Unterschrift (Betreuer)

TECHNISCHE UNIVERSITÄT WIEN
VIENNA UNIVERSITY OF TECHNOLOGY

MEDIZINISCHE UNIVERSITÄT WIEN

MAGISTERARBEIT

Ein graphischer Anfragegenerator für klinische Datenbanken

Ausgeführt am Institut für

Medizinische Informations- und Auswertesysteme

der Besonderen Einrichtung für Medizinische Statistik und Informatik
an der Medizinischen Universität Wien
für die
Technische Universität Wien

unter der Anleitung von

Ao. Univ.-Prof. Dr. Walter Gall

durch

Predrag Steric, Bakk. techn.

Kundratstraße 16/4/6
A-1100 Wien

Wien, am _20.05.2007___

Unterschrift (Student)

Zusammenfassung

In der Medizin fallen einerseits große Mengen an Patientendaten an — die klinischen Patientendaten mancher Krankenhäuser umfassen mehrere Millionen Datensätze — und müssen mit großem Aufwand in riesigen elektronischen Datenbanken abgelegt werden, andererseits wird in Krankenhäusern, nicht zuletzt auch aus wirtschaftlichen Gründen, Forschung betrieben und diese Datensätze stellen einen großen Datenschatz dar. Die elektronischen Datenbanken werden in einer eigenen Sprache abgefragt, mit der die Mediziner nicht zwangsweise vertraut sind. Auf der Suche nach einer Lösung dieses Dilemmas wird die Idee eines vorgeschalteten Anfragegenerators verfolgt. Dieser soll einerseits das Schema der darunter liegenden Datenbank und ihrer Zugriffslogik vollkommen verstecken und andererseits eine intuitive Anfragegenerierung ermöglichen.

Diese Arbeit macht im Laufe der einzelnen Kapitel einen Bogen angefangen bei Archimed, der Basis der Magisterarbeit, weitergehend über die Grundlagen des zugrunde liegenden Themas, die die Ideen der visuellen Anfragegenerierung umsetzen helfen, bis hin zu Details der Implementierung eines entsprechenden Prototypen.

Kapitel 1 macht eine erste Einführung in das Thema. Es führt in die Datenanalysesysteme der klinischen Forschung ein und definiert grobe Anforderungen an diese Arbeit. Kapitel 2 gibt eine Beschreibung über Archimed dem medizinischen Informations- und Auswertesystem am AKH Wien – der Basis dieser Arbeit. Kapitel 3 stellt einige grundlegende Begriffe aus den Bereichen Datenanalyse, visuelle Anfragegenerierung sowie zur Nutzung von Metadaten und Metaphern im Rahmen der Modellierung einer Benutzeroberfläche zur visuellen Datenanalyse vor. Kapitel 4 geht anschließend auf bestehende Datenanalysesoftware und einige Beispielsysteme ein, um daraus Anforderungen an diese Arbeit im Kapitel 5 abzuleiten, die zusammen mit den Anforderungen aus Kapitel 1, im Kapitel 6 zum visuellen Interface VISAmed (**V**isual **I**nterface **S**upporting **A**d-hoc queries in **med**icine) zusammengeführt werden. Eine Bewertung und ein Ausblick auf die Nutzung der vorgestellten Konzepte runden die Arbeit im Kapitel 7 ab.

Für Sabine und Stephanie

Vorwort

Die höchste Pflicht des Intellektuellen unserer Zeit ist es,
die einfachsten Wahrheiten in den einfachsten Worten
auszusprechen.

George Orwell

In Sinne des obigen Zitats zu Beginn ein Wort zur sprachlichen Gestaltung der Arbeit. Nicht jeder ist ein Schriftsteller — Wissenschaftler sind das in der Regel überhaupt nicht und Studenten noch weniger. Ich werde mich um eine verständliche Darstellung bemühen und versuchen nicht in den einschlägigen wissenschaftlichen Jargon zu verfallen, den wir vor allem im deutschsprachigen Raum aus der wissenschaftlichen Literatur kennen.

In der Medizin fallen einerseits große Mengen an Patientendaten an — die klinischen Patientendaten mancher Krankenhäuser umfassen mehrere Millionen Datensätze — und müssen mit großem Aufwand in riesigen elektronischen Datenbanken abgelegt werden, andererseits wird in Krankenhäusern, nicht zuletzt auch aus wirtschaftlichen Gründen, Forschung betrieben und diese Datensätze stellen einen großen Datenschatz dar. Frei nach dem Motto — *Finde Gold in Deinen Daten* — ist jeder forschende Mediziner angehalten diesen Datenschatz zu nutzen, wenn da bloß nicht ein Problem wäre — die elektronischen Datenbanken wollen in einer eigenen Sprache abgefragt werden, mit der die Mediziner nicht zwangsweise vertraut sind. Auf der Suche nach einer Lösung dieses Dilemmas wurde die Idee eines vorgeschalteten Anfragegenerators verfolgt. Dieser sollte das Schema der darunter liegenden Datenbank und ihrer Zugriffslogik vollkommen verstecken.

Dieser Ansatz eines vorgeschalteten Anfragegenerators ist auch im Archimed, dem medizinischen Informations- und Auswertesystem am AKH Wien realisiert. Das Archimed-Auswertesystem ist der Ausgangspunkt dieser Arbeit. Es stellt, in Verbindung mit SAS, mächtige Funktionen für medizinisch statistische Auswertungen zur Verfügung. Dennoch ist die Bildung von Patientenkollektiven, als ersten Schritt einer Auswertung, für den forschenden Mediziner nicht ganz einfach. Für komplexe Kollektivbildungen sind unterstützende Techniken vorzusehen, die etwa das Problem der richtigen Bildung logischer Konstrukte mindern. Wichtig sind intuitive und interaktive Bedienbarkeit, sowie unterstützende Tech-

niken die eine Strukturierung der Abläufe ermöglichen und die vorgenommenen Auswertungsfolgen für den Anwender transparent und flexibel modifizierbar machen. Das ist keine einfache Aufgabe und es gibt noch keine zufrieden stellende Lösung.

Ausgehend vom bestehenden Anfragegenerator Archimed, wird in dieser Arbeit mit VISAmed (**V**isual **I**nterface **S**upporting **A**d-hoc queries in **med**icine) eine mögliche Lösung diskutiert.

Die vorliegende Arbeit zeigt, dass der Entwurf und die Realisierung eines neuen Anfragegenerators ein aufwendiges und ambitioniertes Vorhaben ist. Im Rahmen *einer* Magisterarbeit kann das Vorhaben nicht in vollem Umfang geleistet werden. Dennoch konnten im Verlauf der Arbeit die visuellen Konzepte und die graphischen Konstrukte von VISAmed definiert werden.

Gleichzeitig zeigte sich, dass typisch medizinische Probleme, wie die sehr hohe Datenkomplexität, die Navigation in Metadaten und die Unerfahrenheit der medizinischen Forscher im Umgang mit Datenbankabfragen, einer besonderen Betrachtung bedürfen.

Letztendlich bleibt das Ziel dieser Arbeit, eine Benutzerschnittstelle für das System Archimed, zur komfortablen, flexiblen und interaktiven Bildung komplexer Kollektive multidimensionaler Daten in klinischen Datenbanken zu diskutieren. Es wird weiterer Forschung und findiger Ideen bedürfen um sich an eine optimale Lösung für die Bedürfnisse der forschenden Mediziner heranzuarbeiten.

Danksagung

Zunächst gilt mein Dank dem Institut für Medizinische Informations- und Auswertesysteme der Medizinischen Universität Wien, an dem ich die Möglichkeit hatte, an diesem Thema zu arbeiten. Im speziellen bedanke ich mich bei meinem Betreuer Ao. Univ.-Prof. Dr. Walter Gall, der diese Arbeit ermöglichte und mir die notwendigen Freiheiten ließ um meine Magisterarbeit zu verfassen.

Meiner Familie möchte ich für viel Zeit, Geduld und moralische Unterstützung danken.

Inhalts-verzeichnis

KAPITEL

1

Einleitung

If we knew what it was we were doing,
it wouldn't be called research.

Albert Einstein

Das erste Kapitel gibt einen Gesamtüberblick und einige motivierende Erläuterungen, die den Einstieg in die Arbeit erleichtern sollen. So weit zum Verständnis hilfreich wird in einigen Punkten der eigentlichen Behandlung den zugehörigen Kapiteln vorgegriffen.

1.1. Motivation

Die Suche nach Daten in einem medizinischen Informationssystem erfordert oft vom Fragesteller gute Kenntnisse von Datenbankanfragesprachen.

Rechnergestützte Informationssysteme gehören heute zur Grundausstattung der Krankenhäuser. In einem Informationssystem werden Daten digital erfasst, redigiert, gespeichert und analysiert sowie graphisch und alphanumerisch präsentiert. Trotz dieses Anspruchs ist die Suche nach Daten in einem Informationssystem oft mühsam. Der Fragesteller muss in der Regel die Struktur und den Wortschatz der zugrunde liegenden Datenbank kennen und die Syntax der Anfragesprache gut beherrschen.

Informationssysteme stellen dem Anwender zumindest formale Anfragesprachen, wie beispielsweise SQL, die von der Datenbank direkt unterstützt werden zur Verfügung. Mit diesen Anfragesprachen kann man Informationen aus einer Datenbank durch textorientierte Anfragen präzise abrufen,

vorausgesetzt werden jedoch gute Kenntnisse und Erfahrung im Umgang mit der Anfragesprache. Gelegenheitsbenutzern wird der Zugang zu solchen Informationssystemen dadurch erschwert.

Wissenschaft extrahiert Daten aus Informationen.

Die grundlegende Aufgabe der Wissenschaften, speziell auch der Medizin, besteht darin, aus Daten Informationen zu extrahieren. In der Medizin fielen schon immer große Datenmengen an, aber mit den steigenden Speicher-, Transfer- und Verarbeitungskapazitäten moderner Rechenanlagen wird dieser Trend noch verstärkt. Aus der Auswertung der gesammelten Daten erhofft man sich neue Erkenntnisse. Die Datenanalyse liefert die notwendigen Hilfsmittel dazu.

Medizinische Datenanalyse bedarf der Zusammenarbeit verschiedener Fachdisziplinen und der Unterstützung durch die Technik.

Bei der Analyse medizinischer Datenbestände wirkt eine ganze Reihe unterschiedlicher Fachdisziplinen zusammen. Die jeweiligen medizinischen Disziplinen arbeiten zusammen mit Disziplinen aus dem Bereich der Statistik und Informatik. Eine möglichst gute Kombination dieser Fachgebiete ermöglicht es aus einem gegebenen Datenbestand die wirklich interessante Information zu extrahieren. Von zentraler Bedeutung ist dabei das Zusammenwirken von menschlichem Datenanalysten, hier beispielsweise Forscher, und dem rechnerbasierten Analysesystem. Die Forscher sind darauf angewiesen, sich entgegen ihres Aufgabengebietes mit Anfragesprachen und der Technologie der verwendeten Datenbanksysteme auseinanderzusetzen. Man mag versucht sein, dass Problem dadurch zu lösen, dass bestimmte anwendungsspezifische Anfragen fest implementiert werden und sozusagen auf Knopfdruck gestellt werden können, aber einige Anwendungen benötigen gerade die Flexibilität der Anfrageformulierung.

Informationssysteme müssen auch von Gelegenheitsbenutzern sinnvoll bedient werden können.

Medizinische Forschung steht vor der Aufgabe, die für ihre Aufgaben notwendigen Informationen aus großen, elektronisch gespeicherten Datenmengen wiederzugewinnen, über deren Inhalt und Strukturen sie nur Vermutungen anstellen kann. Aus diesem Grunde sollte ein Informationssystem so gestaltet sein, dass der Benutzer ohne Kenntnis des Datenbankschemas und der formalen Anfragesprache Zugang zu den interessierenden Daten bekommen kann. Damit dieses interaktive Zusammenspiel zwischen Computer und Benutzer reibungslos funktionieren kann, bedarf es einer geeigneten Schnittstelle. Dem Benutzer sollen ein umfassender Überblick und flexible Eingriffsmöglichkeiten in den Ablauf der Analyse angeboten werden, damit er die Datenanalyse voll ausschöpfen und die Analyseergebnisse korrekt interpretieren kann.

1.2. Einordnung der Arbeit

Diese Arbeit beschäftigt sich speziell mit der Betrachtung visueller Anfragemöglichkeiten in der Analyse multidimensionaler Daten. Alle Datenwerte in solchen Datenbeständen beziehen sich auf Gruppen von Individuen, hier spricht man allgemein von Makrodaten. Diese Makrodaten setzen sich zusammen aus Einzelwerten, die oft die Basisdaten bilden und Mikrodaten genannt werden. Forschungsgebiete die mit dieser Arbeit gestreift werden, fallen unter die Schlagworte wie "Data Warehousing" oder "Decision Support".

1.2.1. DataWarehousing

Data Warehousing — eine zentrale Datensammlung.

Aus dem Informationsmanagement in der Betriebswirtschaft stammt der Begriff Data Warehousing. Es handelt sich dabei um eine zentrale Datensammlung, meist eine Datenbank, deren Inhalt sich aus Daten unterschiedlicher Quellen zusammensetzt.

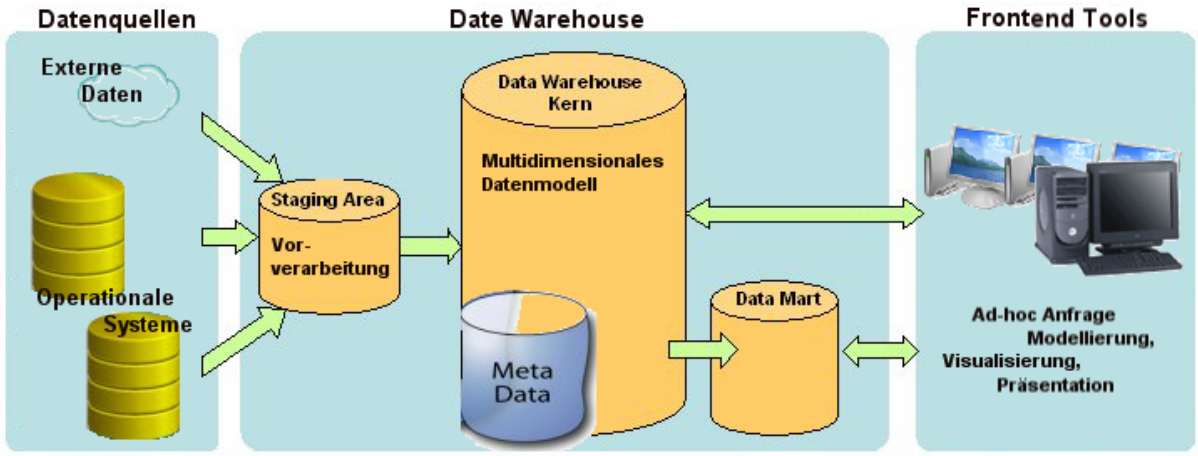


Abbildung 1.1: Ein Data Warehouse trennt entscheidungsunterstützende von den operativen Datenbeständen.

Es gibt keine verbindliche Definition von Data Warehousing. Inmon hat den Begriff Data Warehouse folgendermaßen definiert:

Definition von Inmon.

"*A data warehouse is a subject-oriented, integrated, time-variant, non-volatile collection of data management's decision-making process*" [Inmon1994].

Data Warehousing wird meist nur im betriebswirtschaftlich orientierten Umfeld eingesetzt.

Inmon beschränkt den Zweck eines Data Warehause auf die Entscheidungsunterstützung des Managements. Aus dieser Definition heraus ist auch die herrschende Vorstellung in den Köpfen nachzuvollziehen, dass Data Warehousing nur etwas ist für "Decision Support" und für das "Managementreporting". So werden die Methoden und Techniken des Data Warehousing fast nur im betriebswirtschaftlich orientierten Umfeld des mittleren und oberen Managements eingesetzt. Ein Data Warehouse sollte jedoch vielmehr immer dann er-

stellt werden, wenn eine integrierte Sicht auf bestimmte Datenobjekte erforderlich ist, unabhängig vom Zweck dieser integrierten Sicht.

Effektives Datenmanagement und Data Warehousing.

Aufgaben mit den integrierenden Methoden des Data Warehousing stellen sich häufig im Datenmanagement. Ein effektives Datenmanagement wiederum ist für wissensgebundene Dienstleistungen, wie etwa Forschung immer wichtiger. Es besteht also eine Lücke zwischen dem Potential der Data Warehousing Konzepte und der derzeitigen Praxis.

KIS ist ein Data Warehouse.

Ein KIS (Krankenhausinformationssystem) stellt ein Data Warehouse dar. Daten von externen Systemen, wie etwa LIS (Laborinformationssystem) werden zentral gesammelt. Anschließend können die Daten von verschiedenen Tools analysiert, modelliert, visualisiert, usw. werden. Die Abbildung 1.1 zeigt einen schematischen Aufbau eines Data Warehouse. Ein Data Warehouse kann neben der Gesamtsteuerung des Krankenhauses auf der Managementebene auch für Forschung genutzt werden.

OLAP Tools verwalten große Datenmengen und …

So genannte OLAP (OnLine Analytical Processing) Tools dienen der Verwaltung großer Datenmengen in einem Data Warehouse. Sie modellieren den betrachteten Datenbestand als eine Menge multidimensionaler Datenräume, sog. Datenwürfel[1], die durch kategorielle Attribute, sog. Dimensionen aufgespannt werden. Interessierende Kennzahlen bekommt man durch entsprechende Kombination von Kategorien auf verschiedenen Aggregationsebenen (Abbildung 1.2).

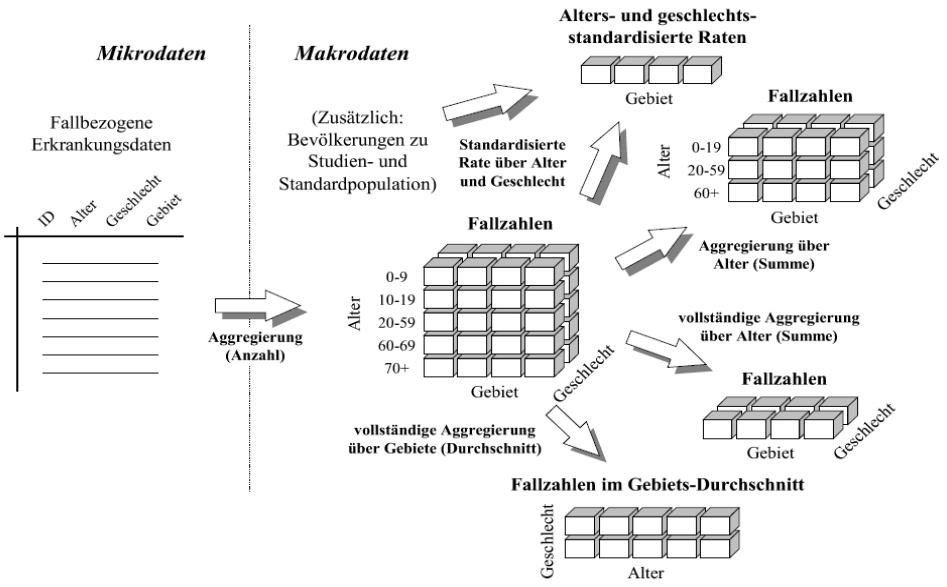


Abbildung 1.2: Varianten der Agreggierung (aus [Wietek2000], S. 53)

[1] Eigentlich handelt es sich um Datenquader, da die jeweiligen Dimensionen selten gleich groß sind.

OLAP Tools bieten eine möglichst einfache Bedienung für Anwender ohne Statistik- oder Programmierausbildung. Die Analyse mehrdimensionaler Datenräume ist gerade in der medizinischen Forschung relevant. So stellen etwa Neuerkrankungszahlen nach Alter, Geschlecht, betrachtetem Zeitraum und Art der Erkrankung typische multidimensionale Datenräume dar.

... bieten Anwendern eine einfache Bedienung bei der Analyse an.

1.2.2. Defizite von Datenanalysesystemen

Ein Analyseprozess besteht aus langen, aufeinander aufbauenden Sequenzen einzelner Analyseschritte. Der Benutzer kann dadurch oft den Überblick über die bereits durchgeführten Analyseschritte verlieren. Die Anwendung der Analysemethoden aber auch die Interpretation der Ergebnisse wird erschwert. Zwischen dem Entwurf einer Datenanalyse zur Bearbeitung eines Forschungsziels und der Umsetzung auf dem Rechner fehlt eine intuitive, kognitiv angemessene Repräsentation der Analyseschritte unter Nutzung geeigneter Interaktionsmetapher, die eine intuitive Manipulation von Analyseschritten ermöglichen, damit sich der Benutzer auf die eigentliche Analyse konzentrieren kann.

Anwendung der Analysemethoden und Interpretation der Ergebnisse wird durch mangelnden Überblick über durchgeführte Analyseschritte erschwert.

Bei der Kollektivbildung, als ersten Schritt bei der Durchführung einer Datenanalyse in der medizinischen Forschung, werden die Benutzer beispielsweise mit relationalen Modellen verwirrt, statt ihnen eine intuitiv verständliche Benutzersicht auf das logische Datenschema zu geben.

Bei der Eigenentwicklung von klinischen Applikationen in Krankenhäusern oder Universitäten steht meist kein Team von professionellen Entwicklern oder Designern zur Verfügung. Die Applikationen werden für eine sehr begrenzte Anzahl von Benutzern entwickelt oder die Benutzer stellen eine ganz bestimmte Gruppe von Experten dar. Da bei der Entwicklung der Applikationen der Fokus auf die Funktionalität der Software steht, wird die Schnittstelle zum Benutzer funktional gehalten, deren Design ist problem-orientiert. Benutzeroberflächen für Datenanalyse im klinischen Bereich sollten dem user-zentrierten Design folgen.

Problem-orientierter vs. user-zentrierter Entwurf der Systeme.

1.2.3. Graphische Interaktionstechniken als Lösungsansatz

Die Grundlage für die Kommunikation zwischen Benutzer und Analysesystem bildet die Visualisierung von Daten. Nur wenn der Benutzer sieht was er analysiert, kann er die richtigen Anfragen stellen.

Systeme die beliebige "ad-hoc" Anfragen erlauben haben als Grundlage standardisierte, eindimensionale Anfragesprachen, wie etwa SQL (Structured Query Language). Eindimensionale Anfragesprachen fordern jedoch vom

Eindimensionale Anfragesprachen ...

… überfordern ungeübte Anwender.

