frac{S^2}{n-2}$$

Formel A-7: Schätzer für Varianzverhalten

die unter allen erwartungstreuen Schätzern die kleinste Varianz besitzt.

A.2.3 Zusammenhang zwischen Regression und Korrelation

Der Pearson Koeffizent r_{XY} zweier Merkmale steht im engem Zusammenhang zum Schätzer b des Steigungsparameters β. Es gilt:

$$
\begin{aligned}
b &= \frac{\sum_{i=1}^{n} (x_i - \bar{x})\,(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \\
&= \frac{\left(\sum_{i=1}^{n} (x_i - \bar{x})\,(y_i - \bar{y}) \right) \sqrt{\sum_{i=1}^{n} (y_i - \bar{y})^2}}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \cdot \sum_{i=1}^{n} (y_i - \bar{y})^2 \cdot \sum_{i=1}^{n} (x_i - \bar{x})^2}} \\
&= r_{XY} \cdot \frac{\sum_{i=1}^{n} (y_i - \bar{y})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \\
&= r_{XY} \cdot \frac{s_Y}{s_X}
\end{aligned}
$$

Formel A-8: Zusammenhang Regresssion und Korrelation bei linearer Regression

Aus der Korrelation r_{XY} kann also der Steigungsparameter β der linearen Regression von Y nach X geschätzt werden und umgekehrt.

Als Maß für die Güte der Anpassung, die eine Regression erzielt, dient das *Bestimmtheitsmaß* der Regression.

$$
\begin{aligned}
B_{Y,X} &= 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \\
&= 1 - \frac{\sum_{i=1}^{n} (\hat{y} - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}
\end{aligned}
$$

Formel A-9: Bestimmtheitsmaß der Regression bei linearer Regression

$$= \quad r_{XY} \cdot \frac{s_{\hat{Y}}}{s_Y}$$

Formel A-9: Bestimmtheitsmaß der Regression bei linearer Regression

Also das Verhältnis der Varianzen der geschätzten Werte $\hat{y}_i$ und den beobachteten Werten, genauer der Anteil an der Varianz Y, der durch das Merkmal X erklärt werden kann. Es gilt stets: $0 < B_{X,Y} < 1$ eine optimale Anpassung ist natürlich erreicht, wenn gilt $B_{X,Y} = 1$, denn dann wird Y durch die Regression vollständig erklärt.

Die Größe $U_{X,Y} = 1 - B_{X,Y}$ nennt man das *Unbestimmtheitsmaß* einer Regression. Speziell bei der linearen Regression ist das Bestimmtheitsmaß identisch mit dem Quadrat der Korrelationen von X und Y, das heißt es gilt: $B_{X,Y} = r_{XY}^2$.

A.2.4 Konfidenzintervall und Testen

Als nächstes wird kurz der Bereich des Testens von Hypothesen zum Niveau γ bei unbekannten Parametern α, β und σ^2 der linearen Regression angeschnitten. Die Entscheidungshilfen zum Testen von Hypothesen H_0 gegen die Alternative H_1 sind in der Tabelle aufgeführt:

Hypothese H_0	Alternative H_1	H_0 wird verworfen falls
$\alpha = \alpha_0$	$\alpha \neq \alpha_0$	$\left\lvert \frac{a - \alpha_0}{s_a} \right\rvert > t_{n-2;(1-\gamma/2)}$
$\alpha \leq \alpha_0$	$\alpha > \alpha_0$	$\frac{a - \alpha_0}{s_a} > t_{n-2;1-\gamma}$
$\alpha \geq \alpha_0$	$\alpha < \alpha_0$	$\frac{a - \alpha_0}{s_a} < t_{n-2;\gamma}$
$\beta = \beta_0$	$\beta \neq \beta_0$	$\left\lvert \frac{b - \beta_0}{s_b} \right\rvert > t_{n-2;(1-\gamma/2)}$

Formel A-10: Test des Konfidenzintervalles

$\beta \le \beta_0$	$\beta > \beta_0$	$\frac{b-\beta_0}{s_b} > t_{n-2;1-\gamma}$
$\beta \ge \beta_0$	$\beta < \beta_0$	$\frac{a-\beta_0}{s_b} < t_{n-2;\gamma}$
$\sigma^2 = \sigma_0^2$	$\sigma^2 \ne \sigma_0^2$	$\frac{(n-2)\,s^2}{\sigma_0^2} > \chi^2_{n-2;(1-\gamma/2)}$
$\sigma^2 \le \sigma_0^2$	$\sigma^2 > \sigma_0^2$	$\frac{(n-2)\,s^2}{\sigma_0^2} > \chi^2_{n-2;1-\gamma}$
$\sigma^2 \ge \sigma_0^2$	$\sigma^2 < \sigma_0^2$	$\frac{(n-2)\,s^2}{\sigma_0^2} < \chi^2_{n-2;\gamma}$

Formel A-10: Test des Konfidenzintervalles

Für den Regressionkoeffizenten β kann man ein $1-\gamma$ Konfidenzintervall berechnen als $[b \mp s_b t_{n-2;1-\gamma/2}]$ und für das Absolutglied $[a \mp s_a t_{n-2;1-\gamma/2}]$ ein Konfidenzintervall zum Niveau $1-\gamma$. Diese bedecken die unbekannten Parameter α und β eines linearen Regressionsproblemes mit der Wahrscheinlichkeit $1-\gamma$. Für die Fehlervarianz σ^2 ergibt sich somit das Intervall

$$\left[(n-2)\cdot\frac{s^2}{\chi^2_{n-2;1-\gamma/2}};(n-2)\frac{s^2}{\chi^2_{n;1-\gamma/2}}\right]$$

Formel A-11: Intervall der Fehlervarianz

ein Konfidenzintervall zum Niveau $1-\alpha$.

Simultane Konfidenzintervalle zum Niveau $1-\alpha$ für die Parameter α und β sind nun $\left[a \pm \sqrt{2 \cdot s_a^2 \cdot F_{2,n-2;(1-\gamma)}}\right]$ und $\left[b \pm \sqrt{2 \cdot s_b^2 \cdot F_{2,n-2;(1-\gamma)}}\right]$. Das erste Intervall bedeckt den Parameter α und das zweite den β. Es wird insgesamt das Niveau $1-\alpha$ eingehalten.

A.2.5 Konfidenz und Prognosestreifen

Bisher wurde nur das Konfidenzintervall für die Parameter, die ein lineares Regressionsproblem beschreiben, betrachtet. Man ist jedoch oftmals daran interessiert Konfidenzintervalle für einen erwarteten Wert $E(y|x_0)$ von y an der Stelle x_0 zu bestimmen. Mit der Konstanten

$$C = s \cdot t_{n-2;1-\gamma/2} \cdot \sqrt{\frac{1}{n} + \frac{(\hat{x} - x_0)^2}{\sum (x_i - \bar{x})}}$$

Formel A-12: Konstante des Konfidenzintervalles

ist ein $1-\gamma$ Konfidenzintervall gegeben durch $\left[a + (b \cdot x_0) \mp C\right]$. Betrachtet man das Konfidenzintervall für y an der Stelle x_0 so ergibt sich ein Konfidenzstreifen der überall verschiedene Breiten hat und an der Stelle $x_0 = \bar{x}$ am schmallsten ist. Dieses Konfidenzintervall bedeckt lediglich den Erwartungswert an der Stelle x_0 mit der Wahrscheinlichkeit $1-\gamma$. Ist man hingegen interessiert ein Intervall zu bestimmen, daß eine Realisation von y an der Stelle x_0 mit der Wahrscheinlichkeit $1-\gamma$ hat, so muß man sein Prognoseintervall für y_0 an der Stelle x_0 mit der Trefferwahrscheinlichkeit $1-\gamma$ bestimmen, das stets breiter ist, als das entsprechende Konfidenzintervall. Das Prognoseintervall $\left[a + (b \cdot x_0) \mp D\right]$ ist gegeben mit:

$$D = s \cdot t_{n-2;1-\gamma/2} \cdot \sqrt{1 + \frac{1}{n} + \frac{(\hat{x} - x_0)^2}{\sum (x_i - \bar{x})}}$$

Formel A-13: Konstante für Prognoseintervall in jedem Punkt

Ein Prognoseintervall dieser Art ergibt sich, wenn man an jeder Stelle x_0 ein Prognoseintervall berechnet. Will man einen simultanen Konfidenzstreifen für die Regressionsgerade, das heißt einen Streifen der zugleich an der Stelle x_0 den Erwartungswert $E(y|x_0)$ von y mit der Wahrscheinlichkeit $1-\gamma$ überdeckt, bestimmen, so berechnet man an jeder Stelle x_0 das Intervall:

$$\left[a + bx_0 \pm \sqrt{2 \cdot s^2 \cdot F_{2,\,n-2;\,(1-\gamma)} \cdot \left(\frac{1}{n} + \frac{(\bar{x} - x_0)^2}{\sum (x_i - \langle \bar{x} \rangle)} \right)} \right]$$

Formel A-14: Prognoseintervall in jedem Punkt

A.2.6 Regression durch vorgegebene Punkte

Es gibt Problemstellungen bei denen es von vornherein sinnvoll erscheint zu verlangen, daß die Regressionsgerade durch einen bestimmten Punkt (x_0, y_0) geht. Bei einer Regression die keinen speziellen Wert schneiden soll ist bekannt, daß die Gerade stets durch den Punkt $(\bar{x}, \bar{y})$ geht; so können wir die Parameter α und β schätzen. Die Verwendung von den Mittelwerten liefert folgende Schätzer

$$b = \frac{\sum_{i=1}^{n} (x_i - \bar{x}) \cdot \sum_{i=1}^{n} (y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}$$

Formel A-15: Schätzer bei Regression durch vorgegebenen Punkt

Bei Festlegung des Schnittpunktes auf den Ursprung entfallen die Verschiebungen:

$$b = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2} \qquad \text{und} \qquad a = 0$$

Formel A-16: Schätzer bei Schnittpunkt mit dem Ursprung

Die Varianz σ^2 bleibt $s^2 = \frac{1}{n} \cdot \sum (y_i - \hat{y}_i)$ mit $\hat{y} = a + b \cdot x_i$

A.3 Residuenanalyse

Die Fehler $e_i = y_i - \hat{y}_i$ werden als *Residuen* bezeichnet. Normiert man die Residuen mit dem Schätzer für die Standardverteilung der Beobachtung $\hat{e}_i = e_i / s$ so ergeben sich *normierte Residuen*. Diese sind, falls der Modellansatz mit normalverteilten Fehlern e_i richtig ist, approximativ $N(0, 1)$ verteilt.

Bei Darstellung von normierten Residuen in Bezug auf $\hat{y}$ ergeben sich Schlüsse über die Güte des Modellansatzes. Man kann normierte Residuen auch in Abhängigkeit von der zeitlichen Reihenfolge der Beobachtung darstellen, um zu untersuchen, ob diese Faktoren einen Einfluß auf die Werte von y haben. Mit Hilfe der graphischen Darstellung ist das Auffinden sogenannter Ausreißer in der Wertemenge möglich, diese lassen sich so gegebenenfalls eliminieren. Als grobe Regel mag der Ansatz $|\hat{e}_i| > 3$ gelten. Ist die Gesamtheit nicht normalverteilt so ist eine Transformation durchzuführen.

A.4 Transformation nach Linearitt

Es kann bei einem anderen als linearen Verhalten eine Transformation der Werte sinnvoll sein. Bei der Transformation ändern sich die Parameterschätzungen durch die Methode der kleinsten Quadrate in gleicher Weise, wie sich die Parameter selbst ändern.

Funktion	transformierte Funktion
$y = \alpha x^{\beta}$	$\ln(y) = \ln(\alpha) + \beta \cdot \ln(x)$
$y = \alpha e^{\beta x}$	$\ln(y) = \ln(\alpha) + \beta \cdot x$
$y = (\alpha + \beta x)^{-1}$	$\frac{1}{y} = \alpha + \beta x$
$y = \frac{x}{\alpha + \beta x}$	$\frac{1}{y} = \alpha + \frac{\beta}{x}$
$y = \alpha e^{\beta / x}$	$\ln(y) = \ln(\alpha) + \frac{\beta}{x}$
$y = \frac{1}{\alpha + \beta e^{x}}$	$\frac{1}{y} = \alpha + \beta e^{x}$
$y = \alpha x^{\beta} e^{\gamma x}$	$\ln(y) = \ln(\alpha) + \beta \ln(x) + \gamma x$

Formel A-17: Transformationstabelle der Regressionsanalyse

Auch kann man in Situationen, in denen eine einfache Regression nicht angebracht ist, die Methode der multiplen Regression anwenden.