Benutzer Fähigkeiten, die die meisten forschenden Mediziner, die in dieser Arbeit primär betrachtet werden, nicht aufweisen. Neben der Beherrschung der Sprachsyntax und der Sprachsemantik, wird die Kenntnis des zugrunde liegenden Datenmodells und des anwendungsspezifischen Datenbankschemas vorausgesetzt. Der Benutzer muss die Anfragen exakt formulieren, vage Vorstellungen über die gesuchte Information reichen nicht aus.

Visuelle Anfragesprachen verbessern die Mensch-Maschine-Kommunikation.

Zur besseren Gestaltung der Mensch-Maschine-Kommunikation können graphische Interaktionstechniken eingesetzt werden. Im Kontext dieser Arbeit werden Datenbankanfragen mit Hilfe graphischer Sprachmittel spezifiziert. Man spricht dabei auch von visuellen oder mehrdimensionalen Anfragesprachen. Ihr Vorteil liegt in der Abstraktion vom zugrunde liegenden Datenbankschema.

Ziele dieser Arbeit

Ein Ziel dieser Arbeit ist die Entwicklung von Konzepten für die graphische Datenbankanfrage mit deren Hilfe ungeübte Anwender interaktiv Anfragen formulieren können. Deren Umsetzung, die visuelle Benutzerschnittstelle VISAmed (**V**isual **I**nterface **S**upporting **A**d-hoc queries in **med**icine) wird prinzipiell offen für die Ankopplung an beliebige Datenbanken sein.

1.3. Das Anwendungsgebiet der klinischen Forschung

Im Folgenden wird ein kurzer Einblick in grundlegende Begriffe und Vorgehensweisen der klinischen Forschung gegeben.

1.3.1. Definition der klinischen Forschung

Klinische Forschung wird nach Bedarfneu definiert.

Die Definition von klinischer Forschung hat Auswirkungen auf die finanzielle Förderung dieses Gebietes und wurde in den letzten Jahrzehnten dementsprechend unterschiedlich festgelegt. In einer *Denkschrift der Deutschen Forschungsgemeinschaft* von 1999 ([Forschung1999], S. 3) wird die klinische Forschung definiert als

- grundlagenorientierte Forschung, in deren Mittelpunkt der Erkenntnisgewinn in biologischen Systemen steht,
- krankheitsorientierte Forschung, die an Modellsystemen Einblick in die Pathophysiologie und in die genetischen Ursachen von Krankheiten zu gewinnen versucht, dazu aber nicht den direkten Kontakt mit dem Patienten benötigt und
- patientenorientierte Forschung, die direkt am und mit dem Patienten durchgeführt wird. Hierunter fallen vor allem klinische und epidemiologische Studien sowie Bereiche der Versorgungsforschung.

Der Begriff "klinische Forschung" wird in dieser Denkschrift [Forschung1999] als Oberbegriff für die grundlagenorientierte krankheitsorientierte und patientenorientierte Forschung verwendet.

Hinter dem Begriff "Klinische Forschung" stehen medizinische Arbeiten, die stark am Patienten orientiert sind und deren Ergebnisse direkt für die Krankenbehandlung nutzbar gemacht werden können. Ihre Aufgabe besteht im evaluieren der Wirksamkeit neuer Therapieformen oder Medikamente. Sie ermöglicht es auch, neue Therapieformen anzuwenden und deren Auswirkungen auf die Patienten zu überprüfen. Das Ziel ist, den Patienten eine möglichst effiziente Behandlung zu ermöglichen

Klinische Forschung steht für medizinische Arbeiten, die stark am Patienten orientiert sind und deren Ergebnisse direkt für die Krankenbehandlung nutzbar gemacht werden können.

1.3.2. Datenanalyse in der klinischen Forschung

Im Kontext dieser Arbeit ist speziell die patientenorientierte klinische Forschung von Interesse. Hierunter fallen vor allem klinische und epidemiologische Studien. Zu verschiedenen Populationen wird eine Reihe von Kennzahlen errechnet, die die Eigenschaften der betrachteten Patientengruppen beschreiben.

Im Wesentlichen sind folgende Arten von Kennzahlen zu unterscheiden:

Einige Kennzahlen aus der klinischen Forschung die bei Ad-hoc Anfragen genutzt werden

Die *Prävalenz* oder *Krankheitshäufigkeit* sagt aus, wie viele Menschen einer bestimmten Gruppe an einer bestimmten Krankheit erkrankt sind. Sie gibt eine absolute Häufigkeit an. Prävalenz kann folgendermaßen präzisiert werden [Chekoway2004]:

Prävalenz

- Die *Punktprävalenz* wird definiert durch einen genau bestimmten Zeitpunkt, wie z.B. "im Augenblick" oder "zum gegebenen Stichtag".
- Die *Periodenprävalenz* wird bestimmt durch einen Zeitraum wie "in den letzten 7 Tagen" oder im letzten Jahr (*Jahresprävalenz*).

Die *Prävalenzrate* ist eine relative Größe. Sie wird bestimmt durch die Zahl der Erkrankten im Verhältnis zur Zahl der Untersuchten und ist kleiner oder gleich 1.

Prävalenzrate

Wird nur die Zahl der Neuerkrankten betrachtet, spricht man von *Inzidenz*.

Inzidenz

Die *Inzidenzrate* ist die Anzahl der Neuerkrankungen (Inzidenz) dividiert durch die Individuenzahl. Das entspricht dem *relativen Risiko*. Diese Kennzahl hilft zu beschreiben, welche Krankheiten bei welcher Personengruppe häufig ausbrechen.

Inzidenzrate …

... *Attributionelles Risiko* hilft zu beurteilen, wie stark ein bestimmter Faktor zu einer bestimmten Erkrankung beiträgt.

Attributionelles Risiko

Risiko ist die Wahrscheinlichkeit während eines bestimmten Zeitraums an einer bestimmten Krankheit zu erkranken oder zu versterben. Risiken sind also zu einem erwarteten Vergleichswert in Beziehung gesetzte Raten.

Risiko

Raten sind die auf die jeweilige Bevölkerungsgruppe und einen bestimmten Beobachtungszeitraum bezogene *Fallzahlen*.

Raten

Regressionsanalysen versuchen Abhängigkeiten zwischen zwei oder mehr Variablen durch einfache Funktionen zu beschreiben.

Regressionsanalyse

Konfidenzintervalle beschreiben, basierend auf Irrtumswahrscheinlichkeiten, den Bereich rein zufälliger Schwankungen um den beobachteten Wert. Liegt der erwartete Wert innerhalb dieses Bereichs, kann nicht von einer signifikanten Abweichung gesprochen werden.

Konfidenzintervalle

Signifikanzniveaus geben die Irrtumswahrscheinlichkeit eines statistischen Tests an, mit der eine beobachtete Differenz zum Erwartungswert als signifikant bezeichnet werden kann.

Signifikanzniveau

Die genannten Kennzahlen werden beispielsweise bei Ad-hoc Anfragen genutzt. Dabei werden ermittelte Kennzahlen auf unterschiedliche Weise in Tabellen, Diagrammen und Graphiken visualisiert und einander gegenübergestellt.

1.3.3. Anforderungen an diese Arbeit

Zentrale Anforderungen an einen Anfragegenerator

Aus den typischen Arbeitsvorgängen der Datenanalyse in der patientenorientierten klinischen Forschung ergeben sich einige zentrale Anforderungen an einen graphischen Anfragegenerator und damit an das in dieser Arbeit zu konzipierende visuelle Interface VISAmed (**V**isual **I**nterface **S**upporting **A**d-hoc queries in **med**icine):

1. intuitiv verständliche Benutzersicht auf das logische Datenschema. Der Benutzer sollte nicht mit, beispielsweise, relationalen Modellen verwirrt werden.
2. einen mächtigen Anfragegenerator der auch komplexe Anfragen unterstützt.
3. enge Anbindung an statistische Funktionen, damit zu dem gefundenem Patientenkollektiv Auswertungen gemacht werden können.

Das visuelle Interface VISAmed, als Schnittstelle zum Benutzer, versucht die ersten beiden Anforderungen zu erfüllen.

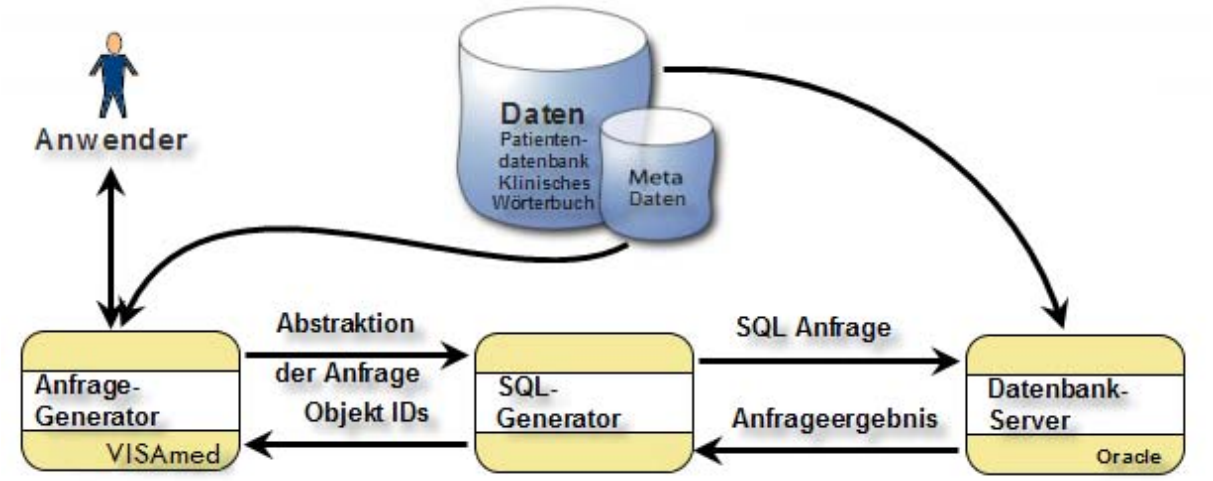


Abbildung 1.3: Einordnung von VISAmed als Anfragegenerator.

Der typische Anwender verfügt über Grundkenntnisse der Datenanalyse, hat keine vertiefte Kenntnisse in der Datenbankabfrage.

Es sollen verschiedene Benutzergruppen mit unterschiedlichen Kenntnisständen im Hinblick auf Datenanalyse und Datenbankabfragen unterstützt werden. Der typische Anwender verfügt zwar zumindest über Grundkenntnisse der Datenanalyse, hat aber nicht notwendigerweise vertiefte Kenntnisse in der Datenbankabfrage. Die Komplexität der Anfragen einerseits und die geringe Erfahrung der Benutzer in der Verwendung logischer Operatoren andererseits, führen oft zu Fehlern in der Abfrage die nicht unmittelbar erkennbar sind. Für komplexe Kollektivbildungen sind unterstützende Techniken vorzusehen, die das Problem der richtigen Bildung logischer Konstrukte, wie beispielsweise Einschließungen und Ausschließungen, mindern. Somit sind intuitive und interaktive Bedienbarkeit sowie unterstützende Techniken, die vorgenommenen Auswertungsfolgen für den Anwender transparent und flexibel modifizierbar machen, wichtig.

1.4. Überblick über die Arbeit

Die Arbeit macht im Laufe der einzelnen Kapitel einen Bogen angefangen bei der Basis der Magisterarbeit, weitergehend über die Grundlagen des zugrunde liegenden Themas, die die Ideen der visuellen Anfragegenerierung umsetzen helfen, bis hin zu Details der Implementierung eines entsprechenden Prototypen.

Kapitel 2 gibt eine Beschreibung über Archimed dem medizinischen Informations- und Auswertesystem am AKH Wien – der Basis dieser Arbeit. Kapitel 3 stellt einige grundlegenden Begriffe aus den Bereichen Datenanalyse, visuelle Programmierung sowie zur Nutzung von Metadaten und Metaphern im Rahmen der Modellierung einer Benutzeroberfläche zur visuellen Datenanalyse vor. Kapitel 4 geht anschließend auf bestehende Datenanalysesoftware und ein-

ige Beispielsysteme ein, um daraus Anforderungen an diese Arbeit im Kapitel 5 abzuleiten, die zusammen mit den Anforderungen aus Kapitel 1, im Kapitel 6 zum visuellen Interface VISAmed (**V**isual **I**nterface **S**upporting **A**d-hoc queries in **med**icine) zusammengeführt werden. Eine Bewertung und ein Ausblick auf die Nutung der vorgestellten Konzepte runden die Arbeit im Kapitel 7 ab.

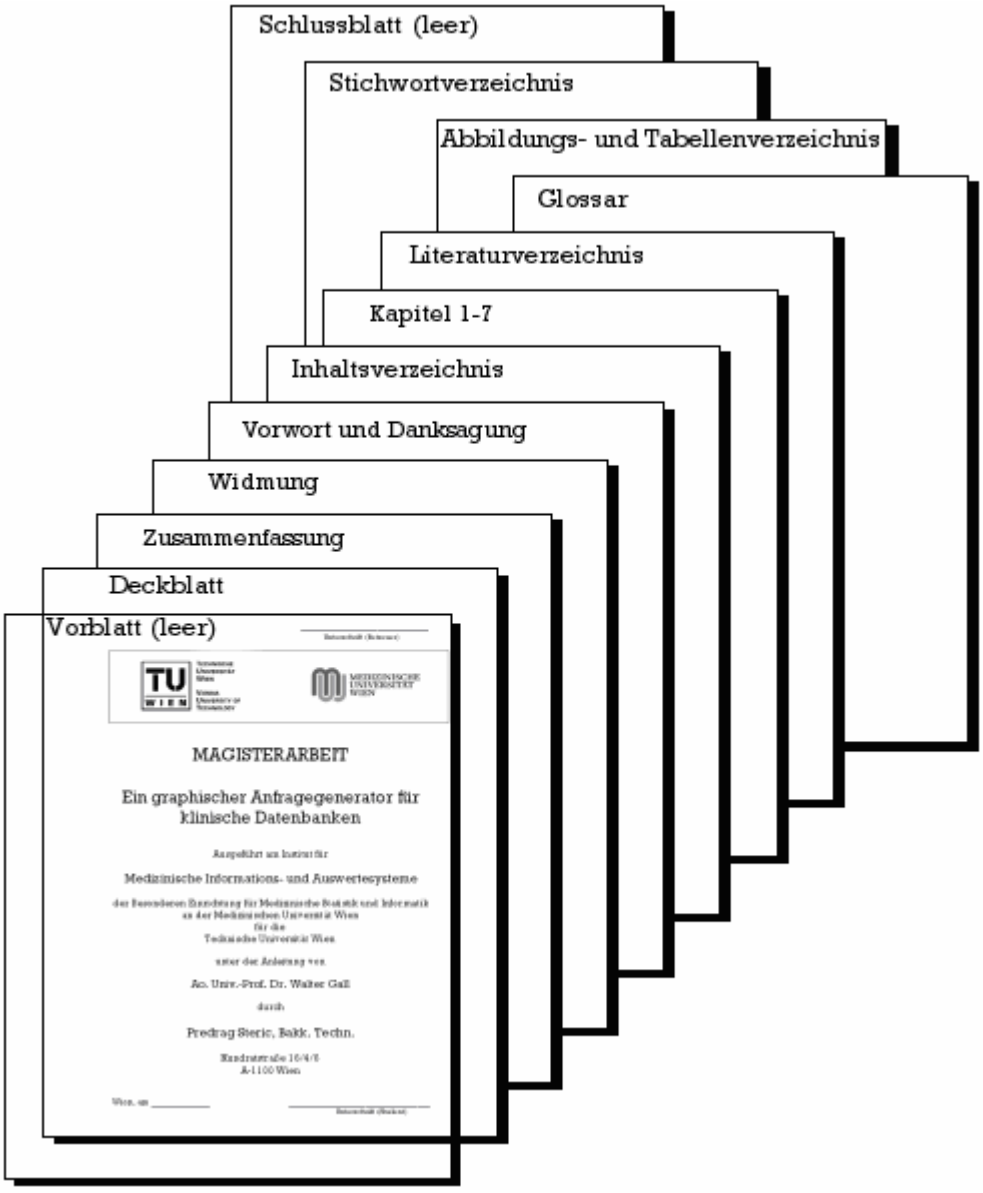


Abbildung 1.4: Aufbau der vorliegenden Magisterarbeit

KAPITEL

2

Basis der Magisterarbeit

Alles Gescheite ist schon einmal gedacht worden, man muß nur versuchen, es noch einmal zu denken...

Johann Wolfgang von Goethe

MEDIZINISCHE Informations- und Auswertesysteme bilden einerseits einen zentralen Arbeitsbereich der medizinischen Informatik, andererseits sind sie unverzichtbar für die moderne medizinische Forschung. Sie ermöglichen eine wissenschaftliche Nutzung aller erhobener Daten und liefern so Planungsunterlagen im Gesundheitswesen und können zur Qualitätskontrolle genutzt werden.

2.1. Archimed

Archimed ein, vom MSI an der Medizinisches Universität in Wien, entwickeltes medizinisches Informations- und Auswertesystem dient zur Unterstützung der klinischen Forschung.

Archimed ist ein medizinisches Informations- und Auswertesystem und dient zur Unterstützung der klinischen Forschung. Derzeit ist ArchiMed an den Universitätsklinken am AKH-Wien und LKH-Graz im Einsatz

Das System ArchiMed wurde am MSI[2] von der Abteilung MIAS[3] entwickelt. ArchiMed ist als Client-Server Anwendung implementiert. Der Datenzugriff erfolgt über das

[2] Die Besondere Einrichtung für Medizinische Statistik und Informatik (MSI) unterstützt die Forschungstätigkeit der Medizinischen Universität Wien durch Expertise in den Bereichen der Medizinischen Informatik und Statistik. http://www.meduniwien.ac.at/msi/

am AKH vorhandene Datennetz und unter verschiedenen Betriebssystemen.

Archimed in Wien

1997 erfolgte der Start des Betriebs an der Universitätsklinik für Chirurgie am AKH Wien. Von 2003 bis 2004 wurden andere wissenschaftliche Systeme (WAMIS, WAMASTAT, WAREL) durch das System ArchiMed an folgenden Klinken am AKH Wien ersetzt:

- Uniklinik für Hals-, Nasen- und Ohrenkrankheiten
- Uniklinik für Innere Medizin I
- Uniklinik für Innere Medizin IV
- Uniklinik für Kinder- und Jugendheilkunde
- Uniklinik für Tiefenpsychologie und Psychotherapie
- Uniklinik für Orthopädie
- Uniklinik für Unfallchirurgie

Archimed in Graz

1998 wurde das System Archimed am LKH-Universitätsklinikum Graz, als dezidiertes Wissenschaftssystem eingeführt. Der Einsatz erfolgte in Kooperation mit IMI[4], dem Grazer Institut für Medizinische Informatik, Statistik und Dokumentation. Die Konfiguration der Software und der Datenbank wurde in enger Zusammenarbeit mit den Wiener Kollegen durchgeführt. Die beiden Systeme an den beiden Standorten Wien und Graz unterscheiden sich nur in wenigen Parametern.

Derzeit wird das System an folgenden Kliniken, des LKH-Universitätsklinikum Graz, verwendet:

- Uniklinik für Chirurgie
- Uniklinik für Orthopädie
- Uniklinik für Kinderchirurgie
- Uniklinik für Neurologie
- Uniklinik für Angiologie

2.1.1. Archimed — ein medizinisches Informations- und Auswertesystem

Im Folgenden wird ein kurzer Einblick in die Teile, die technischen Grundlagen und das Datenmodell von Archimed gegeben.

[3] Medizinische Informations- und Auswertesysteme (MIAS) ist eines von sechs Instituten der MSI an der Medizinischen Universität Wien und umfasst die Arbeitsbereiche Medizinische Informations- und Medizinische Auswertesysteme. http://www.meduniwien.ac.at/msi/mias/

[4] Institut für Medizinische Informatik, Statistik und Dokumentation (IMI) der Medizinischen Universität Graz. http://www.meduni-graz.at/imi/

2.1.1.1. Teile von Archimed

Archimed besteht aus einem Informationssystem und einem Auswertesystem

Das System ArchiMed besteht aus zwei Hauptkomponenten, die auf eine zentrale ORACLE - Datenbank zugreifen (siehe Abbildung 2.1):

- Einem Informationssystem zur Dokumentation medizinischer Daten von klinischen Studien und von Spezialambulanzen, sowie zur Auskunft über Vorbefunde und Verläufe.
- Einem Auswertesystem zur Unterstützung der klinischen Forschung durch statistische Analysen.

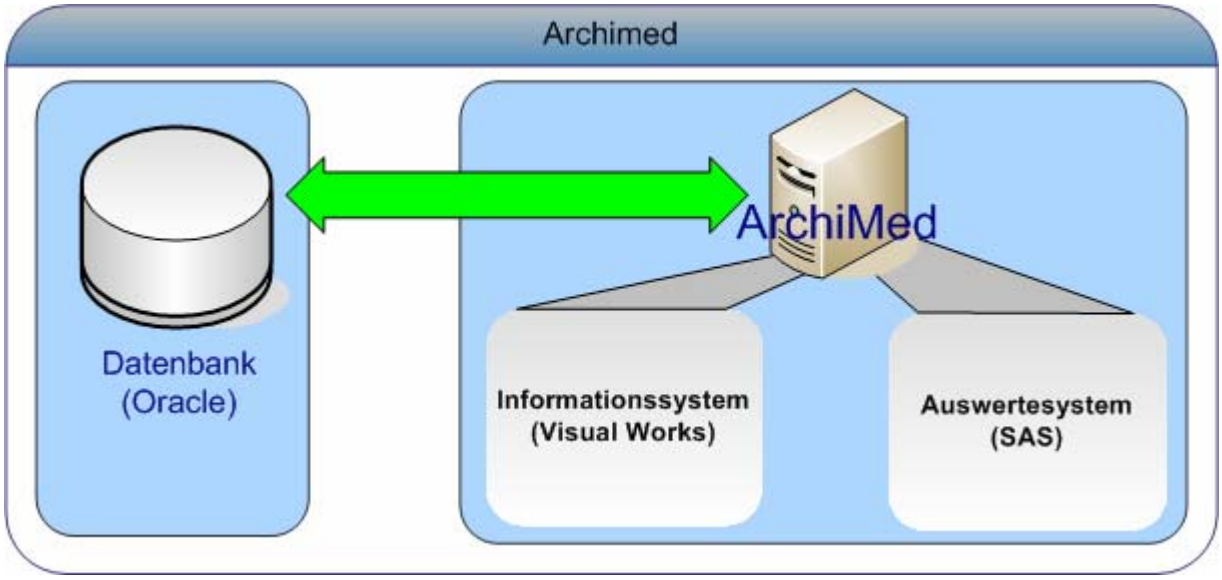


Abbildung 2.1: Archimed besteht aus zwei Hauptkomponenten die auf eine zentrale ORACLE-Datenbank zugreifen.