A.5 Nicht lineare Regression

Die bisher betrachteten Regressionsprobleme wurden immer unter dem Gesichtspunkt einer Regressionsgeraden abgeschätzt und untersucht. Oftmals ist die Wahl eines nichtlinearen Ansatzes bei der Regressionsanalyse angebrachter. Eine nichtlineare Regression zum Beispiel die funktionale Abhängigkeit bei Funktionen 2. Grades. Besteht die Annahme, daß zwischen den Merkmalen X,Y eine quadratischer Zusammenhang besteht, so wird der Ansatz $y_i = a + \beta_1 x_i + \beta_2 x_i^2 + e_i$ mit $i = 1,...,n$ der Regressionsfunktion $\hat{y} = a + bx + cx^2$ abgeschätzt. Es wird unterstellt, daß die Fehlerterme $e_1,...,e_n$ unabhängig voneinander und normalverteilt sind, so daß nach der Methode der kleinsten Quadrate durch Minimierung der Fehlerquadrate die Schätzer a, b_1 und b_2 errechnet sind. Der Ansatz geht wieder über die Normalengleichungen und deren Ableitungen. Dieses Verfahren liefert:

$$a = A\sum y_i + D\sum x_i y_i + E\sum x_i^2 y_i$$

$$b_1 = D\sum y_i + B\sum x_i y_i + G\sum x_i^2 y_i$$

$$b_2 = E\sum y_i + G\sum x_i y_i + C\sum x_i^2 y_i$$

Formel A-18: Schätzer für Regression 2. Grades

mit folgenden Hilfsgrößen:

$$A = [(\sum x_i^2)(\sum x_i^4) - (\sum x_i^3)^2] / K$$

$$B = [n(\sum x_i^4) - (\sum x_i^2)^2] / K$$

$$C = [n(\sum x_i^2) - (\sum x_i)^2] / K$$

$$D = [(\sum x_i^3)(\sum x_i^2) - (\sum x_i)(\sum x_i^4)^2] / K$$

$$E = [(\sum x_i)(\sum x_i^3) - (\sum x_i^2)^2] / K$$

$$G = [(\sum x_i)(\sum x_i^2) - n\sum x_i^3] / K$$

$$K = n\sum x_i^2 \sum x_i^4 + 2\sum x_i \sum x_i^2 \sum x_i^3 - (\sum x_i^3)^3 - n(\sum x_i^3)^2 - (\sum x_i)^2 \sum x_i^4$$

Formel A-19: Hilfsgrößen bei Regression 2. Grades

Damit läßt sich $\hat{y}$ bestimmen und die Varianz der Fehler wird abgeschätzt mit.

$$s^2 = \frac{1}{n-3}\sum (y_i - \hat{y}_i)$$

Formel A-20: Schätzer für Regression 2. Grades

Die empirischen Varianzen und Korrelationen der Schätzer sind dann:

$$s_a^2 = s^2 A \qquad s_{b_1}^2 = s^2 B \qquad s_{b_2}^2 = s^2 C$$

Formel A-21: Varianz der Schätzer

$$\rho_a = D/\sqrt{AB} \qquad \rho_{b_1} = E/\sqrt{AC} \qquad \rho_{b_2} = G/\sqrt{BC}$$

Formel A-22: Korrelation der Schätzer

Das Konfidenzintevall zum Niveau $1-\gamma$ für den Erwarungswert von y ist an der Stelle x_0 bestimmt, der einen Koeffizenzbereich der Kurve liefert.

$$g(x_0) = A + 2Dx_0 + (B + 2E)\, x_0^2 + 2Gx_0^3 + Cx_0^4$$

Formel A-23: Methode der kleinsten Quadrate

ist ein $1-\gamma$ Konfidenzintervall für den Erwartungswert $E(y|x_0)$ von y in x_0 definiert. Häufig ist ein besserer Ansatz eine Regression mit orthogonalen Polynomen durchzuführen, da sich der Aufwand verringert und sich die numerische Stabilität günstiger auswirkt. Ein Regressionspolygon k-ten Grades $y = \sum_{i=0}^{k} \beta_0 x^i$ läßt sich leicht durch Umbenennung von $x^j \rightarrow x_j$ in eine multiple Regressionsrechnung umwandeln.

A.6 Multiple Regression

Betrachte die k+1 Merkmale $Y, X_1, ..., X_k$ und versucht damit einen funktionalen Zusammenhang mit folgendem Ansatz zu finden,

$$y(x_1,...,x_k) = \beta_0 + \beta_1 x_1 + ... + \beta_k x_k = \sum_{i=0}^{k} \beta_i x_i$$

Formel A-24: Multiple Regression mit k - Parametern

so nennt man diese Funktion eine multiple Regressionsgleichung von Regressand Y und Regressoren $X_1, ..., X_k$

Der Ansatz über ein Normalengleichungssystem für Wertepaare $(y_i, x_{1i}, ..., x_{ki})$ liefert

$$\begin{bmatrix} b_1 SQ_{X_1, X_1} + b_2 SQ_{X_1, X_2} + ... + b_k SQ_{X_1, X_k} \\ b_1 SQ_{X_2, X_1} + b_2 SQ_{X_2, X_2} + ... + b_k SQ_{X_2, X_k} \\ ... \\ b_1 SQ_{X_k, X_1} + b_2 SQ_{X_k, X_2} + ... + b_k SQ_{X_k, X_k} \end{bmatrix} = \begin{bmatrix} SQ_{X_1, Y} \\ SQ_{X_2, Y} \\ ... \\ SQ_{X_k, Y} \end{bmatrix}$$

Formel A-25: Normalengleichung der multiplen Regression

wobei die kleinsten Quadratschätzer $b_1,..,b_k$ sind, dabei ist

$$SQ_{U, V} = \sum_{i=1}^{k} (u_i - \bar{u})\ (v_i - \bar{v})$$

Formel A-26: Elemente der Normalengleichung

und für das Absolutglied gilt: $a = \bar{y} - (b_1 \bar{x}_1 + ... + b_k \bar{x}_k)$.