Diese Arbeit beschäftigt sich mit dem Auswertesystem. Das Informationssystem wird, außer in einer kurzen Funktionsübersicht unter Pkt.2.1.2, nicht weiter betrachtet.

2.1.1.2. Funktionsüberblick

KIS Systeme unterstützen die patientenbezogene Informationsverarbeitung. Es fehlen jedoch Möglichkeiten die Forschung zu erleichtern.

Rechnergestützte Informationssysteme gehören heute zur Grundausstattung der Krankenhäuser und werden im weitesten Sinne unter dem Begriff Krankenhausinformationssystem zusammengefasst. KIS Systeme unterstützen die patientenbezogene Informationsverarbeitung, wie etwa Diagnosen, Laborwerte, usw., die Verwaltung, indem sie ein effektives Management des Krankenhauses unterstützen und die Forschung. Gerade im Bereich Forschung kann KIS meist die Anforderungen der Forscher nicht erfüllen. Es fehlen Möglichkeiten die Forschung zu erleichtern.

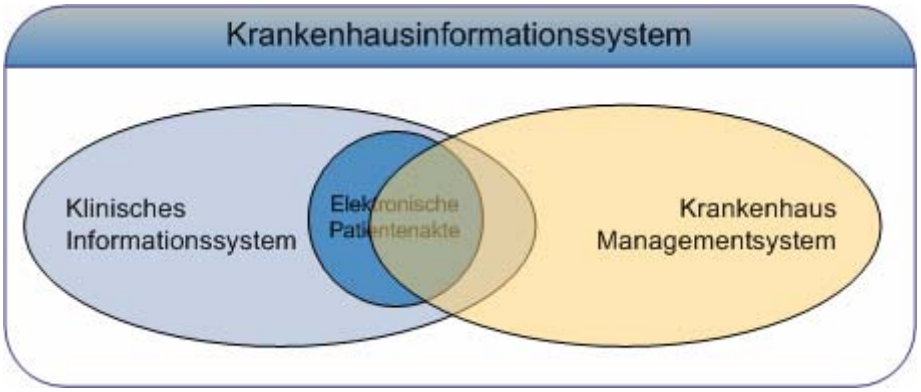


Abbildung 2.2: Das Krankenhausinformationssystem (KIS) besteht aus zwei Subsystemen, dem klinischen Informationssystem, mit der elektronischen Patientenakte als Subsubsystem, und dem Krankenhausmanagementsystem.

Im Wiener AKH hält das KIS Daten von über zwei Millionen Patienten bereit. Das System bietet zwar grundlegende Funktionalität für die Durchführung der Forschung an, aber es werden keine speziellen Funktionen unterstützt um die Forschung zu erleichtern.

Archimed wurde speziell für die Bedürfnisse der klinischen Forschung entwickelt.

Archimed wurde speziell für die Bedürfnisse der klinischen Forschung entwickelt. Archimed verwendet eine eigene Datenbank, in der jedoch die gesamte Information aus dem KIS gespiegelt ist. Zusätzlich können neue Daten, unabhängig vom KIS, eingegeben werden. Es ist möglich bestehende Patientendaten aus dem KIS zusammen mit neuen, für die spezielle Studie eingehobenen Daten zu mischen und gemeinsam zu analysieren [Dorda1999], [Duftschmid2002], [Gall1999].

Die abgedeckte Funktionalität von Archimed kann nach Dorda et al. [Dorda1999], S. 18) in drei Gruppen unterteilt werden:

1. Entwickeln von Formularen und Definieren von Variablen
2. Sammeln und Management von Daten
3. Statistische Analyse

Punkt 1 wird mit Hilfe vom Archimed Informationssystem (siehe Punkt. 2.1.2) durchgeführt. Punkt 2 und Punkt 3 gehören zum Archimed Auswertesystem (siehe Punkt. 2.1.3).

2.1.1.3. Technische Grundlagen

Medizinische Forschung steht vor dem Dilemma, die für ihre Aufgaben notwendigen Informationen aus großen, elektronisch gespeicherten Datenmengen wiederzugewinnen. Archimed ist so gestaltet, dass der Benutzer ohne Kenntnis des

Datenbankschemas und der formalen Anfragesprache Zugang zu den interessierenden Daten bekommen kann.

Archimed wurde mit VisualWorks und SAS entwickelt. Die objekt-orientierte Entwicklungsumgebung VisualWorks basiert aus Smalltalk und erlaubt portable Programmentwicklung. Die Unterstützung der Entwicklung graphischer Oberflächen durch das System war, neben der Portabilität, ein starkes Argument für seinen Einsatz [Dorda1999]. Das Auswertesystem wurde mit SAS entwickelt. Die SAS Entwicklungsumgebung erlabte eine einfache Implementierung der statistischen Funktionen.

Das Auswertesystem wurde mit SAS entwickelt.

2.1.1.3.1. Datenmodell

Archimed wurde explizit als System für die klinische Forschung entwickelt. Keine seiner Funktionen sollten vertieftes Wissen über Datenbanken voraussetzen.

Archimed verwendet eine eigene Datenbank, in der jedoch die gesamte Information aus dem KIS gespiegelt ist. Diese Datenbank basiert auf einem Oracle Datenbankserver und dient als das Data Warehouse für jede klinische Forschung. Die Forderung nach Unterstützung "jeder klinischen Forschung" erfordert ein generisches Datenmodell, das mit Daten unterschiedlichster Herkunft zurechtkommt.

Das Datenmodell von Archimed basiert auf dem Entity-Attribute-Value (EAV) Datenmodell [Nadkarni1997]. Das EAV Modell ist das Korrelat zur Darstellung einer Sparse-Matrix. In einer Sparse-Matrix werden nur nichtnegative Werte abgelegt, analog dazu werden im EAV Modell nur die aktuell zum Patienten passenden Datensätze gespeichert. Das EAV Modell ist nützlich, wenn die Anzahl der Parameter die möglicherweise zu einer Entität gehören kann, deutlich größer ist, als die Anzahl Parameter die zu einer bestimmten Entität gehört - das trifft für Daten in der Medizin zu. Als Entität kann hier der Patient angesehen werden.

Das Datenmodell von Archimed basiert auf dem Entity-Attribute-Value (EAV) Datenmodell

Im EAV Modell werden die Daten einer einzelnen Entität in einer einzelnen Tabelle gespeichert. Die Tabelle weist wenige Zeilen mit drei Spalten auf:

1. Entität (**E**ntity): z. B.: “Patienten ID”.
2. Merkmal (**A**ttribute): z. B:, “Kaliumspiegel”.
3. Wert (**V**alue) des Merkmals: z. B.: “2.9” mEq/dL.

Im EAV Modell werden die Daten einer einzelnen Entität in einer einzelnen Tabelle mit drei Spalten gespeichert.

Der Name EAV Modell leitet sich ebenfalls von diesen drei Spalten im Modell ab.

Abbildung 2.3 zeigt ein Beispiel des EAV Schemas. Die Parameter sind nicht in einer Spalte in der Tabelle fest kodiert, sondern als Metadaten gespeichert (Parameter_info).

Eine einzelne Tabelle (Clinical_Event) speichert die Information über die Entität (Patient und Zeitstempel), Attribute (z. B. Parameter) und Wert jeder Instanz eines Parameters zu einer bestimmten Zeit. Parameter die eine Dauer aufweisen, haben zwei Zeitstempel, wie etwa Beginn- und Endzeit einer Infusion.

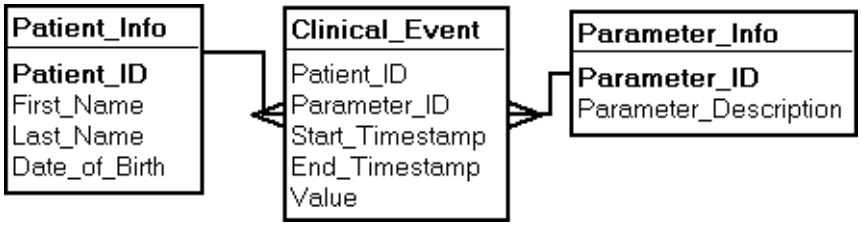


Abbildung 2.3: EAV Schema — In Tabelle Clinical_Event werden die Informationen über die Entität, Attribute und Wert eines Parameters gespeichert.

In EAV Systemen spielen die Metadaten dem Anwender eine klassische Datenorganisation der Daten vor.

Alle DBMS (Database Management System) Systeme nutzen Metadaten, wie etwa Tabelle, Spalte, Zeile, Index für beispielsweise Optimierung von SQL-Anfragen. In EAV Systemen erfüllen Metadaten einen zusätzlichen Zweck — die Illusion einer klassischen Datenorganisation.

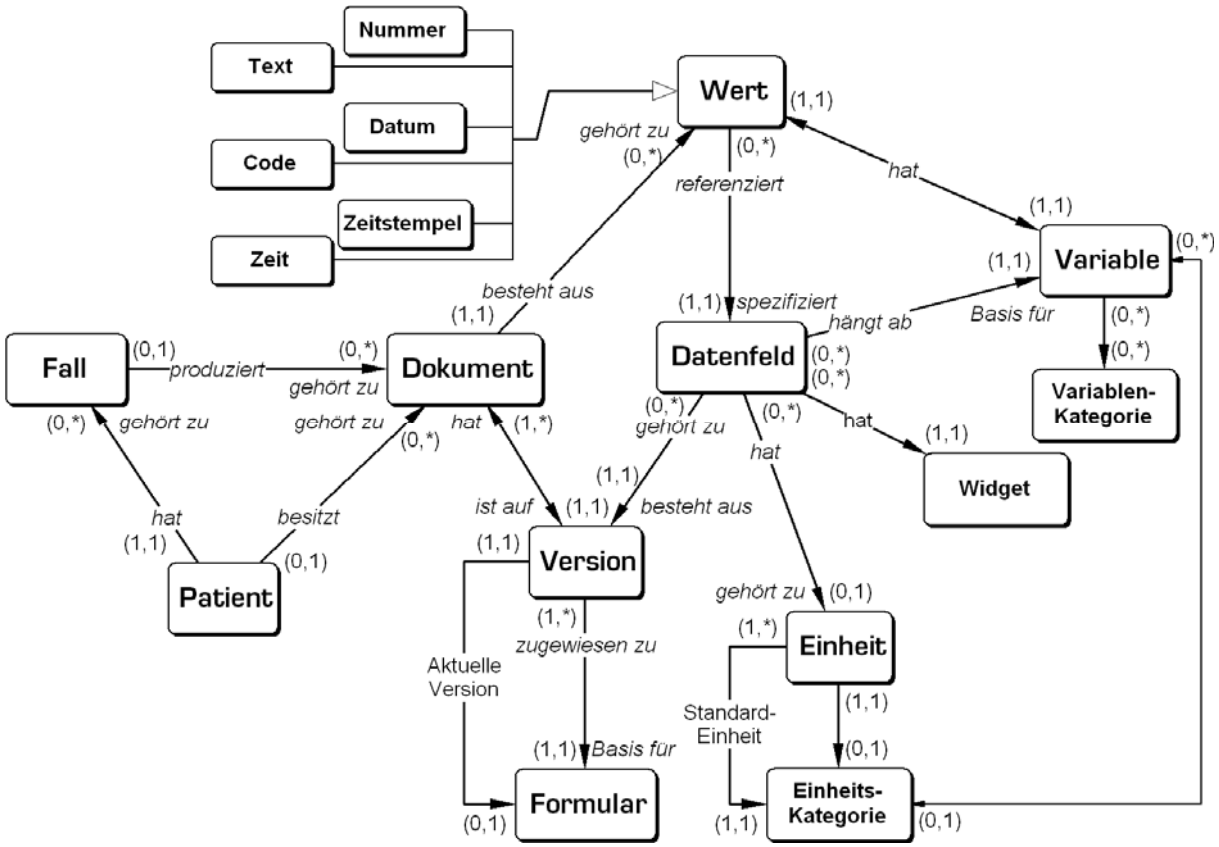


Abbildung 2.4: Zusammenhänge im Archimed Datenmodell — Vereinfachte Darstellung von Zusammenhängen im Datenmodell auf das Archimed aufbaut.

Das auf EAV basierende Metamodell in Archimed erlaubt einerseits eine dynamische Erweiterung von Variablen und Formularen, andererseits macht es die Auswertung von Daten mehrerer unterschiedlicher Formulare möglich.

Das EAV basierende Datenmodell in Archimed baut auf den Elementen Patient, Aufenthalt, Dokument, Formular, Wert und Variable auf und erlaubt eine dynamische Erweiterung von Variablen und Formularen sowie die Auswertung von Daten mehrerer unterschiedlicher Formulare.

Das Datenmodell in Archimed baut auf den Elementen *Patient*, *Aufenthalt*, *Dokument*, *Formular*, *Wert* und *Variable* auf. Abbildung 2.4 verdeutlicht die Beziehungen der Elemente.

Die Variable steht für eine Menge von medizinischen Variablen für die Patientendaten gesammelt werden können. In einem Formular können Daten für die Variablen gesammelt werden. Formulare können in verschiedenen Versionen existieren, wobei immer die aktuelle Version gekennzeichnet ist und das eigentliche Formular darstellt. Vorgängerversionen spiegeln lediglich die Entwicklung des Formulars wieder. Befinden sich Variablen auf dem Formular werden sie zu Datenfeldern. Datenfelder können mit Daten befüllt werden und damit Daten für die Variablen sammeln. Datenfelder auf dem Formular tragen bestimmte Eigenschaften, wie etwa Schriftart oder Position am Formular, und sind mit bestimmten graphischen Symbolen die eine Interaktion zwischen Benutzer und Computer erlauben, sog. Widgets, verknüpft, wie etwa einem Eingabefeld. Numerische Variablen gehören zu bestimmten Kategorien, wie etwa Länge, Gewicht oder Dauer, mit den entsprechenden Einheiten. Datenfelder für numerische Variablen weisen diese Einheiten auf. Ein Dokument ist ein ausgefülltes und abgelegtes Formular. Ein Dokument besteht aus Werten, wie etwa Text, Nummern, Datum, Zeit. Jeder Wert bezieht sich auf ein Datenfeld im Formular und somit letztendlich auf eine Variable. Fälle können für Patienten eröffnet werden. Jeder Fall bezieht sich auf eine Menge von Dokumenten. Dokumente können einem Fall, direkt dem Patienten, oder nicht zugeordnet werden.

2.1.1.3.2. SAS

SAS – der Name leitet sich von der Bezeichnung **S**tatistical **A**nalysis **S**ystem ab, ist jedoch inzwischen nicht mehr als Akronym für eine bestimmte Anwendung zu verstehen. Während die Software ursprünglich ihren Schwerpunkt auf Statistik hatte, wird sie nämlich mittlerweile im gesamten Bereich „Business Intelligence“ eingesetzt, also z.B. für Data Warehousing, Datenanalyse, Data Mining und Reporting. Der Ursprung von SAS ist auf landwirtschaftliche Forschungsarbeiten in den frühen 60er Jahren zurückzuführen. Das Programm wurde vom Software Entwickler *SAS Institute Inc.* auf dem Markt gebracht. *SAS Institute Inc.* gehört seit Jahrzehnten zu den führenden Anbietern von Statistik Software. Das SAS System liegt aktuell in der Version 9.1.3 vor und besteht aus

SAS hatte ursprünglich ihren Schwerpunkt auf Statistik, wird aber mittlerweile im gesamten Bereich „Business Intelligence“ eingesetzt.

SAS besteht aus mehr als 30 Modulen. Unter Windows ist es das Statistikprogramm mit den meisten statistischen Prozeduren. Das System ist zur Verarbeitung großer Datenmangen geeignet. Die Benutzeroberfläche ist jedoch für erfahrene Benutzer ausgelegt und erfordert eine entsprechend lange Einarbeitung.

mehr als 30 Modulen. Das Produkt ist nicht nur für viele Großrechnersysteme verfügbar, sondern auch für Betriebsysteme UNIX, OS/2, DOS und Windows. Zu den Kunden von SAS zählen vor allem größere Institutionen wie Industrieunternehmen, Versicherungen und Universitäten, bei denen heterogene EDV Konfigurationen zu standardisieren sind.

Das SAS System für Windows umfasst unter den heute erhältlichen Statistiksystemen, die größte Zahl statistischer Prozeduren. Darüber hinaus werden anwendungsspezifische Pogrammodule (z. B. für die Medizin) angeboten, die ihrerseits statistische Verfahren enthalten.

Die grafischen Darstellungsmöglichkeiten entsprechen dem Windows Standard und bieten eine Vielzahl von Gestaltungsmöglichkeiten.

Aus den einzelnen SAS Modulen lässt sich ein nach individuellen Bedürfnissen ausgerichtetes, sehr leistungsfähiges Programmsystem, zusammenstellen, das allen Ansprüchen professioneller Anwender genügt. Das SAS System ist darüber hinaus zur Verarbeitung großer Datenmengen geeignet.

Die Benutzeroberfläche des Systems ist für erfahrene Benutzer ausgelegt. Unerfahrene Benutzer sind damit in der Regel überfordert.

Unerfahrene Benutzer sind angehalten nur einfache statistische Funktionen auszuführen um nicht falsche Schlussfolgerungen aus den Ergebnissen zu ziehen. Archimed unterstützt die Ausführung von einfachen statistischen Funktionen durch eine einfache graphische Oberfläche die nur die wichtigsten Funktionen anbietet

Die uneingeschränkte Anwendung von statistischen Prozeduren kann bei unerfahrenen Benutzern zu unerwünschten Ergebnissen mit falschen Schlussfolgerungen führen. Ärzte werden angehalten selbst nur einfache statistische Funktionen auszuführen ([Gall1999], S. 204) und bei Bedarf nach komplexen statistischen Funktionen immer Unterstützung eines Statistikers einzuholen. Archimed unterstützt die Ausführung von einfachen statistischen Funktionen durch eine einfache graphische Oberfläche die nur die wichtigsten Funktionen anbietet (siehe Abbildung 2.18).

Es werden einige Methoden aus der deskriptiven Statistik, wie etwa Mittelwert, Median, Extremwerte, einige statistische Tests, wie etwa der chi-quadrat Test und einige Methoden zur Visualisierung der Daten, wie etwa Histogramme bereitgestellt. Andererseits bietet Archimed dem erfahrenen Benutzer Zugriff auf den gesamten SAS Funktionsumfang. Die Daten können sowohl mit den SAS Modulen, wie etwa SAS/INSIGHT oder SAS/ANALYZE als auch mit selbst geschriebenen Prozeduren weiter verarbeitet werden. Alle Funktionen können direkt in Archimed ausgeführt werden, damit entfällt das Umschalten zu anderen Programmen, verbunden mit dem Verlassen der eigentlichen Auswerteumgebung.

2.1.2. Archimed Informationssystem

Das Informationssystem von Archimed ermöglicht die selbständige Einrichtung von Studien durch den forschenden Arzt.

2.1.2.1. Informationssystem und Studien

Das Informationssystem als Teil von Archimed ermöglicht die Einrichtung von Studien durch den forschenden Arzt.

Das Informationssystem ist ein Teil von Archimed. Es ermöglicht die Einrichtung von Studien durch den forschenden Arzt ohne die Zuhilfenahme von EDV-Personal. Dazu können neue, für Studien zusätzlich benötigten Variablen durch die Eintragung in ein zentrales Variablenverzeichnis durch den Anwender angelegt werden. Weiters kann er neue Dokumentationsformulare mit einem graphischen Formulardesigner leicht erstellen bzw. seine bestehenden Formulare neuen Bedürfnissen anpassen.

Für die Formulare wurde die Form eines Notebooks gewählt, es hat am unteren Ende Reiter. Dadurch ist ein rasches Blättern zwischen den einzelnen Seiten einer Dokumentation möglich.

Daten aus dem KIS-System stehen auf den ArchiMed-Formularen zur Verfügung und können tabellarisch oder als Kurvenverlauf angezeigt werden

Die Stammdaten der Patienten und medizinische Basisdaten werden im AKH im KIS-System erhoben. Durch eine Datenüberleitung zwischen dem System KIS der Gemeinde Wien und dem System ArchiMed stehen diese Daten automatisch auch auf den ArchiMed-Formularen zur Verfügung und können um Daten aus klinischen Studien oder aus Spezialambulanzen ergänzt werden.

Die Auskunftskomponenten von ArchiMed ermöglichen neben einer Ansicht der Daten mittels Formularen auch Verlaufsdarstellungen. Dabei können die gewünschten Daten tabellarisch oder als Kurvenverlauf angezeigt werden.

2.1.3. Archimed Auswertesystem

Das Auswertesystem ermöglicht, im Sinne eines Data Warehouse, gesammelte Daten aus Routinesystemen wie KIS, LIS und Weiteren, den angelegten Studien entsprechend auszuwerten.

2.1.3.1. 3 Hauptkomponenten

Die Hauptkomponenten des Auswertesystems entsprechen den 3 typischen Schritten einer Auswertung.

Die Hauptkomponenten des Auswertesystems entsprechen den typischen Schritten einer Auswertung ([Gall1999], S. 201):

1. Bildung von Patientenkollektiven („Alle Patienten mit Operation A und innerhalb eines Jahres mit Komplikation B“)
2. Auswahl der zu analysierenden Variablen (z. B. Alter, Geschlecht, ...)

3. Analyse der Daten durch statistische Prozeduren (z. B. Deskriptive Maßzahlen, Überlebensfunktion)

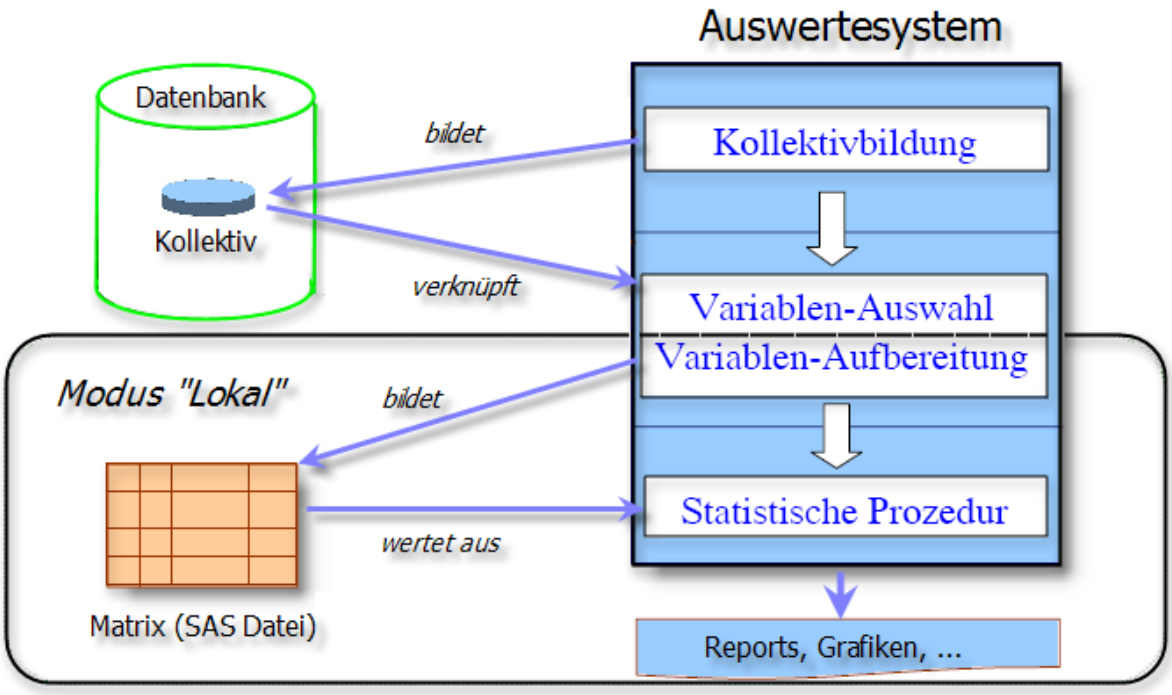


Abbildung 2.5: Hauptkomponenten der Auswertung — Die Hauptkomponenten des Auswertesystems entsprechen den typischen Schritten einer Auswertung (nach [Gall2006], Folie 14)

Neben diesen Hauptkomponenten unterstützt das Auswertesystem weitere Funktionen:

- Thesaurusunterstützung
- Verwaltung von Kollektiven, Abfragen und Ergebnissen
- Listen- und Etikettendruck
- Datenerfassung
- Datenexport
- Einbindung privater Prozeduren (SAS-Datasteps)

2.1.3.2. Attribute für Auswertungen

Attribute für Auswertungen.

Jede Auswertung greift auf eine Menge von Attributen zurück mit denen die Daten eingeschränkt werden.

Mögliche Attribute sind:

- Variablen

  - Vom Typ Text, Numerisch, Datum, Zeitstempel

- Formulardaten

  - Formular, Blatt, Gruppe , Tabelle, Tabellenzeile

- Patientenstammdaten

  - Name, Geschlecht, Alter, Adresse

- Administrative Daten

  - Aufenthalt + Aufenthalts-Datum,

  - Dokument + Dokumentations-Datum

2.1.3.3. Logische Anfrage und Datenselektion

Jede logische Anfrage führt zu einer Datenselektion. Logische Anfragen bestehen aus:

- *Variablen*, das sind die Operanden.

- *Operatoren*

  - Anzahl der Operanden: unär oder binär

  - Art der Operanden: logisch, arithmetisch, statistisch, formularbezogen, zeitbezogen, Vergleiche

- *Klammern*

Die Ergebnisse von Operatoren sind wieder Operanden (Variablen) mit fixer Attributmenge (Abbildung 2.6).

Logische Verknüpfungen stellen dabei eine überlegungstheoretische Fehlerquelle dar.

Binäre Sonderfälle:

ODER: die Zeilen der beiden Operanden werden vereinigt, die Attributmenge ändert sich nicht.

JOIN: die Zeilen der beiden Operanden werden zu einer Zeile verknüpft. Beide Werte kommen in die Ergebniszeile und vergrößern die Attributmenge. Missingwerte können entstehen.

Variablen-Wert
Patient-ID
Geschlecht
Alter
Auftenthalt-ID
Aufenthalt-Datum
Dokument-ID
Dokument-Datum
Blatt
Gruppe
Laufend
Item-ID

Abbildung 2.6: *Ergebnis der Datenselektion — Operanden mit fixer Attributmenge (aus [Gall2006])*

2.2. Anwendungsbeispiel aus der Praxis

Die Anwendung der Attribute aus Abschnitt 2.1.3.2 demonstriert folgendes einfache Beispiel aus der Praxis:

Beispiel:

Wieviele Tonsillektomien an Patienten unter 5 Jahren wurden im Jahre 2005 durchgeführt und wo gab es in den ersten 6 Wochen danach Komplikationen?

2.2.1. Typische Auswerteschritte

Im Abschnitt 2.1.3.1 wurden die drei typischen Schritte bei einer Auswertung beschrieben. Zusammen mit den Attributen aus Abschnitt 2.1.3.2 ergibt sich folgende Verwendung:

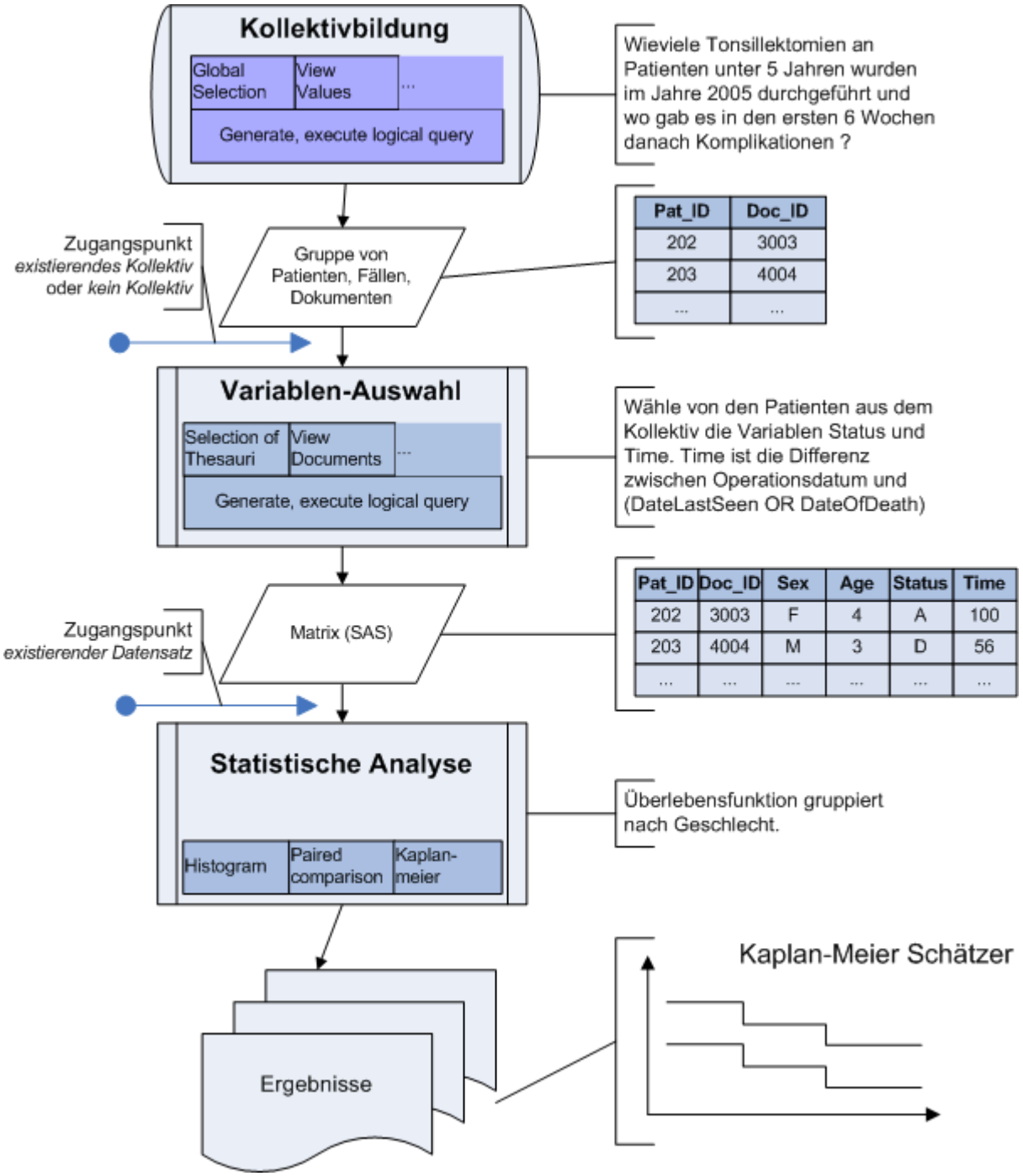


Abbildung 2.7: Die typischen Schritten der Auswertung aus Abschnitt 2.2.1 graphisch verdeutlicht. (Reproduziert nach [Gall1999])

1. Kollektivbildung

   Globale Einschränkung nach „Alter < 5“ und „Dokumentationsjahr = 2005“. Einschränkung auf den Wert „Tonsillektomie“ der Variable „Operationsart“ auf dem Formular „Operationsprotokoll“

2. Variablenauswahl und Variablenaufbereitung

   Auswahl der Variable „Status“ und „Zeit“ zu den Kollektivdokumenten

3. Statistische Analyse

Berechnung der Häufigkeiten . Im Beispiel wird die Überlebensfunktion gruppiert nach Geschlecht angewendet.

2.2.2. Beispiel einer Auswertung

Archimed ist so gestaltet, dass der Benutzer ohne Kenntnis des Datenbankschemas und der formalen Anfragesprache Zugang zu den interessierenden Daten bekommen kann. Die Anfrage erfolgt über einen graphischen Anfragegenerator. Die Kenntnis über die gegenwärtige Benutzeroberfläche ist wichtig für das Verständnis der Intention dieser Arbeit. Das folgende Beispiel demonstriert die Benutzeroberfläche mit ihren Möglichkeiten:

Die Kenntnis über die gegenwärtige Benutzeroberfläche ist wichtig für das Verständnis der Intention dieser Arbeit.

Beispiel:

Es sind alle männlichen Patienten mit mehr als einem erkrankten Gefäß zu einem Kollektiv zusammenzufassen und die Variablen Gewicht und Größe, mittels deskriptiver Statistik, auszuwerten.

2.2.2.1. Kollektiv bilden

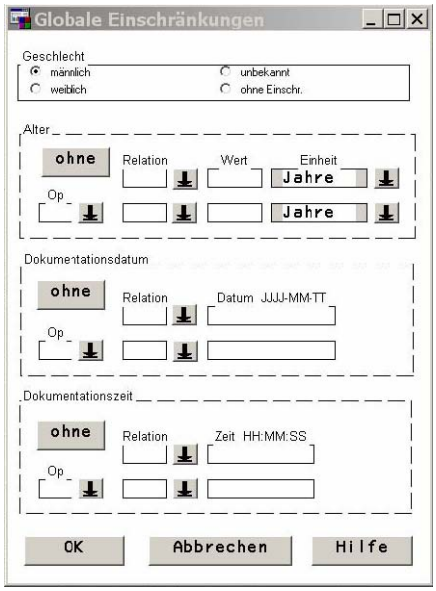
Das Hauptfenster des Archimed Auswerteteils bietet einiges an Funktionalität. Für das obige Beispiel ist nur der nicht ausgegraute Teil von Interesse.

Abbildung 2.8: Hauptmenü der Archimed-Auswertung

Die Pfeile Click 1 bis Click 3 stehen für „Kollektiv bilden“, „Variablen holen“ und „Statistiken“.

Durch Klick auf den Knopf „Kollektiv bilden“(Click 1 in Abbildung 2.8) öffnet sich das Fenster für die Kollektivbildung.

Abbildung 2.9: Das Fenster für Kollektivbildung



Die Patienten sollen mehr als ein erkranktes Gefäß haben. Die Variable „Anzahl erkr. Gefäße“ muss aus den verfügbaren Variablen ausgewählt werden.

Mit Hilfe der globalen Einschränkung (Klick auf GE in Abbildung 2.9) werden nur die männlichen Patienten ausgewählt.

Abbildung 2.10: Globale Einschränkung

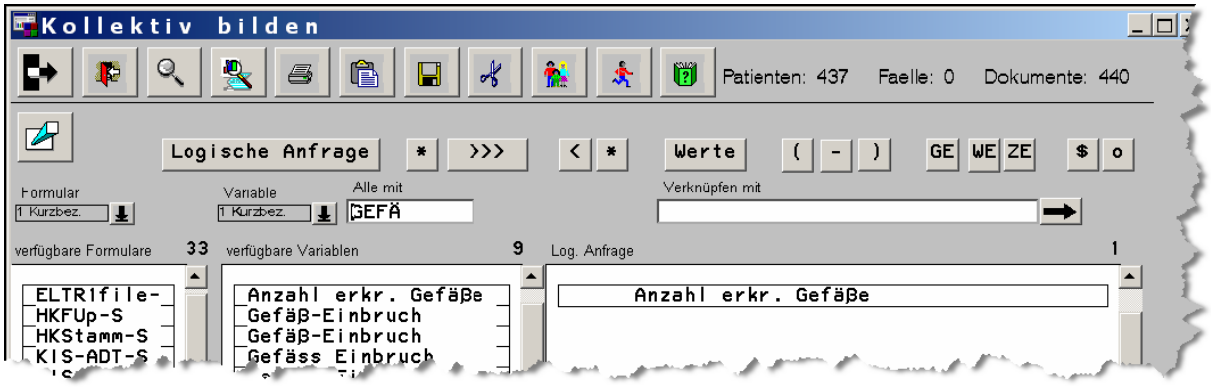


Abbildung 2.11: Variablen auswählen

Um mehr als ein erkranktes Gefäß auszuwählen, wird die entsprechende Variable im Fenster Log Anfrage markiert, der Knopf „Werte“ (siehe Abbildung 2.11) gedrückt.

Im aufgegangenen Fenster Operandenwerte wird durch Klick auf den Knopf WE das Fenster für die Werteeinschränkung aufgerufen (siehe Abbildung 2.12). Für das obige Beispiel muss die Relation > und der Wert 1 gewählt werden.

Die Kollektivbildung kann durch Klick auf den Knopf Starten begonnen werden (siehe Abbildung 2.13)

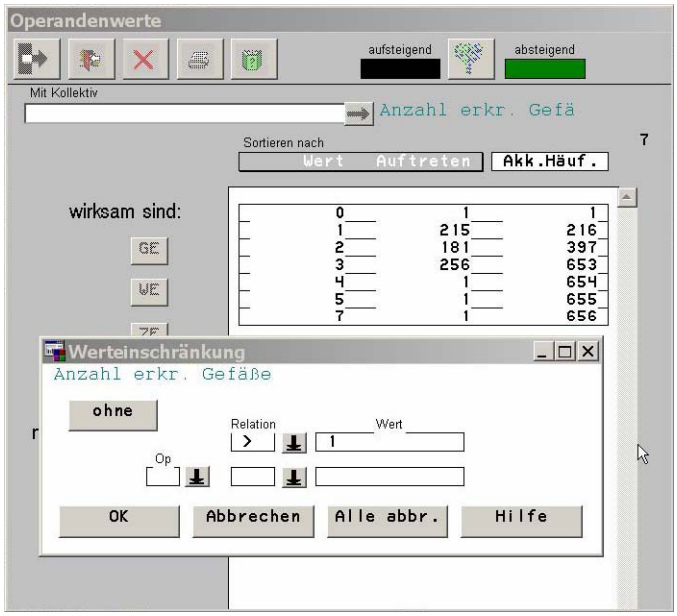


Abbildung 2.12: Werteeinschränkung für die Operandenwerte

Nach vollzogener Kollektivbildung wird rechts oben angezeigt, wie viele Patienten und Dokumente gefunden wurden (Im Beispiel werden 437 Patienten und 440 Dokumente gefunden)

Abbildung 2.13: Kollektivbildung starten.

Optional kann nach erfolgter Kollektivbildung folgendes durchgeführt werden:

- Patienten des Kollektivs ansehen
- Dokumente des Kollektivs ansehen
- Kollektiv oder Kollektivbildung abspeichern

2.2.2.2. Variablenauswahl und -aufbereitung

Für alle Patienten des gebildeten Kollektivs sind die Variablen Gewicht und Größe auszuwerten. Dazu wird aus dem Hauptmenü der Knopf „Variablen holen“ betätigt (siehe Abbildung 2.8, Click 2)

Nach Wahl des entsprechenden Formulars stehen die beiden Variablen zur Verfügung und können gewählt werden. (Abbildung 2.14)

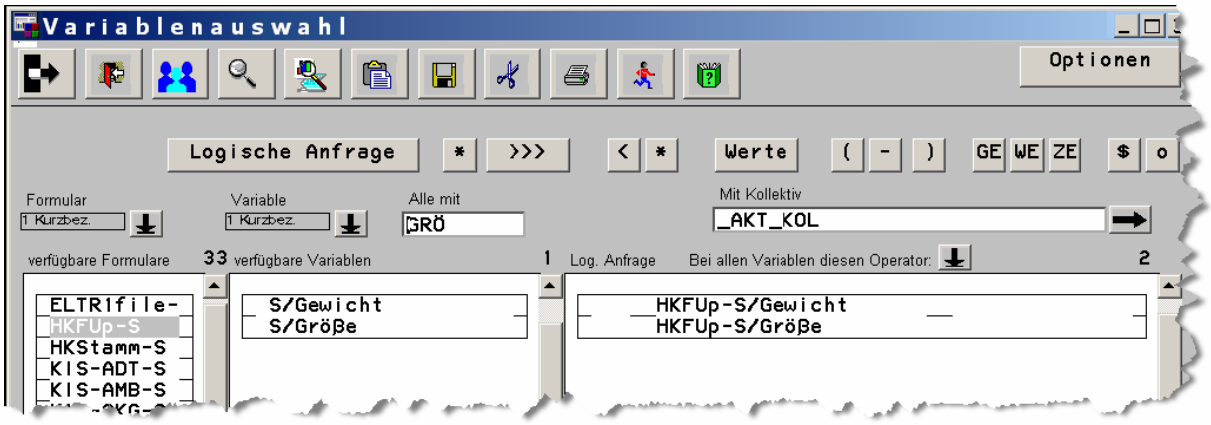


Abbildung 2.14: Variablenauswahl

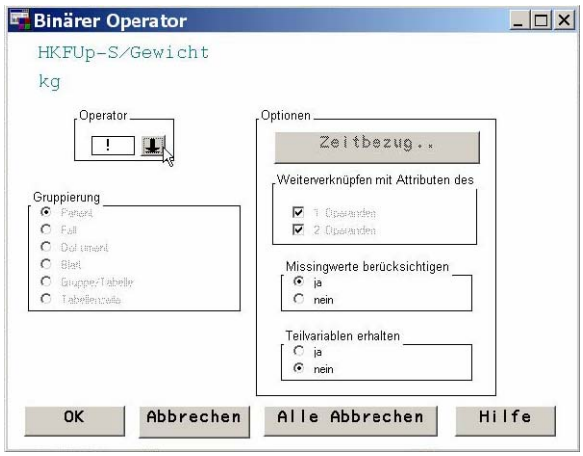


Abbildung 2.15: Binärer Operator

Die gewählte Variablen stehen noch in keiner Beziehung zueinander (Abbildung 2.14). Durch Markieren der ersten Variable und Auswahl des Knopfes *o*, für binäre Operationen wird das Fenster Binärer Operator aufgerufen.

Der Operator ODER wird durch die Eingabe von „!“ im Operatorfeld oder durch die Auswahl aus der Operatorliste formuliert.

Die Variablenselektion wird durch einen Klick auf den Knopf „Starten“ ausgeführt. Es erscheint das Fenster „Kollektiv wie?“. Hier muss die Zeile „patientenorientiert“ auswählt werden. (Abbildung 2.16)

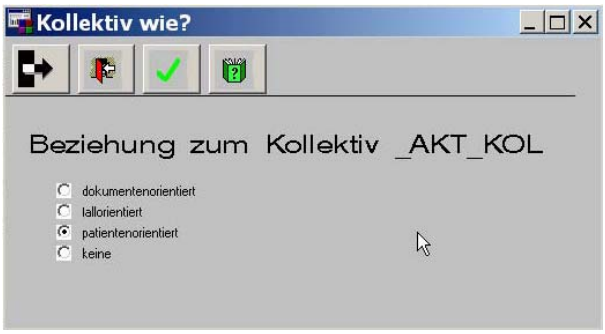


Abbildung 2.16: Fenster Kollektivzugehörigkeit

Mit der Auswahl „patientenorientiert“ werden alle Variablen des Kollektivs berücksichtigt. Die Auswahl „fallorientiert“ selektiert nur alle Werte der Fälle des Kollektivs. Die Auswahl „dokumentenorientiert“ selektiert nur die Werte auf den Dokumenten des Kollektivs.