Das multiple Bestimmungsmaß ist demnach zu berechnen als:

$$B_{Y, (X_1,...,X_k)} = 1 - \frac{\sum_{i=1}^{n} \left(y_i - a - \sum_{j=1}^{k} b_j x_{ji} \right)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} = r^2_{Y, (X_1,...,X_k)}$$

Formel A-27: Bestimmtheitsmaß der multiplen Regression

Diese gibt den Anteil der Varianz des Merkmales Y an, der durch die Regression erklärt werden kann. Sie entspricht den Quadrat der multiplen Korrelationskoeffizienten. Da auch $X_1, ..., X_k$ untereinander korreliert sind gilt stets:

$$B_{Y, X_i} \leq B_{Y, (X_1,...,X_k)} \leq \sum_{j=1}^{k} B_{Y, X_j}$$

Formel A-28: Abschätzung des Bestimmtheitsmaßes

und die Regressionskoeffizenten ändern sich bei Berücksichtigung eines weiteren Merkmals.

Setzt man:

$$\tilde{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix}, \; X = \begin{bmatrix} 1 & x_{11} & .. & x_{k1} \\ 1 & x_{12} & .. & x_{k2} \\ \dots & \dots & .. & .. \\ 1 & x_{1n} & .. & x_{kn} \end{bmatrix}, \; e = \begin{bmatrix} e_1 \\ e_2 \\ \dots \\ e_n \end{bmatrix}, \; \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \dots \\ \beta_k \end{bmatrix}$$

Formel A-29: Matrizenansatz für multiple Regression

so läßt sich die allgemeine Modellgleichung in Matrixdarstellung schreiben, als $\tilde{Y} = X\beta + e$ mit dem Fehlervektor e. Für diesen gilt: $E(e_i) = 0$ und $Var(e_i) = \sigma^2$

Der Schätzvektor b für β ergibt sich somit aus den Normalengleichungen als:

$$X^T X b = X^T \tilde{Y}$$

Formel A-30: Gleichung der multiplen Regression

und mit $X^T X$ invertierbar ergibt sich ein eindeutiger Schätzer für β. Bildet man jetzt

$$b = (X^T X)^{-1} X^T \tilde{Y}$$

Formel A-31: Schätzvektor der multiplen Regression

eine Zwischengröße

$$c = (X^T X)^{-1} = \begin{bmatrix} c_{00} & \dots & c_{0k} \\ c_{10} & \dots & c_{1k} \\ \dots & \dots & \dots \\ c_{k0} & \dots & c_{kk} \end{bmatrix}$$

Formel A-32: Hilfsmatrix der Multiplen Regression

so gilt: $Eb_i = \beta$, $Var(b_i) = \sigma^2 \cdot c_{ii}$ und $COV(b_i, b_j) = \sigma^2 \cdot c_{ij}$

Der Streuparameter σ^2 kann dabei abgeschätzt werden durch

$$s^2 = \frac{1}{n-k-1} \cdot \sum_{i=1}^{n} \left(y_i - a - \sum_{j=1}^{k} b_j x_{ji} \right)^2$$

Formel A-33: Schätzer für Varianz

Ist der Fehlervektor normalverteilt, gilt bei vorgegebenen Niveau γ

$$\frac{\left| b_i - \beta_i^0 \right|}{\sqrt{s^2 \cdot c_{ii}}} > t_{n-k-1;1-\gamma/2}$$

Formel A-34: Schätzer bei Normalverteilung

Das $1-\gamma$ Konfidenzintervall für β_i ergibt sich als $\left[b_i \mp \left(\sqrt{s^2 c_{ii}} \cdot t_{n-k-1;1-\gamma/2} \right) \right]$.

Das simultane Koeffizenzintervalle für die Regressionskoeffizenten ergibt sich als $\left[b_i \mp \left(\sqrt{(k+1)\, s^2 c_{ii} F_{k+1,\, n-k-1;1-\gamma}} \right) \right]$.

Anhang B
Korrelation

Die Korrelation liefert ein Maß für den Grad der Abhängigkeiten verschiedener Merkmale. Zu nennen ist hier der Test auf Signifikanz eines Zusammenhanges, das heißt, der linearen Unabhängigkeit sowie die Vergleiche zwischen Korrelationen verschiedener Merkmalskombinationen. Es werden die Pearson Korrelationskoeffizenten als auch die Rangkorelationskoeffizienten von Spearmann und Kendall angeschnitten.

Grundlegend muß zwischen den betrachteten Merkmalen ein nachvollziehbarer Zusammenhang bestehen, da sonst eine sogenannte Nonsens-Korrelation ermittelt wird. Vorsicht ist auch bei den sogenannten Scheinkorrelationen geboten, wenn zwei nicht korrelierende Merkmale über ein drittes korreliert wird. In diesem Fall sind partielle Koeffizienten anzuwenden; diese sind Bestandteil der multiplen Korrelationsanalyse. Sie sind ein Maß dafür, wie quantitativ gut sich ein Merkmal Y durch Merkmale $X_1,..., X_n$ erklären läßt.

B.1 Korrelation zweier normalverteilter Merkmale

Für die Korrelation von Zufallsvariablen X und Y gilt:

$$\delta = \text{CORR}(x, y) = \frac{\text{Kovarianz}(x, y)}{\sqrt{\text{Varianz}(x) \cdot \text{Varianz}(y)}} \in (-1, 1)$$

Formel B-1: Korrelation zweier Merkmale

mit der

$$\text{Kovarianz}(x, y) = \sum_{i=0}^{n} (x_i - \bar{x}) \bullet (y_i - \bar{y})$$

und der

$$\text{Varianz}(x) = \left(\sum_{i=0}^{n} (x_i - \bar{x}) \right)^2$$

wenn die Variablen X und Y unkorreliert sind dann gilt $\delta = 0$.

Als Schätzer bzw. Stichprobenkorrelation wird folgende Berechnung angewendet:

$$r_{XY} = \frac{s_{XY}}{s_X \cdot s_Y} = \frac{\sum_{i=0}^{n} (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\left(\sum_{i=0}^{n} (x_i - \bar{x})\right)^2 \cdot \left(\sum_{i=0}^{n} (y_i - \bar{y})\right)^2}}$$

$$= \frac{\sum_{i=0}^{n} (x_i \cdot y_i) - n\bar{x}\bar{y}}{\sqrt{\left(\sum_{i=0}^{n} x_i^2 - n\bar{x}^2\right) \cdot \left(\sum_{i=0}^{n} y_i^2 - n\bar{y}^2\right)}}$$

Formel B-2: Schätzer für zwei Merkmale

diese wird auch als *Pearson Korrelation* bezeichnet. Dies testet ob X unabhängig, wenn der Korrelationskoofizent $\Delta = 0$, ist so ist

$$t = \frac{r_{XY} \cdot \sqrt{n-2}}{\sqrt{1 - r_{XY}^2}}$$

Formel B-3: t-Verteilung

die Realisierung eine t-verteilte Zufallsvariable mit n-2 Freiheitsgraden. Die Pearson-schätzung r_{XY} für δ. Es geht also um den Test der Hypothese $H_0{:}\delta = 0$ gegenüber der Alternative $H_1{:}\delta \neq 0$ zum Niveau α zu testen. Die Hypothese H_0 ist zu verwerfen, wenn gilt

$$|t| > t_{n-2;\left(t - \frac{\delta}{2}\right)}$$

Formel B-4: Bedingung für Hypothese $H_0{:}\delta = 0$

Wird aber im Gegensatz dazu gegen die vorhandende Korrelation getestet mit $\delta_0 \neq 0$ so testen wir Hypothese $H_0{:}\delta = \delta_0$ gegen die Alternative $H_1{:}\delta \neq \delta_0$. Dazu wird eine Fischer Z-Transformation mit dem Schätzwert r_{XY} durchgeführt.

$$z = \operatorname{atanh}(r_{XY}) = \frac{1}{2} \cdot \ln\left(\frac{1 + r_{XY}}{1 - r_{XY}}\right)$$

Formel B-5: Fischer Z-Transformation

Mit der Variablen z, diese ist auch für kleine n recht gut approximativ normalverteilt. Mit dem Erwartungswert:

$$E_z = \frac{1}{2} \cdot \ln\left(\frac{1+\delta}{1-\delta}\right) + \frac{\delta}{2 \cdot (n-1)}$$

Formel B-6: Erwartungswert mit Fischer Z Transformation

und der Varianz $V_z = \frac{1}{n-3}$ daher ist $(z - E_z) \cdot \sqrt{n-3}$ normalverteilt. Die Hypothese H_0 muß dann verworfen werden wenn mit der $(1-\alpha/2)$ Quantile $u_{1-\alpha/2}$ der Standardverteilung gilt:

$$\left|(z - E_{z_0}) \cdot \sqrt{n-3}\right| > u_{1-\alpha/2}$$

Formel B-7: Test der Hypothese H_0 mit $(1-\alpha/2)$ Quantile

mit

$$E_{z_0} = \frac{1}{2} \cdot \ln\left(\frac{1+\delta_0}{1-\delta_0}\right) + \frac{\delta_0}{2\,(n-1)}$$

Formel B-8: Erwartungswert nach Fischer Z Transformation an der Stelle z_0

Diese Transformation dient dazu, $1-\alpha$ Konfidenzintervall für δ zu nutzen.

$$\left[\left(E_{z_0} = \frac{1}{2} \cdot \ln\left(\frac{1+\delta_0}{1-\delta_0}\right) + \frac{\delta_0}{2\,(n-1)}\right) \mp \frac{u_{1-\alpha/2}}{\sqrt{n-3}}\right]$$

Formel B-9: Konfidenzintervall

Zur Berechnung der Quantile wird entweder eine Tabelle oder die Approximation nach Hasting benutzt.

Sind $X_1,...,X_n$ normalverteilte Zufallsvariablen dann gilt:

$$R = \begin{bmatrix} 1 & r_{X_1X_2} & \dots & r_{X_1X_n} \\ r_{X_2X_1} & 1 & \dots & r_{X_2X_n} \\ \dots & \dots & \dots & \dots \\ r_{X_nX_1} & \dots & r_{X_nX_{n-1}} & 1 \end{bmatrix}$$

Formel B-10: Korrelationsmatrix nach Pearson für n Dimensionen

Für den Fall von 3 Merkmalen gilt also:

$$R = \begin{bmatrix} 1 & r_{XY} & r_{XZ} \\ r_{XY} & 1 & r_{YZ} \\ r_{XZ} & r_{YZ} & 1 \end{bmatrix}$$

Formel B-11: Korrelationsmatrix nach Pearson für 3 Dimensionen

Möchte man prüfen ob die Korrelation δ_1 und δ_2 von den beiden gewählten Methoden zum Signifikanztest α identisch sind, dann wird dies wieder über eine Fischer Z-Transformation durchgeführt. $z_i = \operatorname{atanh}(r_i)$, mit i aus (1,2) mit r_i geschätzter Korrelation bei Stichprobe n_i für δ_i . Es ist dann:

$$T = \frac{z_1 - z_2}{\sqrt{\frac{1}{n_1 - 3} + \frac{1}{n_2 - 3}}}$$

Formel B-12: Test auf Identität

Für den Fall von $\delta_1 = \delta_2$ ist die Realisierung eine approximativ normalverteilte Zufallsvariable, das heißt im Testproblem $H_0{:}\delta_2 = \delta_2$ gegen $H_0{:}\delta_2 \neq \delta_2$ muß man die Nullhypothese verwerfen falls $|T| > u_{1-\alpha/2}$ gilt. Wird die Hypothese der Gleichheit der Korrelation δ_1 und δ_2 zu Niveau α nicht verworfen, so kann ein gemeinsamer Schätzer entwickelt werden:

$$\tilde{z} = \frac{(n_1 - 3) \cdot z_1 + (n_2 - 3) \cdot z_2}{n_1 + n_2}$$

Formel B-13: Gemeinsamer Schätzer bei zwei Merkmalen

Will man jetzt nicht die Gleichheit von zwei Korrelationen, sondern vielmehr die Homogenität, daß heißt, die Gleichheit von k Korrelationen $\delta_1,...,\delta_k$ überprüfen, bestimmt man zunächst aufgrund von Wertepaaren der Wertemengen $n_1,...,n_k$ Schätzer $r_1,...,r_k$, die man nach $z_1,...,z_k$ transformiert. Im Test zum Niveau α von $H_0:\delta_1 = ... = \delta_k$ gegen $H_1:\exists(i,j) \rightarrow \delta_i \neq \delta_j$ wird die Hypothese verworfen wenn gilt:

$$T = \sum_{i=1}^{n} z_i^2 (n_i - 3) - \frac{\left\langle \sum_{i=1}^{k} z_i (n_i - 1) \right\rangle^2}{\sum_{i=1}^{k} n_i} > \chi^2_{k-1,\,1-\alpha}$$