Die Aufbereiteten Variablen werden für die Weiterverarbeitung mit SAS in das File „my.trans“ abgelegt (Abbildung 2.17)

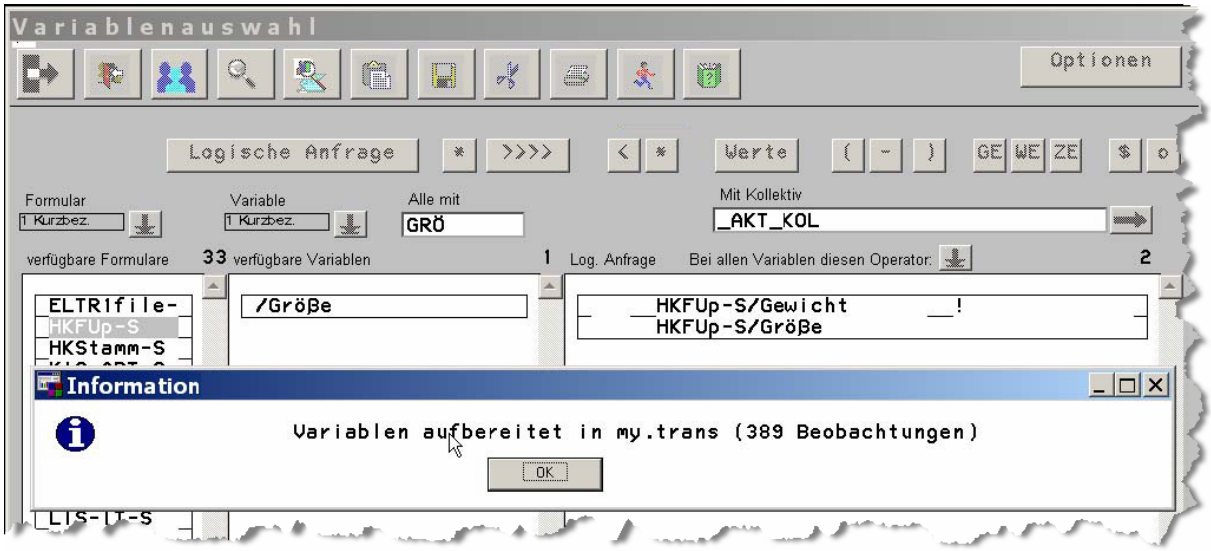


Abbildung 2.17: Aufbereitung der Variablen in „my.trans“

2.2.2.3. Statistische Analyse

Für alle Patienten des gebildeten Kollektivs sind die Variablen Gewicht und Größe statistisch auszuwerten. Dazu wird aus dem Hauptmenü der Knopf „Statistiken“ betätigt (siehe Abbildung 2.8, Click 3). Im Fenster „Statistische Funktionen“ wird unter Univariate Statistik die Methode Deskriptive Statistik gewählt (siehe Abbildung 2.18)

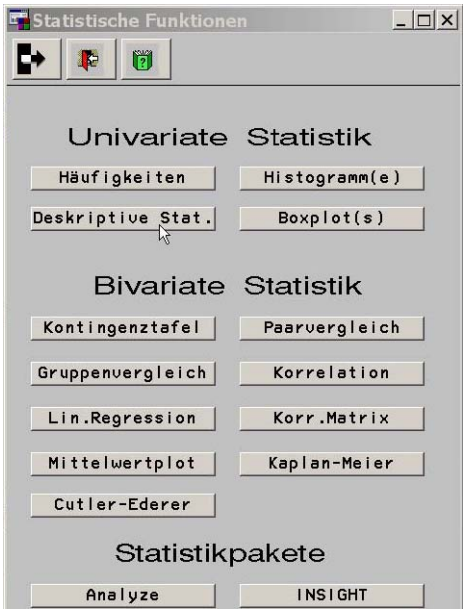


Abbildung 2.18: Fenster Statistische Funktionen

Die Datei „my.trans“ ist schon als SAS Dataset vorgewählt.

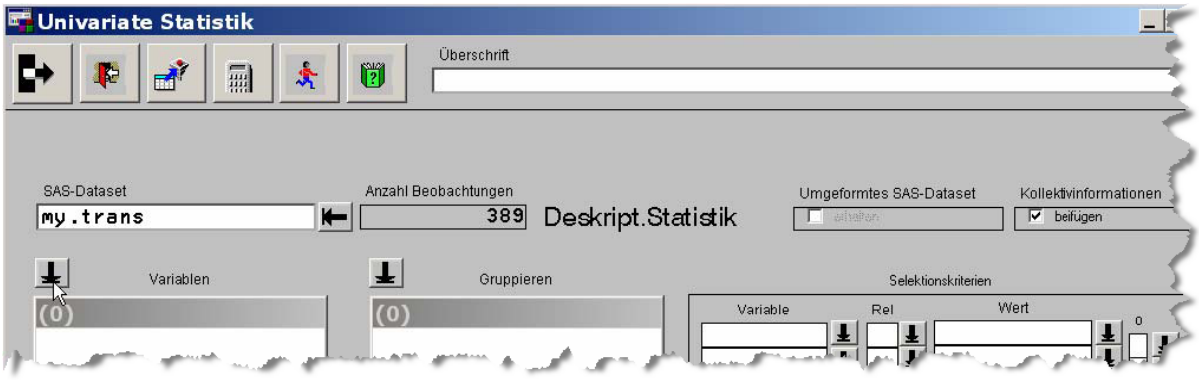


Abbildung 2.19: Fenster Variablenauswahl für Statistik

Mit einem Mausklick auf den Pfeil oberhalb von „Variablen“ können die Variablen Gewicht und Größe ausgewählt werden (siehe Abbildung 2.20)

Abbildung 2.20: Fenster Variablenliste

Die statistische Auswertung wird mit Klick auf Starten begonnen. Die Ergebnisse zeigt Abbildung 2.22 .

Abbildung 2.21: Starten der statistischen Auswertung.

Abbildung 2.22: Ergebnis der statistischen Auswertung.

2.3. Zusammenfassung

Das System Archimed wurde Ende der 90er Jahre als ein medizinisches Informations- und Auswertesystem zur Unterstützung der klinischen Forschung entwickelt. Das System wir gegenwärtig an den Universitätskliniken, AKH Wien und LKH Graz, verwendet.

Archimed wurde speziell für die Bedürfnisse medizinischer Forschung entwickelt. Es wurde die Prämisse verfolgt, ein Tool zu schaffen mit dem auch technisch unerfahrene Benutzer zurecht kommen. Diese Forderung bedingte den technischen Aufbau des Systems. Archimed besteht aus einem Informations- und einem Auswertesystem. Das System verwendet eine eigene Datenbank, in der jedoch die gesamte Information aus dem KIS gespiegelt ist. Zusätzlich können neue Daten, unabhängig vom KIS, eingegeben werden. Es ist möglich bestehende Patientendaten aus dem KIS zusammen mit neuen, für die spezielle Studie eingehobenen Daten zu mischen und gemeinsam zu analysieren. Das Datenmodell basiert auf dem Entity-Attribute-Value (EAV) Modell und ist so konzipiert, dass der Benutzer kein besonders Datenbankwissen zur Benutzung des Systems benötigt.

Die statistischen Funktionen werden durch SAS bereitgestellt. Archimed bietet sowohl dem unerfahrenen Benutzer wie auch dem Statistiker entsprechende Benutzerschnittstellen an. Unerfahrenen Benutzern wird eine begrenzte Menge an einfachen statistischen Auswerte- und Visualisierungsverfahren, über eine graphische Oberfläche angeboten. Erfahrene Benutzer haben Zugriff auf den gesamten SAS Funktionsumfang. Alle Funktionen können direkt in Archimed ausgeführt werden, damit entfällt das Umschalten zu anderen Programmen, verbunden mit dem Verlassen der eigentlichen Auswerteumgebung.

Archimed ist ein mächtiges Informations- und Auswertesystem. Die Benutzerschnittstelle ist funktionell.

Der typische Anwender verfügt zwar zumindest über Grundkenntnisse der Datenanalyse, hat aber nicht notwendigerweise vertiefte Kenntnisse in der Datenbankabfrage. Für komplexe Kollektivbildungen sind unterstützende Techniken vorzusehen, die das Problem der richtigen Bildung logischer Konstrukte, wie beispielsweise Einschließungen und Ausschließungen, mindern. Wichtig sind somit intuitive und interaktive Bedienbarkeit, sowie unterstützende Techniken die eine Strukturierung der Abläufe ermöglichen und die vorgenommenen Auswertungsfolgen für den Anwender transparent und flexibel modifizierbar machen.

KAPITEL

3

Grundlagen

Ich hätte viele Dinge begriffen, hätte man sie mir nicht erklärt...

Stanislaw Jerzy Lec

Im Kern geht es in dieser Arbeit um die bessere Unterstützung der, in der Anfrage von Datenbanken, unerfahrenen Mediziner bei Datenanalysen, d.h. um die komfortable Datenuntersuchung in einer integrierten, interaktiven Analyseumgebung.

Entsprechend bildet die Betrachtung der Anfragesprachen, speziell der visuellen Anfragesprachen, die Modellierung der Metadaten und die Verwendung von Metaphern eine wichtige Grundlage für die weiteren Überlegungen zu der interaktiven Analyseumgebung, die sich dem Anwender letztendlich in Form einer Benutzeroberfläche, als Schnittstelle zur drunter liegenden Datenbanksprache präsentiert.

3.1. Datenanalyse

In der Datenbank abgelegte Daten alleine sind wertlos. Erst die richtige Interpretation der Daten macht aus ihnen Informationen. Nach Fayyad [Fayyad1996] besteht die Aufgabe der Datenanalyse gerade darin, diese Ableitung von Informationen aus Daten durch geeignete Verfahren zu unterstützen.

Daten in der Datenbank werden erst durch richtige Interpretation zur Information.

Erscheint diese Definition der Datenanalyse recht einfach, so ist der Vorgang selbst meist komplex. In einem itera-

Der Vorgang der Datenauswertung ist meist komplex.

tiven Vorgang werden unterschiedliche Methoden zur Ableitung neuer Daten ausgewählt und auf die Zwischenergebnisse der vorangegangenen Schritte angewendet. Eine Illustration ist unter Abbildung 2.7 auf Seite 32 zu sehen.

Statistik ist einer wissenschaftlichen Disziplin und Datenanalyse ihre praktische Anwendung.

Datenanalyse ist zweifellos eng mit Statistik verbunden und beide Bezeichnungen werden gelegentlich synonym verwendet. Ich möchte im Rahmen dieser Arbeit unterscheiden zwischen Statistik, einer wissenschaftlichen Disziplin, die eine Reihe von Analysetechniken bereitstellt und Datenanalyse[5], die deren praktische Anwendung in einem komplexen Datenverarbeitungsprozess darstellt.

Gerade dieser Datenverarbeitungsprozess stellt eine Herausforderung dar, dem, in der Anfrage von Datenbanken unerfahrenen, Anwender ein Werkzeug in die Hand zu geben, mit dem er seine Anfragen intuitiv und interaktiv spezifiziert. Graphische Sprachmittel stellen hierbei ein wichtiges Werkzeug dar. Ihr Vorteil liegt in der Abstraktion vom zugrunde liegenden Datenbankschema.

3.1.1. Anfragesprachen

Der Begriff der Anfragesprache wird zunächst allgemein erläutert. Die interessierenden visuellen Anfragesprachen werden im Abschnitt 3.1.2 näher erläutert. Weiter unten wird auf die Probleme eindimensionaler[6] Anfragesprachen eingegangen.

3.1.1.1. Begriffsbildung

Datenmodelle beschreiben die reale Welt in Datenbanken

Die Modellierung der realen Welt wird in allen Datenbanksystemen durch Datenmodelle realisiert. Das Datenmodell legt die Struktur fest, in der ein Abbild der realen Welt im Computer generiert wird. Es beinhaltet die Möglichkeit:

- das Schema der Datenbank zu beschreiben und zu erzeugen
- die anwendbaren Operationen zur Manipulation der Daten durch den Anwender festzulegen.

Analogie zwischen Datenmodell einer Datenbank und einer Programmiersprache.

Nach Kemper ([Kemper2004], S. 21) besteht eine Analogie zu einer Programmiersprache. Während das Datenmodell die Strukturen und Operatoren festlegt, die zur Modellierung einer Anwendung ausgenützt werden können, legt die Programmiersprache die Typen und Sprachkonstrukte, die zur Realisierung von Anwendungsprogrammen herangezogen werden können, fest.

[5] Im Zusammenhang mit Archimed sprechen wir, in Anlehnung an dessen Auswertesystem, besser von Datenauswertung.

[6] Textuelle Sprachen, wie etwa SQL, stellen eindimensionale Sprachen dar. Visuelle Sprachen stellen mehrdimensionale Sprachen dar.

Das Datenmodell besteht somit aus zwei Teilsprachen:

Das Datenmodell besteht aus der DDL (Data Definition Language) und der DML (Data Manipulation Language), die ihrerseits aus der Anfragesprache und der Datenmanipulationssprache besteht

- der Datendefinitionssprache (DDL[7]), mit der die Datenobjekte beschrieben werden und
- der Datenmanipulationssprache (DML[8]), zur Festlegung der anwendbaren Operatoren

Für Kemper ([Kemper2004], S. 21f) besteht die Datenmanipulationssprache (DML) aus:

- der Anfragesprache (Query Language) und
- der Datenmanipulationssprache, zum Einfügen, Löschen und Modifizieren von Daten.

Außerdem kann die Datenmanipulationssprache (DML) auf zwei Arten genutzt werden:

- interaktiv, durch direkte Eingabe von DML-Befehlen
- eingebettet in eine höhere Programmiersprache, die ihrerseits die DML-Befehle enthält.

In dieser Arbeit wird keine Unterscheidung zwischen Anfragesprache und DML gemacht.

Entgegen Kemper werde ich in dieser Arbeit, den Begriff Anfragesprache als Synonym für die Datenmanipulationssprache verwenden. Ich teile jedoch die Auffassung Kempers und weise darauf hin, dass die in dieser Arbeit verwendete, intuitive Bedeutung des Begriffs Anfragesprache keine vollständige DML darstellt. Im Kontext dieser Arbeit ist diese Differenz jedoch vernachlässigbar.

Visuelle Programmierung lässt sich von der verbalen Programmierung durch die Verwendung visueller Elemente abgrenzen.

Die Kognitionspsychologie unterscheidet zwischen der Verarbeitung visueller und verbaler Informationen durch den Menschen ([Anderson2001], S. 37ff). Analog dazu lässt sich die visuelle von der textuellen[9] Programmierung, durch die Verwendung visueller Elemente abgrenzen [Schiffer1996]. Für Schiffer können visuelle Elemente ausschließlich visuell wahrgenommen werden. Textuelle Objekte sind zwar sichtbar, aber nicht visuell informativ, da deren Information verlustfrei auch von anderen Sinnen erfasst werden kann. Visuell informative Elemente sind etwa nach Wietek ([Wietek2000], S. 32f):

- Graphische Komponenten (Diagramme, usw.),
- geometrische Attribute (Form, Größe, usw.)
- topologische Eigenschaften (Verbindungen, Berührungen, usw.)
- typographische Merkmale (Seitengestaltung, Schriftart, usw.)

[7] Data Definition Language, DDL
[8] Data Manipulation Language, DML
[9] Der Komplementärbegriff zu „visuell“ ist nach [Schiffer1996]der Begriff „verbal“ und nicht der Begriff „textuell“.

In [Myers1986] wird dabei auch von visueller oder mehrdimensionaler Programmierung, im Gegensatz zur linearen oder eindimensionalen Programmierung, gesprochen.

Visuelle und eindimensionale Anfragesprache

In dieser Arbeit wird analog dazu von visueller (mehrdimensionaler) und von eindimensionaler[10] Anfragesprache gesprochen.

Eindimensionale Anfragesprachen überfordern den Anwender durch hohe linguistische, strukturelle und intentionale Forderungen.

Eine Datenbankanfrage stellt nach Kemper ([Kemper2004], S. 69ff) einen logischen Ausdruck über die im Datenschema definierten Objekte und Relationen dar. Demnach müssen Informationen über Objekte, Relationen des Datenschemas und Syntax der logischen Ausdrücke zur Verfügung stehen. Eindimensionale Anfragesprachen stellen diese Informationen beim Anwender als bekannt voraus. Der Anwender muss zudem die Anfragen exakt formulieren, denn vage Vorstellungen über die gesuchte Information reichen nicht aus

3.1.1.2. Probleme eindimensionaler Anfragesprachen

SQL …

Die Sprache SQL (Structured Query Language) ist *die* Datenbanksprache für relationale Datenbanksysteme. Sie dient sowohl zur Datendefinition und Datenmanipulation als auch zur Anfrageformulierung. SQL wurde 1986 vom American National Standards Institute (ANSI) und der internationalen Normierungsorganisation (ISO) standardisiert. Da SQL sehr weit verbreitet ist, wird auf eine ausführliche Vorstellung der Konstrukte dieser Sprache verzichtet und auf einschlägige Literatur verwiesen, wie etwa [Kemper2004]. SQL ist eine eindimensionale Anfragesprache, mit der interaktiv komplexe, logische Anfragen an die Datenbank gestellt werden können. Resultierend aus dem bereits Gesagten stellen solche Anfragesprachen den Anwender vor eine Reihe von Problemen:

… ist eine eindimensionale Anfragesprache die hohe Anforderungen an den Anwender stellt.

- der Anwender muss in der Lage sein, dass was er wissen möchte, in eine korrekte Anfrage zu formulieren.
- die Kenntnis der Syntax und Semantik der Anfragesprache wird vorausgesetzt
- die Kenntnis des relationalen Datenmodells und des Datenbankschemas wird vorausgesetzt
- der Anwender muss wissen über welche Schlüssel- und Fremdschlüsselkombinationen die Beziehungen der Relationen untereinander beschrieben werden, die Namen von Relationen und Attributen, uvm. .

[10] Textuelle Sprachen sind eindimensionale Sprachen. Die einzige Möglichkeit Wörter der Sprache zu verändern besteht darin, die Aneinanderreihung der Buchstaben in eine Richtung zu variieren.

- die mathematischen Konzepte, wie etwa join-Konstrukte, Projektionen oder Aggregierungsfunktionen erfordern ein höheres Maß mathematischem Verständnis.
- das fehlende Feedback bei der Anfrageformulierung verringert die Chance, eine komplexe Anfrage, im ersten Versuch korrekt zu formulieren.
- die fehlende Möglichkeit Metainformationen[11] zu betrachten, führt bei ungeübten Anwendern zur Vernachlässigung von Informationen, die nur implizit in der Datenbank enthalten sind.

3.1.2. Visuelle Sprachen

Visuelle Sprachen sind weder selbsterklärend, noch kann man ihren Inhalt auf den ersten Blick erfassen. Trotzdem sind sie den textbasierten Sprachen in mancher Hinsicht überlegen.

Ich werde den Begriff visuelle Sprache und visuelle Anfragesprache synonym verwenden. Streng genommen sind visuelle Anfragesprachen jedoch Spezialisierungen von visuellen Sprachen.

3.1.2.1. Begriffsbildung

Eine einheitliche Terminologie für visuelle Sprachen ist schwer zu finden. Zunächst muss der Begriff „visuell“ definiert werden. Nach Schiffer ([Schiffer1996], S. 270) sind unter „visuell“ die Komponenten oder Attribute zu verstehen, die ausschließlich visuell wahrgenommen werden können. Schiffer spricht in diesem Zusammenhang von „visuell informativ“ und meint damit, dass die Information über andere Sinne nicht gewonnen werden kann. Er führt beispielhaft an, dass farbige Objekte etwa, falls die Farbe eine Bedeutung hat, visuell informativ sind, da die Interpretation der Farbe nur über das visuelle Wahrnehmungssystem erfolgen kann.

Visuelle Komponenten oder Attribute können ausschließlich visuell wahrgenommen werden.

Myers [Myers1986] definiert eine visuelle Sprache als ein System, das Graphik einsetzt. Diese Definition ist sehr weit gefasst. So lassen sich hierunter im weitesten Sinne alle Aspekte graphischer Benutzerschnittstellen unterbringen. Schiffer in ([Schiffer1996], S.270) definiert eine visuelle Sprache als eine formale Sprache mit visueller Syntax oder Semantik und dynamischer oder statischer Zeichengebung. Mit visueller Syntax meint Schiffer eine graphische Notation der Grundsymbole und mit visueller Semantik den Laufzeit-

Visuelle Sprache ist eine formale Sprache mit visueller Syntax oder Semantik und dynamischer oder statischer Zeichengebung.

[11] Metainformationen sind Informationen die nicht explizit in der Datenbank gespeichert sind, sondern die aus gespeicherten Daten abgeleitet werden. (Aus gespeichertem Geburtsdatum etwa kann das Alte berechnet werden.) Im Unterschied zu Metadaten, die Daten zur Beschreibung von Daten darstellen.

zustand der visuellen Objekte. Bei dynamischer Zeichengebung werden Informationen nicht statisch, sondern durch einen zeitlichen Ablauf repräsentiert, wie etwa das Aufblinken eines Objektes, das sich verändert.

Wietek führt in ([Wietek2000], S. 33) an, dass visuelle Programmiersprachen in vielen Abwandlungen über die Verwendung, wie er schreibt, „bedeutungsvoller" visueller Elemente definiert werden. Dabei bleibt für Wietek die Abgrenzung zu

- der Verwendung graphischer Softwareentwicklungsumgebungen
- der Programmvisualisierung[12]
- der Programmierung graphischer Benutzeroberflächen oder
- der Nutzung informeller Notationen

so wie die genaue Definition des Begriffs „bedeutungsvoll" in der Literatur unklar formuliert.

Visuelle Umgebungen und visuelle Sprachen.

Shu ([Shu1999], S. 200ff) unterscheidet zwischen (siehe Abbildung 3.1):

- Visuellen Umgebungen
  - zum visuellen Unterrichten
  - zur Visualisierung von Daten, Programmen und Softwareentwurf
- Visuellen Sprachen
  - zur Bearbeitung vom visuellen Daten
  - zur Unterstützung visueller Interaktion
  - zur Programmierung mit visuellen Ausdrücken

Wobei genau genommen nur die letzte Klasse der visuellen Sprachen die visuelle Programmierung erlaubt. In den anderen Klassen bezieht sich der Begriff „visuell" nur auf die Form der dargestellten Information.

Visuelle Sprachen sind eine Obermenge der textuellen Sprachen.

Schmidt ([Schmidt2006], S. 10) hat den Begriff der visuellen Sprache noch verfeinert. Er unterscheidet zwischen, „echt visuellen" Sprachen, deren „wesentliche Teile" eine graphische Notation haben und textuellen Sprachen, in denen graphische Elemente vorhanden sein können, aber nicht bedeutungsvoll sind. Schmidt definiert visuelle Sprachen als eine Obermenge der textuellen Sprachen.

[12] Hier ist die reine Darstellung von Programmstruktur und Programmablauf gemeint, nicht aber die Konstruktion von Programmen.

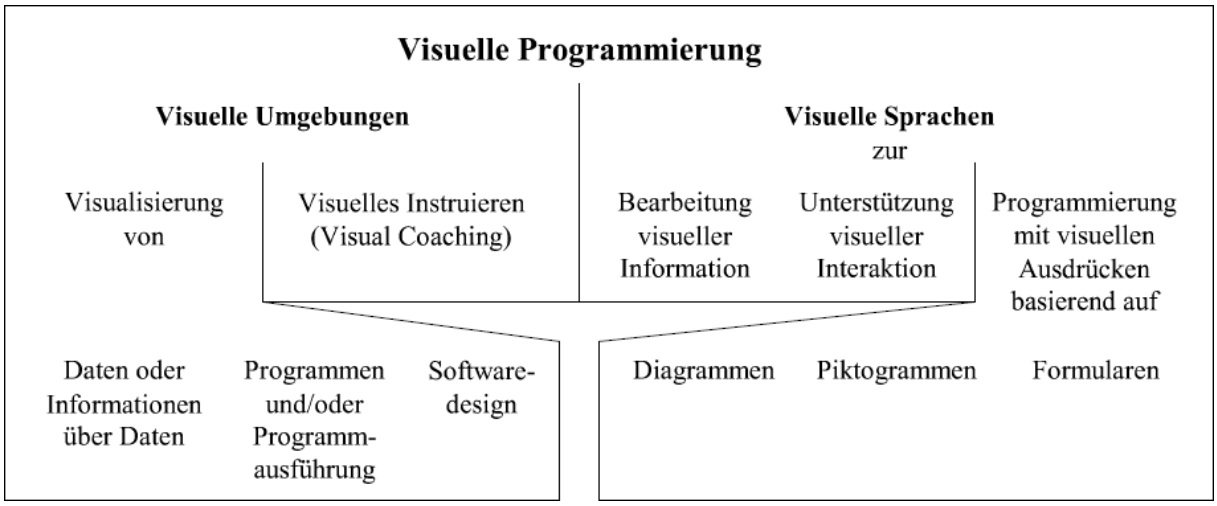


Abbildung 3.1: Klassifikation visueller Programmierung (aus [Wietek2000], S. 34 nach [Shu1999])

Visuelle Sprachen gewinnen an Bedeutung, weil sie grobe Strukturen und Zusammenhänge besser veranschaulichen können als textuelle Sprachen.