Formel B-14: Test der Hypothese

Dabei ist mit der $(1-\alpha)$ Quantile $\chi^2_{k-1,\,1-\alpha}$ der χ^2 Verteilung mit k Freiheitsgraden gegeben. Wird H_0 zum Niveau α nicht verworfen, so kann eine gemeinsame Variable $\tilde{z}$ gebildet werden die sich nach $\tilde{r}$ transformieren läßt.

$$\tilde{z} = \frac{\sum_{i=1}^{k} z_i (n_i - 1)}{\sum_{i=1}^{k} n_i}$$

Formel B-15: Gemeinsame Variable der beiden Merkmale

B.2 Rangkorrelation zweier Merkmale

Wird die Korrelationen von Zufallszahlen, die nicht normalverteilt sind geschätzen, so bilden die Koeffizenten der Rangkorrelation von Kendall und Spearman eine Alternative. Die Korrelation wird in diesem Fall nur über die Ranginformation, der entsprechenden Einträge, ermittelt.

B.2.1 Spearman Korrelationskoeffizenten

Werden die Rangzahlen $R(x_i)$, $R(y_i)$ für die n Realisationen der Zufallszahlen X,Y vergeben, so daß in jeder der beiden Reihen die kleinste Realisation den Rang 1 und die größte den Rang n erhält und umgekehrt, so folgt:

$$r_s = \frac{\sum_{i=1}^{n}\left(R(x_i) - \overline{R(x)}\right)\left(R(y_i) - \overline{R(y)}\right)}{\sqrt{\sum_{i=1}^{n}\left(R(x_i) - \overline{R(x)}\right)^2 \cdot \sum_{i=1}^{n}\left(R(y_i) - \overline{R(y)}\right)^2}}$$

$$= 1 - \frac{6\sum_{i=1}^{n}(R(x_i) - R(y_i))^2}{n(n^2-1)} = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \text{ mit } d_i = R(x_i) - R(y_i)$$

Formel B-16: Schätzer für zwei Merkmale mit Korrelationskoeffizenten nach Spearman

Treten mehrere Bindungen auf, das heißt es sind mehrere Realisationen einer Zufallsvariable genau gleich, so wird das arithmetische Mittel der in Frage kommenden Ränge genommen. Basierend auf den Spearman Rangkoeffizienten r_s kann die Unabhängigkeit von x_i,y_i mit $i \in (1, n)$ zum Niveau α getestet werden. Man verwendet anstelle von r_s die Hotteling Past Statistik: $D = \sum_{i=0}^{n} d_i = \sum_{i=0}^{n} R(x_i) - R(y_i)$

Die Hypothese H_0 (unabhängig) muß im Test gegen H_1 (abhängig) zum Niveau α verworfen werden falls $D < h_{n;\alpha/2}$ oder $D > h_{n;(1-\alpha/2)}$ und es gilt:

$$h_{n;\alpha} = \frac{1}{3n(n^2-1)} - h_{n;(1-\alpha)}$$

Formel B-17: Test der Hypothese

Stehen die kritischen Werte $h_{n;\alpha}$ nicht zur Verfügung so macht man sich die Tatsache zunutze das unter H_0 gilt:

$$E(D) = \frac{1}{6}n(n^2-1) - \frac{1}{12}\sum_{j=1}^{p} d_{1j}(d_{1j}^2 - 1) - \frac{1}{12}\sum_{k=1}^{q} d_{2k}(d_{2k}^2 - 1)$$

Formel B-18: Erwartungswert der Spearman Korrelationskoeffizenten

$$V(D) = \frac{n^2(n-1)(n+1)^2}{36}\left(1-\frac{\sum_{j=1}^{p} d_{1j}(d_{1j}^2-1)}{n(n^2-1)}\right)\left(1-\frac{\sum_{k=1}^{q} d_{2k}(d_{2k}^2-1)}{n(n^2-1)}\right)$$

Formel B-19: Varianz der Spearman Korrelationskoeffizenten

mit p und q als Anzahl unterschiedlicher Werte von x_i und y_i und d_{1j} und d_{2k} Anzahl Beobachtungen, welche den j-ten dieser unterschiedlichen Werte haben. Man weiß, daß dann unter H_0:

$$T = \frac{D-E(D)}{\sqrt{Var(D)}} > u_{1-\alpha/2}$$

Formel B-20: Test der Hypothese mit Spearman Koeffizenten

approximative Standardnormalverteilt ist, daß heißt man kann die Hypothese verwerfen, falls $T > u_{1-\alpha/2}$ gilt, aber Vorsicht wenn die Hypothese H_0 nicht verworfen wird, sollte man sich davor hüten zu folgern, daß die Meßreihen unabhängig voneinander sind.

B.2.2 Kendall'sche Rangkoeffizienten

Der *Kendall'sche Rangkoeffizent* τ wird basierend auf den Rangzahlen gebildet. Die n Realisationen der Zufallszahlen X,Y bilden natürliche Beobachtungspaare $(x_i, y_i),...,(x_n, y_n)$. Werden nun Rangzahlen vergeben, so ordnet man zur Berechnung von Kendal τ so an, daß im ersten Paar der Rang der Realisation gerade 1 ist. Dadurch werden auch die Reihenfolge der Rangzahlen der Variablen Y festgelegt. In dieser Reihenfolge bestimmt man jetzt für jede Rangzahl $R(y_i)$ die Anzahl q_i der Rangzahlen $R(y_i)$ die kleiner oder gleich der ursprünglichen Rangzahlen sind und in der Reihenfolge hinter diesen stehen. Es ergibt sich somit:

$$\tau = 1-\left(\frac{4\cdot\sum_{i=1}^{n} q_i}{n(n-1)}\right)$$

Formel B-21: Rangkoeffizent nach Kendall

Ein Test auf Unabhängigkeit zweier Meßreihen X,Y läßt sich basierend auf Kendall τ durchführen denn:

$$K = \frac{1}{2}n(n-1) - 2\sum_{i=1}^{n} q_i = \frac{n(n-1)}{2} \cdot \tau$$

Formel B-22: Kendall'sche Statistik

ist gerade die Kendall'sche Statistik. Die kritischen Werte $K_{n;(1-\alpha)}$ lassen sich aus den Tabellen ermitteln. Die These H_0 wird verworfen, falls gilt: $|K| > K_{n;(1-\alpha/2)}$. Stehen keine Werte $K_{n;(1-\alpha)}$ zur Verfügung, so verwendet man die standardnormalverteilte Prüfung

$$K^* = \frac{K}{\sqrt{\frac{n(n-1)\ (2n+5)}{18}}}$$

Formel B-23: Prüfung der Kendall Rangkoeffizenten

mit der Bedingung $|K^*| > u_{1-\alpha/2}$

B.3 Partielle Korrelation

Oft läßt sich eine Korrelation zwischen 2 Meßreihen X,Y nur finden, weil beide Merkmale mit einem dritten Merkmal U korreliert sind. Die Korrelation zwischen X,Y ist dann eine reine Scheinkorrelation. Daher ist es oftmals von Interesse Korrelationen zwischen X und Y, unter Partialisierung eines Merkmales U, d.h. die Korrelation zwischen X und Y die ohne Einfluß von U vorhanden ist, zu bestimmen. Eine solche Korrelation $\delta_{(X,Y)/U}$ wird auch als partielle Korrelation von X,Y unter U bezeichnet.

$$\delta_{(X,Y)/U} = \frac{\delta_{XY} - \delta_{XU}\delta_{YU}}{\sqrt{(1-\delta_{XU}^2) \cdot (1-\delta_{YU}^2)}}$$

Formel B-24: Partielle Korrelation

B.3.1 Partielle Korrelation mit normalverteilten Merkmalen

Als Schätzer kann dann der *Pearson Korrelationskoeffizent* verwendet werden.

$$r_{(X,Y)/U} = \frac{r_{XY} - r_{XU} r_{YU}}{\sqrt{(1 - r_{XU}^2) \cdot (1 - r_{YU}^2)}}$$

Formel B-25: Koeffizienten der partiellen Korrelation nach Pearson

Die Hypothese $H_0: \delta_{(X,Y)/U} = 0$ wird im Test gegen die Alternative H_1 zum Niveau α verworfen wenn gilt:

$$\left| \frac{t_{(X,Y)/U} \cdot \sqrt{n-3}}{\sqrt{1 - r_{(X,Y)/U}^2}} \right| > t_{n-3;(1-\alpha/2)}$$

Formel B-26: Test der Hypothese bei partieller Korrelation

Die Quantile $t_{v;y}$ der t-Verteilung mit v Freiheitsgeraden können fest abgelegt werden oder sind dynamisch ermittelbar.

B.3.2 Partielle Korrelation nach Kendall

Sind die Merkmale zumindest ordinal skaliert, so kann der Schätzer von Kendall verwendet werden.

$$\tau_{(X,Y)/U} = \frac{\tau_{XY} - \tau_{XU} \tau_{YU}}{\sqrt{(1 - \tau_{XU}^2) \cdot (1 - \tau_{YU}^2)}}$$

Formel B-27: Schätzer für partielle Korrelation nach Kendall

B.4 Bi-partielle Korrelation

Sind die Merkmal X mit U und Y mit V korreliert und besteht eine Korrelation zwischen U und V, so wird die Korrelation zwischen X,Y von U,V stark beeinflußt. Will man anhand einer Wertemenge die Korrelation zwischen X,Y schätzen, so empfiehlt es sich, die Merkmale X von V und Y von U zu partialisieren. Diese Korrelation heißt bipartielle Korrelation von X unter Partialisierung von U und Y unter Partialisierung von V

$$\delta_{X/U,\,Y/V} = \frac{\delta_{XY} - \delta_{XU}\delta_{YV} - \delta_{XV}\delta_{YU} + \delta_{XU}\delta_{UV}\delta_{YV}}{\sqrt{(1-\delta_{XU}^2)\,(1-\delta_{YV}^2)}}$$

Formel B-28: Pearson-Koeffizienten bei bi-partieller Korrelation

Bei Normalverteilung gilt:

$$\tau_{X/U,\,Y/V} = \frac{\tau_{XY} - \tau_{XU}\tau_{YV} - \tau_{XV}\tau_{YU} + \tau_{XU}\tau_{UV}\tau_{YV}}{\sqrt{(1-\tau_{XU}^2)\,(1-\tau_{YV}^2)}}$$

Formel B-29: Kendall-Koeffizenten bei bi-partieller Korrellation

Zum Niveau α läßt sich die Hypothese $H_0{:}\delta_{(X,\,Y)\,/U} = 0$ gegen die Alternative H_1 verwerfen wenn gilt:

$$\left|\frac{r_{X/U,\,Y/V}\cdot\sqrt{n-3}}{\sqrt{1-r_{X/U,\,Y/V}^2}}\right| > t_{n-3;\,(1-\alpha/2)}$$

Formel B-30: Test der Hypothese bei bi-partieller Korrellation

B.5 Multiple Korrelation

Die multiple Korrelation ist ein Maß für die Abhängigkeit eines Merkmals Y von p anderen Merkmalen $X_1,..,X_p$, dabei wird vorausgesetzt, daß die Merkmale in ihrer Gesamtheit normalverteilt sind. Sie ist definiert als die betragsmäßig größte Korrelation unter den Korrelationen zwischen dem Merkmal Y und allen möglichen Linearkombinationen $\sum_{i=1}^{p} a_i X_i$ der Merkmale X_i mit der Gewichtung a_i.

Will man die multiple Korrelation $\delta_{Y,\,(X_1,...,X_p)}$ abschätzen, so schätzt man zunächst alle möglichen einfachen Korrelationen δ_{YX_i} und $\delta_{X_iX_j}$ mittels den Koeffizienten nach Pearson ab.