Um visueller Sprachen zu implementieren, ist Expertenwissen aus verschiedenen konzeptionellen und technischen Bereichen erforderlich. Zu nennen sind hier visuelle Entwurfsaspekte, Techniken zur Realisierung der grafischen Darstellung, benutzerfreundliche Interaktions- und Layoutmechanismen sowie allgemeine Methoden zur Analyse und Transformation visueller Programme.

Die Implementierung visueller Sprachen erfordert Expertenwissen aus verschiedenen konzeptionellen und technischen Bereichen.

3.1.2.2. Vor- und Nachteile visueller Sprachen

Die Kognitionspsychologie hat ergeben, dass verbale Informationen sequentiell, visuelle Informationen jedoch parallel verarbeitet werden. Das führt dazu, dass über eine visuelle Darstellung mehr Informationen gleichzeitig wahrgenommen werden können und somit vom Menschen besonders wirkungsvoll verarbeitet werden ([Anderson2001], S. 75ff). Andersons Überlegungen beruhen auf zwei grundlegenden Tatsachen:

Über die visuelle Darstellung kann mehr Informationen gleichzeitig wahrgenommen werden.

- Die Informationsverarbeitung des menschlichen Gehirns wird durch analytische Aufgabe anders angesprochen als durch visuelle Eindrücke. Die rechte Gehirnhälfte unterstützt parallele, die linke sequentielle Verarbeitung der Information. Nutzung visueller Information kombiniert die Fähigkeiten beider Gehirnhälften und führt zu höherer Verarbeitungskapazität (siehe Abbildung 3.2)
- Das menschliche Gehirn löst die Welt um sich herum in Grundeinheiten die miteinander in Beziehung stehen auf. Es bildet ein semantisches Netz an bekannten

Konzepten. Neue Eindrücke werden durch Einordnung in diese semantischen Netze, auf bereits bekanntes Wissen abgebildet. Die Information wird umso besser behalten, je besser diese Abbildung gelingt. Verwendung von visueller Information, in Form von intuitiven, natürlichen Metaphern, kann diese Einordnung der Information unterstützen.

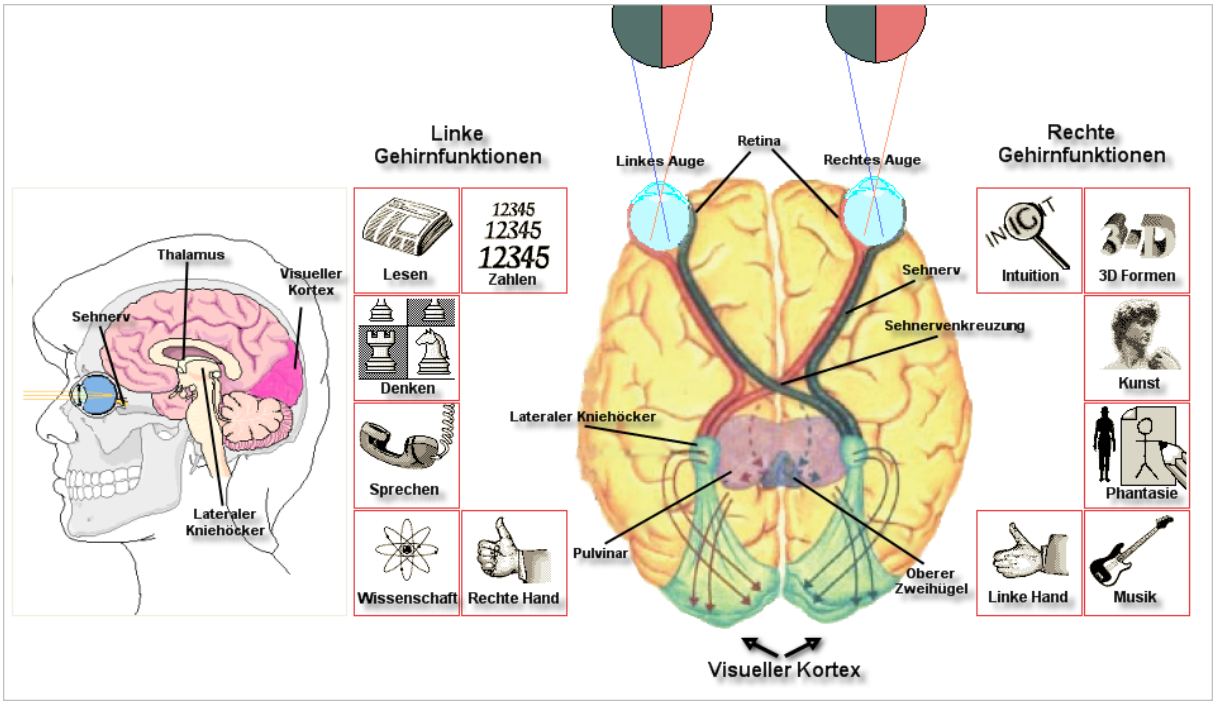


Abbildung 3.2: Gehirn — Darstellung des visuellen Kortex

Metaphern erlauben einen besseren Zugang zu einem künstlichen System.

Natürliche Metaphern spielen, seit der Einführung von piktogrammorientierten[13] Systemen[14] zur Steuerung des Computers, in der Bedienung des Computers eine wichtige Rolle und haben zu ihrer Vereinfachung geführt.

Visuelle Metaphern erlauben einen besseren Zugang zu einem künstlichen System. Es ist jedoch häufig schwierig angemessene Metaphern zu finden, die die Semantik der darzustellenden Objekte richtig auszudrücken. Die Benützung dieser Bilder kann dann zu Missverständnissen führen. Eine Klassifikation von Metaphern findet sich in [Lohse1994].

Direkte Manipulation

Shneiderman [Shneiderman1983] hat den Begriff „Direkte Manipulation“ geprägt, der bei der Verwendung graphischer Objekte große Bedeutung erlangt hat. Shneiderman führt einige Prinzipien dazu an:

[13] Ein Piktogramm ist ein einzelnes Bildsymbol, das eine Information durch vereinfachte grafische Darstellung vermittelt.
[14] Damit sind Benutzerschnittstellen bzw. Desktops gemeint.

- Permanente graphische Darstellung
- Steuerung durch einfache, intuitive physische Aktionen, statt einer komplexe Kommandozeilensyntax
- schnelle, umkehrbare Operationen, deren Wirkung unmittelbar sichtbar wird

Prinzipien der Direkten Manipulation

Direkte Manipulation erlaubt es dem Benutzer, sich auf seine eigentliche Aufgabe zu konzentrieren. Der Computer wird „transparent". Beim Versuch den Begriff „direkte Manipulation" zu erklären hat Don Hatfield[15] bei IBM den Begriff Wysiwyg (what you see is what you get) geprägt.

Direkte Manipulation macht den Computer „transparent".

Der oft zitierte Satz „Ein Bild sagt mehr als 1000 Worte" scheint bei der visuellen Programmierung nicht immer zu zutreffen. Ein Bild kann unterschiedlichen Betrachtern, aufgrund fehlender Exaktheit auch unterschiedliches sagen [Schiffer1996, S. 272]. Es gibt in visuellen Sprachen, im Gegensatz zu verbalen Sprachen, keine Regeln wie Diagramme zu lesen sind. Letzten Endes können komplexe Abläufe auch zu komplexen graphischen Darstellungen führen, die mindestens genauso schwer verständlich sind, wie die verbale (textuelle) Darstellung.

Ein Bild kann unterschiedlichen Betrachtern, aufgrund fehlender Exaktheit unterschiedliches sagen.

Schiffer [Schiffer1996, S. 272ff] nennt fünf Thesen für den Einsatz visueller Sprachen gegenüber rein Textuellen:

Fünf Thesen für den Einsatz visueller Sprachen gegenüber rein textuellen Sprachen

1. Visuelle Programme erleichtern die Kommunikation: über ausdruckstarke Bilder lassen sich Informationen besser vermitteln, jedoch stellt die Normierung der Darstellung ein Problem dar. Abstrakte Inhalte etwa sind nur schwer visualisierbar.
2. Visuelle Programme sind leicht verständlich: unerfahrenen Anwendern erleichtern Bilder die Bedienung eines Systems und das Erfassen der Wirkungsweise von Programmen. Komplexe Diagramme werden jedoch schnell unübersichtlich und sind eher verständnishemmend als verständnisfördernd.
3. Visuelle Programme sind leicht merkbar: Anderson [Anderson2001, S. 82ff] führt Experimente an, bei denen die gute visuelle Erinnerungsleistung des menschlichen Gehirns belegt wird. Die bildliche Information wird jedoch, im Gehirn, durch eine abstrakte Repräsentation der Bedeutung des Bildes ersetzt. Dadurch muss die Bedeutung des Bildes, das gemerkt werden soll, erfasst werden. Umfangreiche und detailreiche Codeteile sind dadurch schwer zu merken. Bei einem Verlust der Detailinformation, bleibt nur die Grob-

[15] Don Hatfield war von 1965 bis 1992 bei IBM beschäftigt. Derzeit ist er Technologie Consultant und Mitbegründer einer Firma die Wysiwyg Editoren für MathML und andere XMLs baut.

struktur des Programms im Gedächtnis und kann zu fehlerhafter Erinnerung führen.

4. Visuelle Programme sind leicht erlernbar: visuelle Programmierumgebungen bieten für den Anfänger einen spielerischen Einstieg in die Sprache. Erste Erfolge setzen schnell ein und sind förderlich für die Motivation. Die Gefahr der Frustration ist jedoch hoch, wenn die visuelle Sprache nicht mächtig genug zur Bewältigung komplexer Problemstellungen ist und die anfänglich flache Lernkurve steil ansteigt.
5. Visuelle Programme überwinden Sprachbarrieren: Bilder sind international verständlich, jedoch können nicht alle Aspekte einer Sprache sinnvoll in Bilder codiert werden. Sie erreichen dadurch nicht die gleiche Aussagekraft wie verbale (textuelle) Sprachen.

Visuellen Sprachen arbeiten auf einem extrem hohen Abstraktionsniveau.

Ein wesentlicher Vorteil von visuellen Sprachen ist, dass sie auf einem extrem hohen Abstraktionsniveau arbeiten. Syntaktische Elemente einer textuellen Sprache werden durch graphische Objekte mit Attributen repräsentiert. Geeignete visuelle Sprachen erlauben es dem Benutzer nur syntaktisch korrekte Programme zu schreiben. Sie erlauben nur so genanntes strukturelles Editieren. Objekte können nur dort eingefügt werden wo sie auch syntaktisch erlaubt sind. Gute visuelle Sprachen unterstützen den Entwickler bei der Arbeit durch geeignete Fehlermeldungen und Konsistenzüberprüfungen bzw. automatische Hilfefunktionen. Das hohe Abstraktionsniveau erlaubt eine Entkopplung von der Zielsprache, die sogar austauschbar sein kann.

Strukturelles Editieren – geeignete visuelle Sprachen erlauben es dem Benutzer nur syntaktisch korrekte Programme zu schreiben.

Leicht modifizierbar

Ein weiterer Vorteil von visuellen Sprachen ist, dass Objekte leicht nachträglich modifiziert werden können. In einem UML[16] Klassendiagramm kann etwa sehr einfach eine neue Klasse eingefügt werden, indem andere Objekte entsprechend verschoben werden. In einer rein textuellen Darstellung ist dies nicht ohne weiteres möglich.

Der Lernaufwand für das erlernen des Systems bleibt niedrig.

Visuelle Sprachen sind besonders von Vorteil wenn Metaphern einer bestimmten Domäne benutzt werden oder wenn graphische Objekte strengen allgemein gültigen Konventionen unterliegen. Der notwendige Lernaufwand für das erlernen des Systems bleibt niedrig. Dadurch ist gerade in speziellen Domänen ein Einsatz sinnvoll, wenn der Anwenderkreis nicht aus Programmierern besteht. Das trifft für das Umfeld dieser Arbeit zu.

[16] UML: Die Unified Modeling Language ist eine von der Object Management Group (OMG, http://www.omg.org/) entwickelte und standardisierte Sprache für die Modellierung von Software und anderen Systemen.

Der Platzbedarf und der Aufwand bei der Konstruktion eines Programms, sind als Nachteil einer visuellen Sprache anzusehen. So benötigt eine visuelle Sprache wesentlich mehr Platz auf dem Bildschirm als eine Textuelle. Dadurch kann weniger Information gleichzeitig dargestellt werden als bei einer textuellen Sprache (sog. Screen Space Problem[17]). Die Konstruktion eines Programms mittels einer visuellen Sprache ist, bedingt durch die großen Mauswege, manchmal aufwändig.

Der Platzbedarf auf dem Bildschirm ist wesentlich größer als bei textuellen Sprachen.

3.1.2.3. Begriffsdefinition und Anforderungen

Die Definition der visuellen Anfragesprache in dieser Arbeit ist an die Definition visueller Programmiersprachen nach Schiffer in [Schiffer1996] angelehnt und erweitert:

> *Visuelle Anfragesprachen sind Sprachen mit visuell informativen, zwei- oder mehrdimensionalen Elementen, die syntaktisch oder semantisch signifikant sind. Sie visualisieren Datenbankinhalte und Datenbankstrukturen und unterstützen die Datenselektion.*

Definition der visuellen Anfragesprache

Es können etwa Schemadiagramme auf den Bildschirm gezeichnet werden, in denen man die Elemente mit der Maus anklicken kann. Dadurch können Details oder auch die einzelnen Datensätze ausgegeben werden.

Unter visueller Anfragesprache wird nach der obigen Definition ein System verstanden, dass einem Benutzer Datenbankanfragen mit mehrdimensionalen[18] Hilfsmitteln erlaubt.

Um eine entsprechende graphische Repräsentation zu erhalten, muss die Anfragesprache in der Lage sein jede Anfrage, die durch das Datenbanksystem unterstützt wird zu visualisieren.

Die im Abschnitt 3.1.2.2 erläuterten Prinzipien der „Direkten Manipulation“ nach [Shneiderman1983] finden auch bei einer Datenbankumgebung Anwendung:

Prinzipien der Direkten Manipulation bei Datenbankanwendungen

- visuelle Repräsentation der Anfragekomponenten,
- visuelle Repräsentation der Ergebnisse,
- schnelle und reversible Steuerung der Anfragen,
- Selektion durch Anklicken, nicht durch Eintippen,
- unmittelbares und ständiges Feedback.

[17] Screen Space Problem: "The problem with visual languages is that you cannot have more than 50 visual primitives on the screen at one time" (in [Davies1997], S. 15)

[18] Beispielsweise graphische Repräsentation im Gegensatz zur konventionellen eindimensionalen textuellen Darstellung.

Diese Prinzipien erzeugen sowohl bei unerfahrenen Datenbankanwender wie auch Experten Vorteile. Unerfahrenen Anwendern helfen visuelle Anfragesprachen bei der Formulierung der Anfragen, da sie mit der Terminologie und dem Schema der Datenbank nicht vertraut sein müssen[19]. Datenbankexperten profitieren von visuellen Anfragesprachen, da sie komplexere Anfragen überblicken können. Durch die visuelle Darstellung können Informationen übersichtlicher dargestellt werden, so dass schwer verständliche Ergebnisse leichter ausgewertet werden können.

Hohe Anforderungen an Hard- und Software

Visuelle Anfragesprachen stellen, durch die graphische Komponente, hohe Anforderungen an die Hard- und Software. Beim heutigen technischen Fortschritt stellt das aber kein Problem dar.

3.1.3. Abbildungsschichten

Hierarchisches Ebenenmodell

Zur allgemeinen Beschreibung des Systementwurfs eines Datenbanksystems und Gewährleistung der Datenunabhängigkeit[20] wird nach ([Vossen2000], S. 22ff) ein hierarchisches Ebenenmodell herangezogen (Abbildung 3.3).

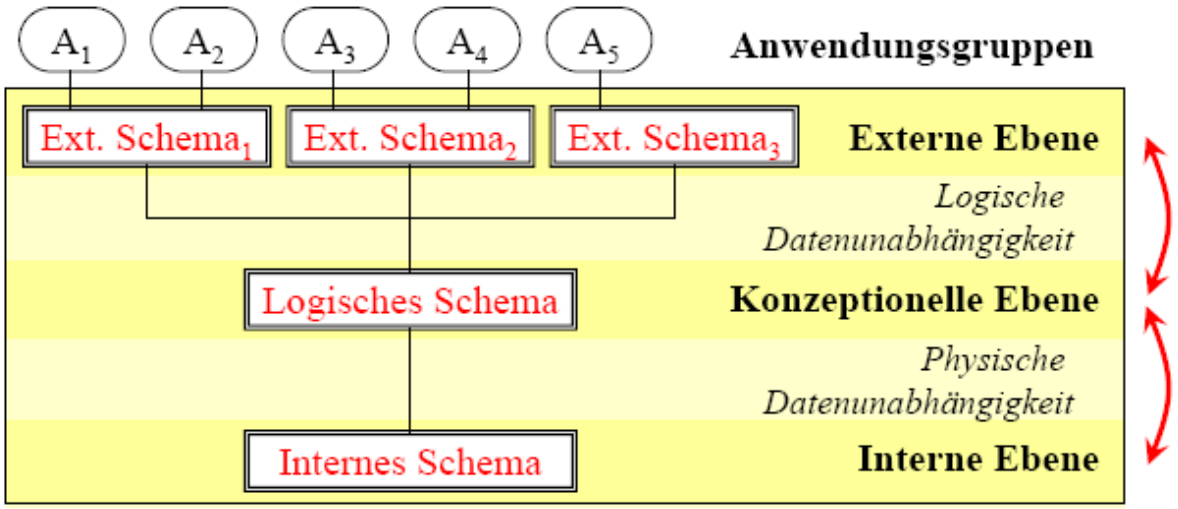


Abbildung 3.3: Das 3-Ebenenmodell nach ANSI/SPARC. Es beschreibt die Trennung von Benutzer, dem DBMS und der Datenspeicherung.

[19] Ich lasse hier außer Acht, dass ein entsprechendes Wissen über die Datenbank bei der Formulierung der Anfragen nützlich ist.

[20] Die Datenunabhängigkeit kann nach [Kemper2004] in zwei Aspekte aufgeteilt werden:
Die *Implementierungsunabhängigkeit* bedeutet, dass die konzeptuelle Sicht auf den Datenbestand unabhängig von der physischen Realisierung der Speicherung der Daten ist, d.h. dass das konzeptuelle Schema unabhängig von dem internen Schema ist.
Die *Anwendungsunabhängigkeit* bedeutet, dass sich die Datenstrukturen der Anwendungssysteme unabhängig von der Datenbank entwickeln können, d.h. dass das konzeptuelle Schema und das externe Schema unabhängig voneinander sind.

- Die *interne Ebene* definiert die interne Organisation des Datenbanksystems zur Speicherung des Datenbestandes. In dieser Ebene werden die Implementierungsdetails, wie etwa. Dateiorganisation und Zugriffspfade, der gespeicherten Daten beschrieben. Diese Ebene ist abhängig vom verwendeten Basissystem. Sie beschreibt also, wie und wo die Daten gespeichert werden.

Interne Ebene

- Die *konzeptionelle Ebene* definiert eine konzeptionelle Sicht auf die gesamte Datenbank. In dieser Ebene werden die Daten aus einer allgemeinen, globalen, implementierungs- und anwendungssystemunabhängigen Sichtweise beschrieben. Diese Ebene wird spezifiziert in einem systemunabhängigen Datenmodell, zum Beispiel dem ER-Modell[21]. Sie beschreibt also das Datenmodell (hier das relationale Modell). Mit diesem Modell wird beschrieben, wie auf die Daten zugriffen wird und in welchem Zusammenhang diese zueinander stehen

Konzeptionelle Ebene

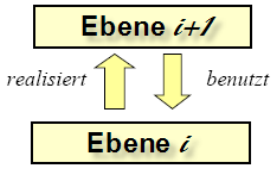
- Die *externe Ebene* definiert die partiellen Sichten der Benutzer und Anwendungssysteme auf die globale, konzeptionelle Ebene. Diese Ebene beschreibt alle externen Datenstrukturen und ihre Ableitung aus der konzeptionellen Ebene. Sie beschreibt also alle Möglichkeiten, die der Benutzer hat, um auf die Datenbank zuzugreifen (Programme, Funktionen und Schnittstellen)

Externe Ebene

Die hierarchisch angeordneten Ebenen stellen unterschiedlich mächtige Abstraktionsniveaus dar. Jede Ebene bietet an ihrer oberen Schnittstelle eine Reihe von Objekten und Operationen an, die unter Verwendung der Objekte und Operatoren der darunter liegenden Ebene realisiert werden. Keine Ebene ruft Operationen aus einer höheren Ebene auf. Darüber liegende Ebenen setzen auf dem jeweiligen Abstraktionsgrad auf. Darunter liegende Ebenen stellen den jeweiligen Abstraktionsgrad zur Verfügung.

Die hierarchisch angeordneten Ebenen bauen aufeinander auf.

In Abbildung 3.4 werden schematisch die Abhängigkeiten zwischen den Ebenen i und $i+1$ über die Operationen Op ihrer Schnittstelle dargestellt:



- Ebene i realisiert die Operationen Op der Schnittstelle
- Ebene $i+1$ benutzt die Operationen Op der Schnittstelle

Abbildung 3.4: Abhängigkeiten der Ebenen eines Datenbanksystems

[21] ER-Modell: das Entity-Relationship-Model (Gegenstands-Beziehungs-Modell), dient im Rahmen der Datenmodellierung einen Ausschnitt der realen Welt zu beschreiben.