Für $p = 1$ ergibt sich so $r_{Y,\,(X_1)} = \left|r_{Y,\,(X_1)}\right|$ und für $p = 2$ ergibt sich:

$$r_{Y,\,(X_1,\,X_2)} = \sqrt{\frac{r_{YX_1}^2 + r_{YX_2}^2 - 2r_{YX_1}r_{YX_2}r_{X_1X_2}}{1 - r_{X_1X_2}^2}}$$

Formel B-31: Pearsonkoeffizienten bei multipler Korrelation mit Dimensionsgrad 2

Allgemein gilt: $r_{Y,\,(X_1,..,X_p)} = \sqrt{r_{YX}^T R_{XX}^{-1} r_{YX}}$

$$r_{Y,\,(X_1,...,X_p)} = \left[\left[r_{YX_1}, ..., r_{YX_p} \right] \begin{bmatrix} 1 & r_{X_1X_2} & ... & r_{X_1X_p} \\ r_{X_2X_1} & 1 & ... & r_{X_2X_p} \\ ... & ... & ... & ... \\ r_{X_pX_1} & ... & r_{X_pX_{p-1}} & 1 \end{bmatrix}^{-1} \begin{bmatrix} r_{YX_1} \\ r_{YX_2} \\ ... \\ r_{YX_p} \end{bmatrix} \right]^{\frac{1}{2}}$$

Formel B-32: Pearsonkoeffizienten bei multipler Korrelation mit Dimensionsgrad n

Die Größe $B = r_{Y,\,(X_1,...,X_p)}$ wird auch das *Bestimmungsmaß* der multiplen Regression von Y genannt. Es ist ein Maß dafür wie gut das Merkmal Y sich durch die Merkmale $X_1,...,X_p$ erklären läßt. B gibt den Anteil der Varianz wieder, der durch die anderen p Merkmale erklärt werden kann. Es gilt die multiple Korrelation ist genau dann gleich 0 wenn alle einfachen Korrelationen gleich 0 sind. Will man die Hypothese $H_0: \delta_{Y,\,(X_1,...,X_p)} (= \delta_{YX_1} = ... = \delta_{YX_p}) = 0$ zum Niveau α gegen die Alternative testen, so verwendet man die Prüfgröße:

$$F = \frac{r_{Y,\,(X_1,...,X_p)}\,(n - 1 - p)}{\langle 1 - r_{Y,\,(X_1,...,X_p)}^2 \rangle \cdot p}$$

Formel B-33: Prüfgröße

Die Hypothese wird verworfen falls gilt,

$$F > F_{p,\,n-1-p;\,(1-\alpha)}$$

Formel B-34: Test der Hypothese

wobei $F_{p,\,v;\gamma}$ die γ Quantile der F-Verteilung bezeichnet; diese läßt sich iterativ berechnen.

B.6 Test auf Unabhängigkeit von p - Meßreihen

Die Beobachtungen $x_{i_1},...,x_{i_n}$ der i. Meßreihe mögen einer Normalverteilung $N(\mu_i, \sigma_i^2)$ entstammen. Dabei gilt für den Index $i = 1,...,p$ und die Meßreihen i,k paarweise verschieden δ_{ik} besitzen, dann kann mittels multipler Vergleiche nach dem Prinzip von Holm zum Niveau α die Nullhypothese $H_{0i}:\delta_{ik} = 0$ gegen $H_{0i}:\delta_{ik} \neq 0$ für $1 < i < k < p$ getestet werden.

$$r_{ik} = \frac{\sum_{j=1}^{n}\left(x_{ij} - \overline{x_i}\right)\left(x_{kj} - \overline{x_k}\right)}{\sqrt{\left(\sum_{j=1}^{n}\left(x_{ij} - \overline{x_i}\right)^2\right)\left(\sum_{j=1}^{n}\left(x_{kj} - \overline{x_k}\right)^2\right)}}$$

Formel B-35: Test auf Unabhängigkeit

Es handelt sich um ein simultanes Testen auf Paarweise Unabhängigkeit von p Meßreihen und ordnet dann die $p(p-1)/2$ nach abfallender Größe $R_{i_1k_1} \geq R_{i_2k_2}$ $R_{i_1k_1} \geq ... \geq R_{i_{p(p-1)/2}k_{(p(p-1))/2}}$ und überprüfen suzzessive ob für $m = 1,...,p(p-1)/2$ gilt:

$$R_{i_mk_m} < t_{n-2;1-\frac{\alpha}{p(p-1)+2-2m}}$$

Formel B-36: Test der Hypothese

Ist dies für m der Fall, so sind die Korrelationen $\delta_{i_1k_1},...,\delta_{i_{m-1}k_{m-1}}$ signifikant verschieden von 0 zum Niveau α zu betrachten. Im Gegensatz dazu sind nicht alle verbleibenden Paare signifikant verschieden von 0. Die Paare $(i_1k_1),...,(i_{m-1}k_{m-1})$ von je zwei Meßreihen sind demnach nicht unabhängig. Läßt sich ein solches m nicht finden, so sind die Meßreihen als Paarweise unabhängig zu betrachten.

Anhang C
Statistik zur Implementierung

C.1 Effektive und absolute Programmierzeilen

Die Implementierungsphase dauerte von Anfang Juni 1996 bis November 1996. Zu Beginn konnten einige elementare Klassen implementiert werden. In der Implementierung wurden 80.248 Programmzeilen (LOC) erstellt, wobei die Kommentare mitgerechnet wurden, da der Autor Wert auf umfangreiche Kommentierung und einheitlichen Codierungstil gelegt hat. Es muß hier mit einem Faktor zwischen 0.3 und 0.5 gerechnet werden um eine effektive Anzahl von Codezeilen zu erhalten (ELOC). Zur Berechnung der Programmzeilen wurde ein Unixskript verwendet.

Es wurden ca. 476 Klassen implementiert, was einer durchschnittlichen Klassengröße von 168 Programmzeilen entspricht.

Eine nähere Analyse der Klassengröße:

- 92 Klassen 0..50 LOC
- 148 Klassen 51 .. 100 LOC
- 137 Klassen 101 .. 150 LOC
- 56Klassen 151 .. 200 LOC
- 30Klassen 201 .. 250 LOC
- 30 Klassen 250 .. 500 LOC
- 19 Klassen 500 .. 1000 LOC
- 7 Klassen 1000 .. 2000 LOC

Hiermit versichere ich, daß ich diese Arbeit selbständig verfaßt und bei der Erstellung nur die angegebenen Hilfsmittel verwendet habe.

_____

Sven Van Steenkiste