Abstraktionsniveaus reduzieren die Komplexität des Systems.

Die unterschiedlich mächtigen Abstraktionsniveaus des Ebenenmodells reduzieren die Komplexität des Systems und gewährleisten die Erweiterbarkeit, Konfigurierbarkeit und Portabilität.

Eigenschaften der Ebenenbildung:

- Abstraktion: Konzentration auf das Wesentliche, Verdrängen des Unwesentlichen.
- Lokalität: Physische Zusammenfassung von Zusammengehörigem (Daten und Algorithmen).
- Verdecken: Beschränkung der Sichtbarkeit von Details auf diejenigen Teile eines Systems, die sie auch benötigen.

5-Schichtenmodell

Um diese Eigenschaften auch für eine visuelle Anfragesprache sicherzustellen, wird diese in das 5-Schichtenmodell (siehe [Vossen2000], S. 28f) eingeordnet

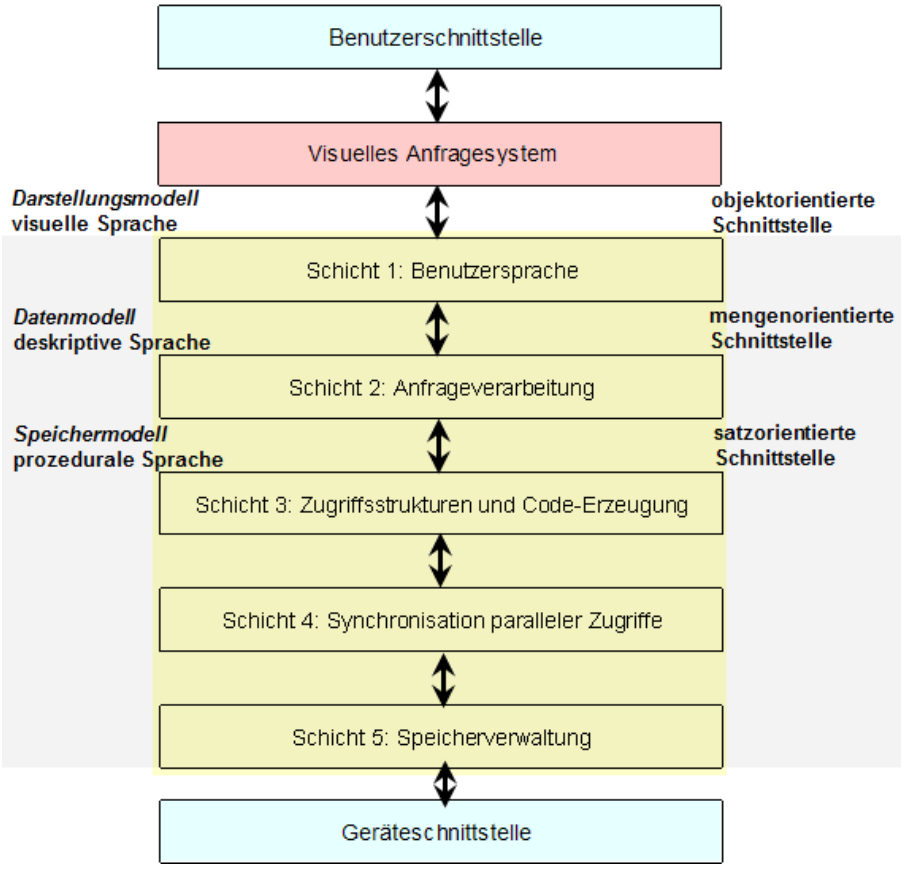


Abbildung 3.5: Das 5-Schichtenmodell, erweitert um die visuelle Anfragesprache.

Im Gegensatz zum 3-Ebenenmodell, dass die Transformationsschritte und die dabei behandelten Datenstrukturen noch etwas ungenau beschreibt, werden im 5-Schichten-

modell die Transformationskomponenten eines Datenbank-Systems detaillierter erklärt. Abbildung 3.5 veranschaulicht das um die Schicht des visuellen Anfragesystems erweiterte 5-Schichtenmodell.

5-Schichtenmodell

Jeder Schicht sind bestimmte Aufgaben zugeordnet:

Schicht 1

- Schicht 1 nimmt die Anfrage des Benutzers in der Anfragesprache des Datenbank-Systems entgegen und überprüft die syntaktische Korrektheit. Im Relationenmodell, aber auch in den meisten anderen Datenmodellen, handelt es sich um eine mengenorientierte Anfrage, d.h. der Gegenstand der Anfrage sind Mengen von Tupeln.

Schicht 2

- Schicht 2 übersetzt die deklarative Anfrage in imperative Funktionsaufrufe.

Schicht 3

- Schicht 3 bietet die Implementierungen der imperativen Funktionen bezüglich der intern vorhandenen Zugriffsstrukturen. Hier entsteht also der „ausführbare" Code auf der internen Repräsentation der Daten. Mit Hilfe dieses Codes kann die Anfrage berechnet werden.

Schicht 4

- Schicht 4 bildet die Zugriffe auf einzelne Datensätze auf lesende und schreibende Zugriffe auf Seiten ab und synchronisiert zeitgleich vorliegende Zugriffe.

Schicht 5

- Schicht 5 schließlich lokalisiert die angeforderten Seiten und transferiert diese zwischen Haupt- und Sekundärspeicher.

3 weitere Schichten

Diese Einteilung eines DBS in fünf Schichten muss noch nach oben und unten um insgesamt drei Schnittstellen ergänzt werden:

Benutzerschnittstelle

- Der Benutzerschnittstelle, die dem Benutzer die Eingabe von Anfragen und Änderungsoperationen ermöglicht und die Ergebnisse präsentiert.

Geräteschnittstelle

- Der Geräteschnittstelle, die der Schicht 5 den Zugriff auf das die Datenbank speichernde Gerät ermöglicht.

Visuelles Anfragesystem

- Das visuelle Anfragesystem, das die deskriptive Sprache aus Schicht 1 weiter abstrahiert und dem Endanwender als eine visuelle Sprache zur Verfügung stellt.

Modell für eine abstrakte Datenschnittstelle.

Im Ansatz von Banhart und Klaeren in [Banhart1995] wird herausgearbeitet wie die Schicht zur Realisierung einer visuellen Sprache aufgebaut ist. Um den Anwendungen eine abstrakte Datenschnittstelle zur Verfügung zu stellen, so dass diese vollkommen befreit sind von dem Schema der darunter liegenden Datenbank und ihrer Zugriffslogik, wird in [Banhart1995] die Idee verfolgt, auf die Datenbank ein abstraktes Datenmodell für Datenzugriffe aufzusetzen. Das entwickelte Datenmodell verbindet Merkmale objektorientierter Systeme

mit dem Entity-Relationship Ansatz, um die semantischen Informationen für die Modellierung von forschungsorientierten Patientendaten explizit repräsentieren zu können. Das Datenmodell wurde in zwei Ebenen als eine Sicht auf eine relationale Datenbank implementiert, in der die relationalen Datenstrukturen als eine Sammlung von miteinander in Beziehung stehenden Objekten interpretiert werden. Im Gegensatz zum Viewkonzept relationaler Datenbanken ist eine Sicht hier nicht nur auf eine Verbindung beschränkt, sondern umfasst eine Vielzahl miteinander verbundener Objekte. Das Konzept stellt eine objektorientierte Interpretation der in einer relationalen Datenbank gespeicherten Daten. Eine vollständige Beschreibung der zugrunde liegenden Datenstruktur kann durch die Konzepte der tieferen Ebene, die so genannten „Primitive" des Datenmodells, erreicht werden. Auf einer höheren Ebene wurden zusätzliche Konzepte für die Modellierung benutzerspezifischer Sichten implementiert (Abbildung 3.6).

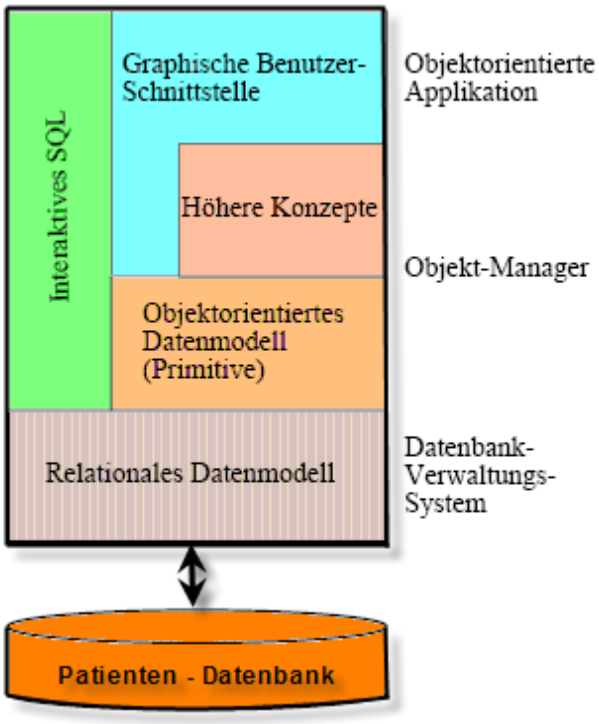


Abbildung 3.6: Ebenen des Datenmodells nach [Banhart1995]. (Reproduziert nach [Beesten1995], S. 21)

Anfragegenerator

Ein Anfragegenerator visualisiert eine graphische Abstraktion dieses Datenmodells mit Hilfe gespeicherter Metadaten. Die erstellten Datenbankanfragen werden einem separaten Prozess, dem SQL-Generator, übergeben, der einerseits den SQL-Befehl zusammensetzt und an den Datenbank-Server überstellt und andererseits das Ergebnis an das Anfragemodul zurückliefert (Abbildung 3.7).

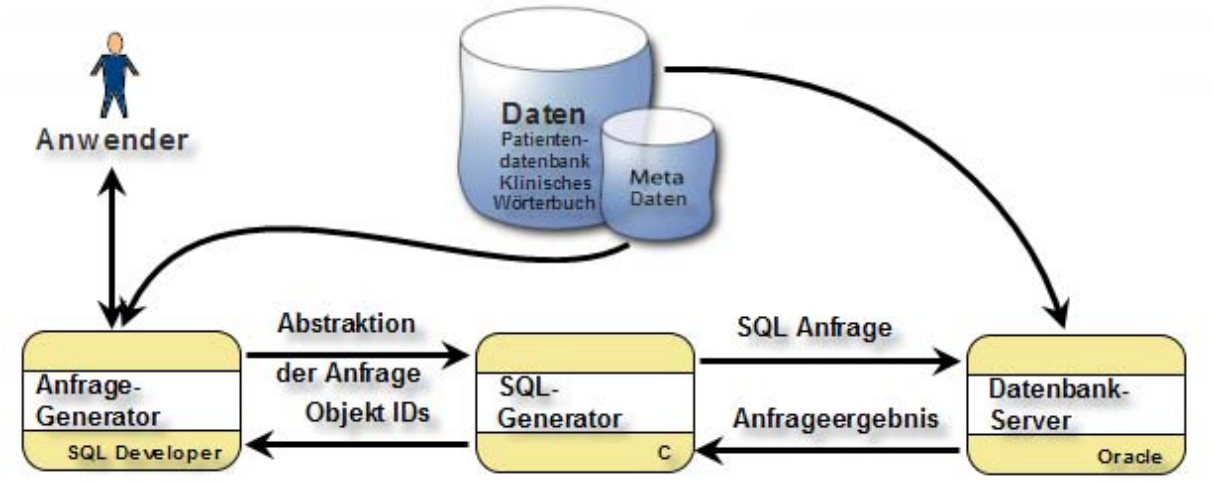


Abbildung 3.7: Architektur des Anfragegenerators. (Reproduziert nach[Beesten1995], S. 22)

3.2. Metadaten

Metadaten finden sich in allen Bereichen der Daten- und Informationsverarbeitung. Parallel zur wachsenden Größe der Datenbestände in rechnerbasierten Informationssystemen, wächst auch der Bedarf an zusätzlichen Metadaten zur näheren Beschreibung[22] dieser Daten. Während der Begriff "Metadaten" vergleichsweise neu ist, ist sein Prinzip jahrhunderte lange bibliothekarische Praxis.

3.2.1. Begriffsbildung

Metadaten

Metadaten können definiert werden als Daten, die eine oder mehrere Ressourcen beschreiben oder als Daten, die mit einem Objekt verbunden sind und dieses Objekt beschreiben.

„Daten über Daten“

Metadaten stellen daher eine Beschreibung von Dokumenten, Objekten oder Diensten dar und enthalten Informationen zu deren Inhalt, Struktur oder Form. Metadaten sind salopp formuliert Beschreibungen von Daten bzw. "Daten über Daten".

Metadaten vs.Mikrodaten und Makrodaten

In statistischen Datenbanken werden diejenigen Daten als Metadaten bezeichnet, die nicht direkt den Inhalt einer Statistik darstellen. Zu den statistischen Metadaten zählen auch Beschreibungen der Datenfelder in Umfrageformularen, oder sogar auch komplette Formularbeschreibungen. Die eigentlichen statistischen Daten bezeichnet man, in Abgrenzung zu den Metadaten, als Mikrodaten und Makrodaten (siehe auch Abschnitt 1.2 auf Seite 13).

[22] Metadaten werden für die Definition, Dokumentation, Selektion, Manipulation und Darstellung von Daten benötigt.

Im Kontext statistischer Datenbanken führt Wietek in ([Wietek2000], S. 63) eine Definition für Metadaten von H.J. Lenz (1982) an:

Metadaten

> *Metadaten beschreiben Mikro- und Makrodaten sowie auf diesen durchführbare Operationen auf semantischer, struktureller, statistischer und physikalischer Ebene, um eine sinnvolle Speicherung, Transformation, Abfrage und Verbreitung zu ermöglichen. (H.J. Lenz, 1982 in [Wietek2000])*

Unterscheidung zwischen Daten und Metadaten ist nicht immer eindeutig

Wietek führt an, dass der Kontext und die Art der Betrachtung der Daten eine große Rolle spielt und eine Unterscheidung zwischen Daten und Metadaten nicht immer eindeutig möglich ist ([Wietek2000], S. 63). Letztlich ist es eine Frage des Standpunktes ob Informationen als Daten oder Metadaten aufgefasst werden. Für den Leser eines Buches etwa stellt der Inhalt die eigentlichen Daten dar, während der Name des Autors oder Nummer der Auflage Metadaten sind. Für den Herausgeber eines Bücherkatalogs sind diese beiden Eigenschaften aber unmittelbar interessant und stellen für ihn die eigentlichen Daten dar.

Bei der Verwendung von Metadaten findet meist keine Trennung zwischen Objektebene und Metaebene statt. So spricht man etwa davon in einem Katalog ein Buch zu suchen und nicht nur seine Metadaten.

Der Vorteil der Metadaten besteht darin, dass diese getrennt von den eigentlichen Nutzdaten in Speichern mit kurzer Zugriffszeit gespeichert und dadurch schnell wieder aufgerufen werden können. Die Nutzdaten werden hingegen auf Speichermedien mit längerer Zugriffszeit gespeichert.

Standardisierung der Metadaten

Metadaten werden eingesetzt, um Beziehungen zwischen, beispielsweise, Informationen herzustellen. Das setzt in der Regel erst eine gewisse Standardisierung der Metadaten voraus. Zu nennen ist hier die Meta Data Facility (MOF)[23] Spezifikation der Object Management Group[24] (OMG), die eine abstrakte Sprache und ein Framework zur Verwaltung von plattformunabhängigen Metamodellen beschreibt. Beispiele solcher Metamodelle sind die Unified Modeling Language (UML)[25], das Common Warehouse Metamodel (CWM)[26] und die MOF selbst.

[23] MOF: http://www.omg.org/mof/ (Stand: 13.01.2007)
[24] OMG: http://www.omg.org/ (Stand: 13.01.2007)
[25] UML: http://www.uml.org/ (Stand: 13.01.2007)
[26] CWM: http://www.omg.org/technology/cwm/ (Stand: 13.01.2007)

3.2.2. Klassifikation von Metadaten

Art der Metadaten nach Anwendungszweck

Der Anwendungszweck bestimmt die Art der Metadaten. Wietek ([Wietek2000], S. 63ff) differenziert die Metadaten nach der Art ihrer Verwendung (siehe Abbildung 3.8):

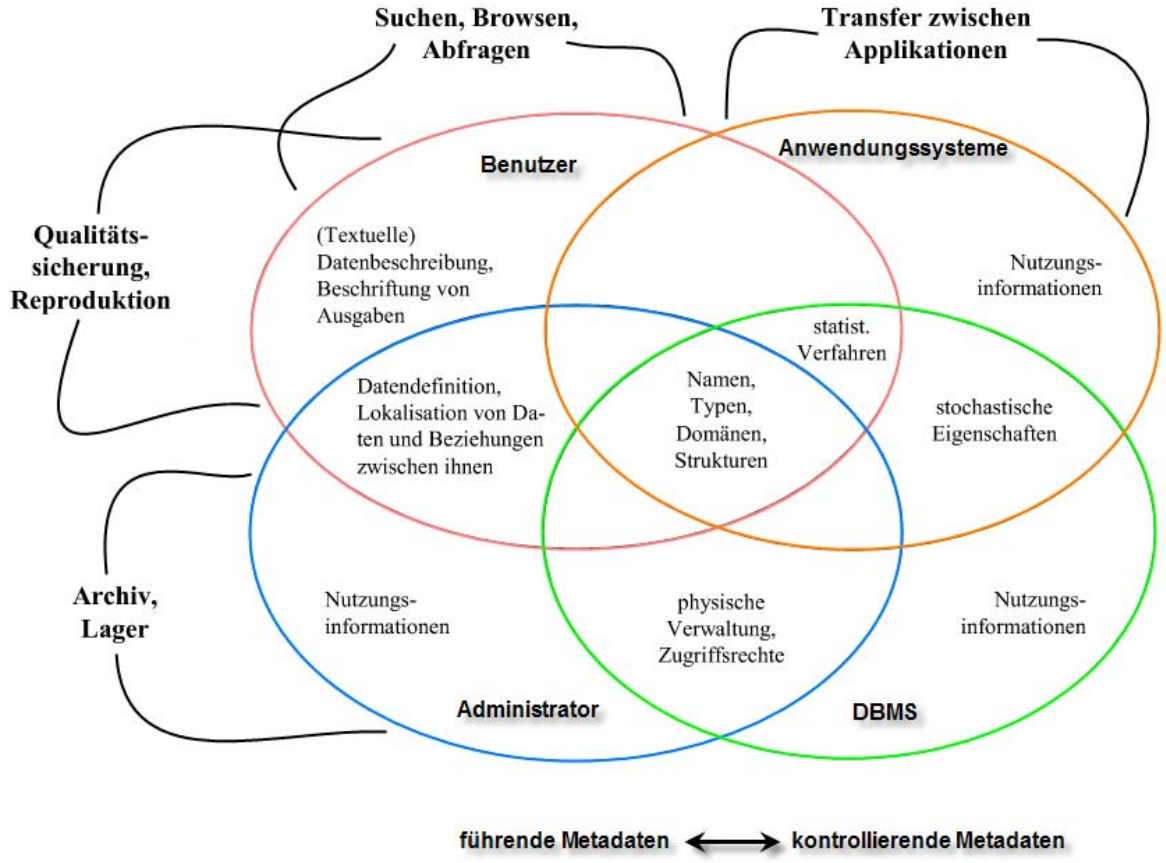


Abbildung 3.8: Metadaten nach der Art der Anwendung (Reproduziert nach [Wietek2000], S. 64)

Nutzer von Metadaten

- Nutzer von Metadaten: Metadaten werden von Systembenutzern, Administratoren, Anwendungssystemen und dem DBMS verwendet.

Aufgaben von Metadaten

- Aufgaben von Metadaten: Metadaten werden beim Suchen, Browsen, Abfragen, bei der Reproduktion von Daten und beim Transfer der Daten zwischen Applikationen verwendet.

Führende und kontrollierende Metadaten

- Führende und kontrollierende Metadaten: führende Metadaten stellen eine Informationsquelle für den Benutzer dar, während kontrollierende Metadaten die interne Funktionalität gewährleisten.

Metadaten auf externer, konzeptioneller und interner Schemaebene

In Anlehnung an die Unterscheidung von externer, konzeptioneller und interner Schemata beim Datenbank-Schichtenmodell (vgl. [Vossen2000], S. 28f) unterscheidet Wietek ([Wietek2000], S. 63ff) Metadaten auf

externer und konzeptioneller Schemaebene:

- Elementare und zusammengesetzte Metadaten: Metadaten können einfache Bezeichner usw. sein oder komplexe Strukturen bilden.
- Datentypen: Metadaten können textuelle, numerische, aber auch viele andere, auch zusammengesetzte Formen aufweisen.
- Granularität: Metadaten beschreiben Daten beginnend bei einfachen Merkmalen bis hin zu ganzen Datenbasen.
- Interne und externe Metadaten: interne Metadaten sind auf das jeweilige Informationssystem beschränkt; auf externe Metadaten greifen externe Systeme zu

und auf interner Ebene:

- Implizite und explizite Metadaten: explizite Metadaten können eindeutig von den jeweiligen Daten unterschieden werden, wohingegen implizite Metadaten Teil der Daten sind.
- Getrennte Verwaltung der Metadaten: Daten und Metadaten werden getrennt im jeweiligen Informationssystem oder in unabhängigen Systemen verwaltet.

3.3. Metapher

Der Begriff Metapher spielt eine wichtige Rolle, wenn es um Benutzerfreundlichkeit, Informationsarchitektur und User Interface Design geht. Metapher sind aber auch kompliziert und empfindlich. Ihre Bedeutung kann leicht Überschätzt werden und wenn sie überstrapaziert werden, verlieren sie ihren Nutzen.

3.3.1. Begriffsbildung

Metaphern beruhen auf Analogien

Der Ausdruck „Metapher“ geht etymologisch zurück auf Griechisch „metaphorá“ (griechisch μεταφορά, „Übertragung“), das zusammengesetzt ist aus „metá“ (über) und „phérein“ (tragen) (vgl. Duden Herkunftswörterbuch). Metaphern beruhen auf Analogien. Sie nutzen die Vertrautheit eines Begriffes (Quellbereich) um einen anderen, unvertrauten Begriff (Zielbereich) zu erklären, d.h. eine Metapher lässt „etwas“ stellvertretend für „etwas anderes“ stehen. Metaphern machen neue, komplizierte Erkenntnisse plausibel.

Metaphern sind für unsere Sprache nicht ungewöhnlich, vielmehr gewinnt die Sprache an Ausdrucksmöglichkeiten. Einige Beispiele für Metaphern seien hier angeführt:

Bedeutung	*Metapher*
Größe	*… wie ein Elefant …*
Über sich nachdenken	*Nabelschau*
Aufgeben	*… die Flinte ins Korn werfen*
arrogant	*hochnäsig*
Kamel	*Wüstenschiff*
Inhaltslos reden	*Leeres Stroh dreschen*

Tabelle 3.1: Einige Beispiele für Metapher in der Sprache

Metaphern sind nicht allgemeingültig

Metaphern sind allerdings nicht allgemeingültig. Sie existieren oft nur innerhalb von Kulturkreisen. Die Eule, etwa, als Symbol der Weisheit im westlichen Kulturkreis gilt in Südostasien als besonders dummes und böses Tier.

Metaphern sind nicht nur ein sprachliches Ausdrucksmittel. Sie beruhen auf Analogien und Analogien sind ein wichtiger Aspekt des Lernens. Lernen findet häufig über Analogien statt, ein Umstand, den man besonders deutlich bemerkt, wenn man jemanden etwas erklären soll, von dem er keine Ahnung hat.

Interface Metaphern spielen im User Interface Design eine große Rolle.

Metaphern spielen im User Interface Design bzw. ganz allgemein in der Mensch Computer Interaktion eine große Rolle. Im Gegensatz zu den oben erwähnten verbalen Metaphern werden die Interface Metaphern nicht explizit benannt, sondern machen, als Teil der Benutzeroberfläche, die Bedienung eines Programms intuitiver und leichter. Interface Metaphern können einzelne Interfacefunktionen oder auch ganze Benutzeroberflächen sein und stellen eine Analogie zu Objekten aus der realen Welt dar. Ein Papierkorb etwa dient dazu Weggeworfenes aufzunehmen. Als Icon auf dem Bildschirm hat er eine intuitiv verständliche Funktion. Der Benutzer kennt das in der Metapher Dargestellte aus der Realität und kann dadurch Rückschlüsse auf den Gebrauch und Funktion ziehen.

Unbekannte Metaphern

Im ungünstigen Fall, meint Prein ([Prein999], S. 169), haben die Benutzer keine Assoziationen mit der Metapher, womit die Orientierung an der Metapher bedeutungslos wird. Oder sie Überschätzen die Leistung des Systems, weil sie eine Funktionalität in die Metapher interpretieren, die nicht realisiert ist.

Eine ausführliche Beschäftigung mit der Thematik Metaphern in der Informatik findet sich in [Steffen2006].

3.3.2. Metaphertypen und Beispiele

Metaphern beruhen auf Analogien, die wiederum unterschiedliche Zusammenhänge aufweisen:

- Visuelle Analogie: beruht auf ähnlichem Aussehen
- Funktionelle Analogie: beruht auf Ähnlichkeiten in der Aufgabe
- Strukturelle Analogie: beruht auf Ähnlichkeiten in der Struktur

In der folgenden Tabelle werden Metaphern dargestellt, die sich aus diesen Analogien ableiten lassen:

Analogie	*Metapher*	*Bestandteile*
Visuell *Sieht aus wie …*	*ein Werbespot*	*Bewegung, Unterhaltung, Produkt,*
Funktionell *Macht das Gleiche wie …*	*ein Versandkatalog*	*Preislisten, Bilder der Produkte, Bestellmöglichkeit. …*
Strukturell *Ist aufgebaut wie …*	*eine Stadt*	*Häuser, Türen, Gänge, Treppen, Etagen, , Strassen. …*

Tabelle 3.2: Analogien und Metaphern.

Desktop-Metapher

Bei der Entwicklung von graphischen Benutzeroberflächen werden häufig Metaphern eingesetzt. Das wahrscheinlich bekannteste Beispiel für Metaphern in Computeranwendungen ist die Desktop-Metapher. Nach Prein ([BernhardPrein:1999], S. 170) geht es dabei darum, virtuelle Objekte zu entwickeln die an reale Objekte, mit ihren Funktionen, aus dem Büroalltag erinnern. Durch diese Analogie wird der Umgang mit dem System (der Computeranwendung) vereinfacht. Systeme Desktop-Metaphern präsentieren sich als virtueller Schreibtisch. Aus unserer Erfahrung mit solchen Systemen wissen wir, dass auf dem Schreibtisch (Desktop) Dokumente abgelegt und in Ordnern sortiert werden können. Sie können in den Papierkorb bewegt werden, wobei

sie erst nach dem Leeren des Papierkorbs gelöscht werden, und vieles mehr. Mit der Desktop-Metapher sind aber auch Aktionen möglich, die ein realer Schreibtisch nicht bietet. Dokumente können zum Beispiel skaliert werden oder der Desktop kann automatisch aufgeräumt werden.

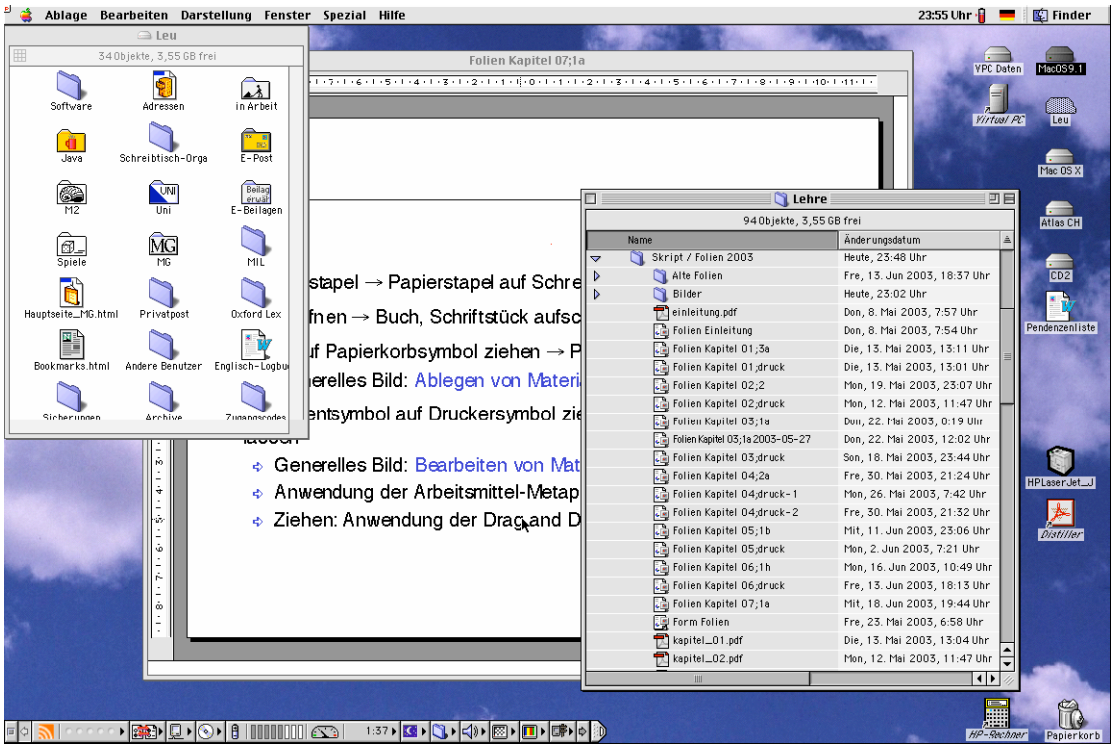


Abbildung 3.9: Die Desktopmetapher, wie sie sich in MacOS 9.1 präsentiert

Prein ([BernhardPrein:1999], S. 170) nennt noch andere Beispiele für Metaphern:

Beispiele für Metaphern:
Raummetapher
Reisemetapher
Buchmetapher
Funktionsmetapher

- Raummetapher: durch Einsatz von raumorientierten Anordnungsformen verankern sie die Information in einer räumlichen Dimension. Sie nutzen unsere Gewohnheit aus sich räumlich zu orientieren Der Anwender kann etwa durch eine virtuelle Stadt oder über eine Landkarte geführt werden. Die Informationseinheiten sind durch Querverbindungen, wie etwa Gänge oder Türen, verbunden.
- Reisemetapher: der Anwender wird in die räumlich-zeitliche Dimension, über die es etwas zu lernen gilt, versetzt. Er hat ein Ziel zu erreichen und muss dazu Aufgaben erledigen, Kämpfe bestehen oder Hindernissen ausweichen und darf dabei das Ziel nicht aus den Augen verlieren. Die Aufgaben finden meist an einem bestimmten Ort statt und deren positive Erledigung bringt den Anwender dem Ziel näher, erlaubt ihm das Erreichen des nächsten Ortes. Das ist auch der Unterschied

zur Raummetapher, wo die Verbindungen zwischen den Informationseinheiten immer offen sind. Die Reisemetaphern finden sich bei Computerspielen und vielen Lehrsystemen.

- Buchmetapher: hier werden Metaphern aus dem Bereich des Buchdrucks mit einer linearen Abfolge von Bildschirmseiten kombiniert. Die Buchmetapher wird häufig bei Bildschirmmedien verwendet, wie etwa bei Ausgaben von Online Zeitungen. Über Icons können die Anwender vor und zurück blättern und auch größere Sprünge machen, etwa an den Anfang oder das Ende des Textes. Diese Form der Navigation entspricht ansatzweise der gewohnten Situation bei gedruckten Texten. In der Beschreibung von Hypertextsystemen spricht man auch von Inhaltsverzeichnissen, Kapiteln und Seiten

Wichtig erscheint, in Hinblick auf Benutzeroberflächen, auch die:

- Funktionsmetapher: Die Funktionen von realen, physischen Objekten wie Taschenrechner, Telefon, Radio usw. werden nachgebildet. Um die Bedienung zu erleichtern, orientiert sich die Benutzeroberfläche an den Elementen und Funktionen der nachgebildeten physischen Objekte (siehe Abbildung 3.10).

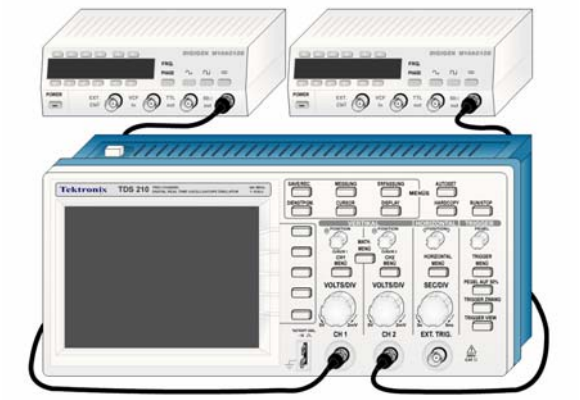


Abbildung 3.10: Die Abbildung zeigt ein virtuelles Oszilloskop und zwei Funktionsgeneratoren. Simuliert wird die Oberfläche des realen Oszilloskops Tektronix TDS 210 sowie des Funktionsgenerators DIGIGEN M10AC15D . Durch Einschalten und Drehen an den Knöpfen kann die Bedienung der Geräte geübt werden.

(http://www.virtphys.uni-bayreuth.de/elek/source/scope083.swf **)**

Ein komplexes System kombiniert unterschiedliche Metaphern

Ein komplexes System besteht immer aus einer Kombination unterschiedlicher Metaphern – die aktuellen Fensteroberflächen sind ein gutes Beispiel dafür. Aus eigener Erfahrung wissen wir auch, dass diese Kombination unterschiedli-

cher Analogien für uns gewöhnlich kein Problem darstellt. Prein ([Bern-hardPrein:1999], S. 177) führt aus, dass der Anwender diese Metaphern tatsächlich gar nicht bewusst wahrnimmt, weil sie nicht explizit in der Oberfläche repräsentiert werden.

3.4. Zusammenfassung

Heute werden in Krankenhäusern große Mengen an Patientendaten gesammelt und archiviert. Für die Forschung sind diese Daten vom großen Wert. Bei der Nutzung dieser wertvollen Daten ergibt sich in der Praxis ein grundlegendes Problem — der forschende Mediziner hat meist nicht ausreichende Datenbankkenntnisse um an die interessierende Information zu kommen.

Die Betrachtung der visuellen Anfragesprachen in Abschnitt 3.1, die Modellierung der Metadaten bzw. deren Betrachtung in Abschnitt 3.2 und die Verwendung von Metaphern bzw. deren Betrachtung im Abschnitt 3.3 bildet eine wichtige Grundlage für die weiteren Überlegungen zur Realisierung von VISAmed.

KAPITEL

4

Visuelle Sprachen

Wenn weise Männer nicht irrten,
müßten die Narren verzweifeln...

Johann Wolfgang von Goethe

Die Datenanalyse ist ein breit gefächertes Gebiet, dass eine Vielzahl von Systemen zu ihrer Unterstützung hervorgebracht hat. Unterschiedliche Anwendungsgebiete und Anwendungsgruppen stellen eigene Anforderungen an Benutzerschnittstellen und Funktionalität der einzelnen Systeme.

An Anfragesprachen für klinische Datenbanken werden hohe Anforderungen gestellt.

Im Allgemeinen arbeiten Benutzer mit einer Datenbank indem sie über eine Anfragesprache Anfragen stellen oder Veränderungen an den Daten vornehmen. Im Gegensatz zu Anfragesprachen in traditionellen Datenbanken werden an Anfragesprachen für klinische (Forschungs-) Datenbanken weitaus höhere Anforderungen gestellt. So enthalten klinische Daten unter Anderem räumliche und zeitliche Merkmale, die durch eine Anfragesprache berücksichtigt werden müssen. Außerdem werden die Anfragen an klinische Datenbanken meist von unerfahrenen Datenbankanwendern gestellt. Anfragesprachen für klinische Datenbanken müssen dies berücksichtigen und benötigen deshalb spezielle Konzepte bei der Anfrageformulierung.

4.1. Eine Auswahl visueller Sprachen

In diesem Abschnitt werden einige visuelle Sprachen vorgestellt. Es geht dabei nicht um die Qualität, Vorteile oder

Visuelle Programmiersprachen, visuelle Modellierungssprachen und visuelle Anfragesprachen

Nutzen der Sprachen an sich, sondern vielmehr sollen sie der Ideenfindung für die visuellen Konstrukte von VISAmed dienen, das im nächsten Kapitel besprochen wird. Zur Orientierung erfolgt die Einteilung in visuelle Programmiersprachen, visuelle Modellierungssprachen und visuelle Anfragesprachen:

- *Visuelle Programmiersprachen*: unter einer visuellen Programmiersprache versteht man eine Sprache die visuelle Elemente, von semantischer und syntaktischer Bedeutung enthält, ihre Syntax auf der räumlichen Anordnung und den Verbindungen zwischen den visuellen Elementen aufbaut und ihre Semantik aus der Interpretation der syntaktischen Strukturen bezieht.
- *Visuelle Modellierungssprachen*: visuelle Modellierungssprachen werden in den frühen Phasen der (Software-) Entwicklung oder Analyse eingesetzt, um Anforderungen an ein System festzulegen. Bekanntester Vertreter ist im Moment die objektorientierte Modellierungssprache UML[27]. Diese Sprachen versuchen eine Spezifikation für ein System durch Darstellung in Diagrammform möglichst verständlich zu machen.
- *Visuelle Anfragesprachen*: allgemein ist eine Anfragesprache eine Sprache zur Suche nach Information in Datenbeständen. Ihr Ergebnis ist eine Teilmenge des zugrunde liegenden Datenbestandes. Man spricht auch von Filterung der Daten. Visuelle Anfragesprachen unterstützen die Suche nach Information durch geeignete graphische Konstrukte und erlauben dem Anwender die Formulierung einer Anfrage ohne der textuellen Ebene.

4.1.1. Visuelle Programmiersprachen

Visuelle Programmiersprache vs. visuelle Programmierumgebung

Die Begriffe visuelle Programmiersprache und visuelle Programmierumgebung[28] werden gerne synonym verwendet. Eine Unterscheidung ist jedoch unbedingt notwendig. Die Definition der visuellen Programmiersprachen wurde bereits im Abschnitt 3.1.2 behandelt. Demnach benutzen visuelle Programmiersprachen mehr als eine Dimension, um die Semantik zu vermitteln, während visuelle Programmierumgebungen visuelle Benutzeroberflächen benutzen, um Teile textlicher Programme zu erzeugen, außerdem erhalten

[27] UML (Unified Modeling Language) siehe Abschnitt 4.1.2.2 auf Seite 78

[28] Unter einer visuellen Programmierumgebung versteht man eine Entwicklungsumgebung, die es dem Anwender ermöglicht, dialogorientierte Komponenten und andere Teile eines Programms "visuell" am Bildschirm zu erstellen. Der Anwender muss dazu keinen Programmcode eingeben, sondern dieser wird von der Programmierumgebung aus der interaktiv erstellten Programmbeschreibung generiert.

sie ihre syntaktische und semantische Bedeutung nicht durch visuelle Elemente, sondern aus geschriebenem Code. Die bekanntesten visuellen Programmierumgebungen, aber keine visuellen Programmiersprachen, sind Visual Basic von Microsoft und Delphi von Borland.

„Deutsch Limit", auch bekannt als Screen Space Problem

Der Nutzen visueller Programmiersprachen, die sich wie etwa C oder Java für das Erstellen beliebiger Programme eignen, ist jedoch heftig umstritten. Dazu sei auf das „Deutsch Limit"[29], auch bekannt als Screen Space Problem, verwiesen, nach dem es das Problem mit visuellen Programmiersprachen ist, dass nicht mehr als 50 primitive visuelle Objekte auf einem Bildschirm Platz finden.

Im Folgenden werden zwei ausgewählte Beispiele für visuelle Programmiersprachen vorgestellt. LabVIEW ist eine der wenigen visuellen Sprachen, die in der Wirtschaft angewendet wird, und ist ein Beispiel für eine datenflussorientierte visuelle Programmiersprache. AgentSheets ist ein Werkzeug zum einfachen Erstellen von Spielen und Simulationen und ein Beispiel für eine regelorientierte visuelle Sprache. In den USA wurde AgentSheets, aufgrund seiner einfachen Handhabbarkeit, schon bei verschiedenen Projekten mit Schülern erfolgreich genutzt.

4.1.1.1. LabVIEW

LabVIEW, eine datenflussorientierte visuelle Programmiersprache

LabVIEW [LabView2007] steht für "Laboratory Virtual Instruments Engineering Workbench" und ist ein für Anwendungen in der Mess- und Steuerungstechnik geeignetes, visuelles Programmiersystem. Das System vereint ein Datenflussmodell mit Modularisierung und Abstraktion und hat sich in verschiedenen Branchen etablieren können.

Grafische Programmiersprache G

LabVIEW besitzt seine eigene grafische Programmiersprache G. Programme werden grafisch in Form von Flussdiagrammen erstellt. Sie werden „Blockdiagramme" genannt. Solche Diagramme eliminieren viele syntaktische Details konventioneller Hochsprachen. LabVIEW benutzt Therme, Symbole und Konzepte die dem Ingenieur und Wissenschafter vertraut sind. LabVIEW definiert, im Gegensatz zu textbasierten Programmiersprachen, ein Programm mit grafischen Symbolen. LabVIEW Programme werden virtuelle

[29]Fred Lakin und Peter Deutsch unterhielten sich über grafische Programmierumgebungen, als Deutsch fragte:
"Well, this is all fine and well, but the problem with visual programming languages is that you can't have more than 50 visual primitives on the screen at the same time. How are you going to write an operating system?" (URL: http://www.dcc.uchile.cl/~rbaeza/cursos/vp/todo.html (27.01.2007))
Inspiriert durch diese Bemerkung prägte Lakin den Begriff `Deutsch Limit', der das Maximum an Symbolen bezeichnet, das sich in einer grafischen Programmierumgebung überschaubar darstellen lässt.

VI, Virtuelle Instrumente

Instrumente (Virtual Instruments), kurz VI, genannt, da ihre Erscheinung und Funktion Hardwaregeräten ähnlich ist. Diese VIs funktionieren wie die Hauptprogramme, Funktionen oder Subroutinen in konventionellen Programmiersprachen wie C, Java oder BASIC. VIs besitzen ein interaktives Benutzeroberflächen- und Quellcodeäquivalent die Daten beliebig austauschen können.

3 Hauptkomponenten

Ein VI (Virtual Instrument), besteht aus drei Hauptteilen:

- Das „front panel“: Dies ist die interaktive Benutzeroberfläche oder Variablenschnittstelle eines VIs. Es simuliert die Frontplatte eines Geräts mit Knöpfen, Schaltern, Grafiken und vielen andern controls (Benutzereingaben) und indicators (Programmausgaben). Der Benutzer gibt die Daten über die Tastatur oder mit der Maus ein und sieht das Resultat nach der Verarbeitung durch das Programm auf dem Bildschirm. Ein Beispiel eines front panel wird in Abbildung 4.1 gezeigt.
- Das Blockdiagramm, gezeigt ebenfalls in Abbildung 4.1, ist der Quellcode des VIs, konstruiert mit LabVIEWs grafischer Programmiersprache G. Dieses Blockdiagramm, obwohl es sehr bildhaft aussieht, ist das lauffähige Programm. Die Teile des Blockdiagramms, die Symbole (icons), stellen untergeordnete VIs dar — von LabVIEW zur Verfügung gestellte Funktionsmodule und Programmstrukturen. Die Blöcke werden mit Drähten (wires) zweckmäßig verbunden. Sie stellen die Variablen dar, welche den Datenfluss im Blockdiagramm und somit den Ablauf des Programms bestimmen.
- Das Symbol (icon) und der Verbinder (connector) eines VIs erlauben den Transfer der Daten zu einem anderen VI. Das Symbol repräsentiert ein VI im Blockdiagramm eines übergeordneten VIs. Der Verbinder definiert die Ein- und Ausgänge des VIs. VIs können hierarchisch und modular in Baumstrukturen aufgebaut werden. Sie können entweder ein top-level-Programm oder ein beliebiges Unterprogramm sein. Ein VI in einem höheren VI wird, in Analogie zur Subroutine, subVI, genannt. Mit diesen Möglichkeiten unterstützt LabVIEW das modulare Programmieren. Zuerst wird eine Applikation in eine Reihe einfacher Module aufgeteilt. Danach baut man die funktionsfähigen Teilbausteine zusammen und kombiniert sie im top-level-Diagramm. LabVIEW fördert eine modulare Programmierstruktur, da jedes subVI autonom lauffähig ist und somit leicht individuell getestet werden kann. Die low level subVIs können von

Programmen an unterschiedlichen Stellen aufgerufen werden.

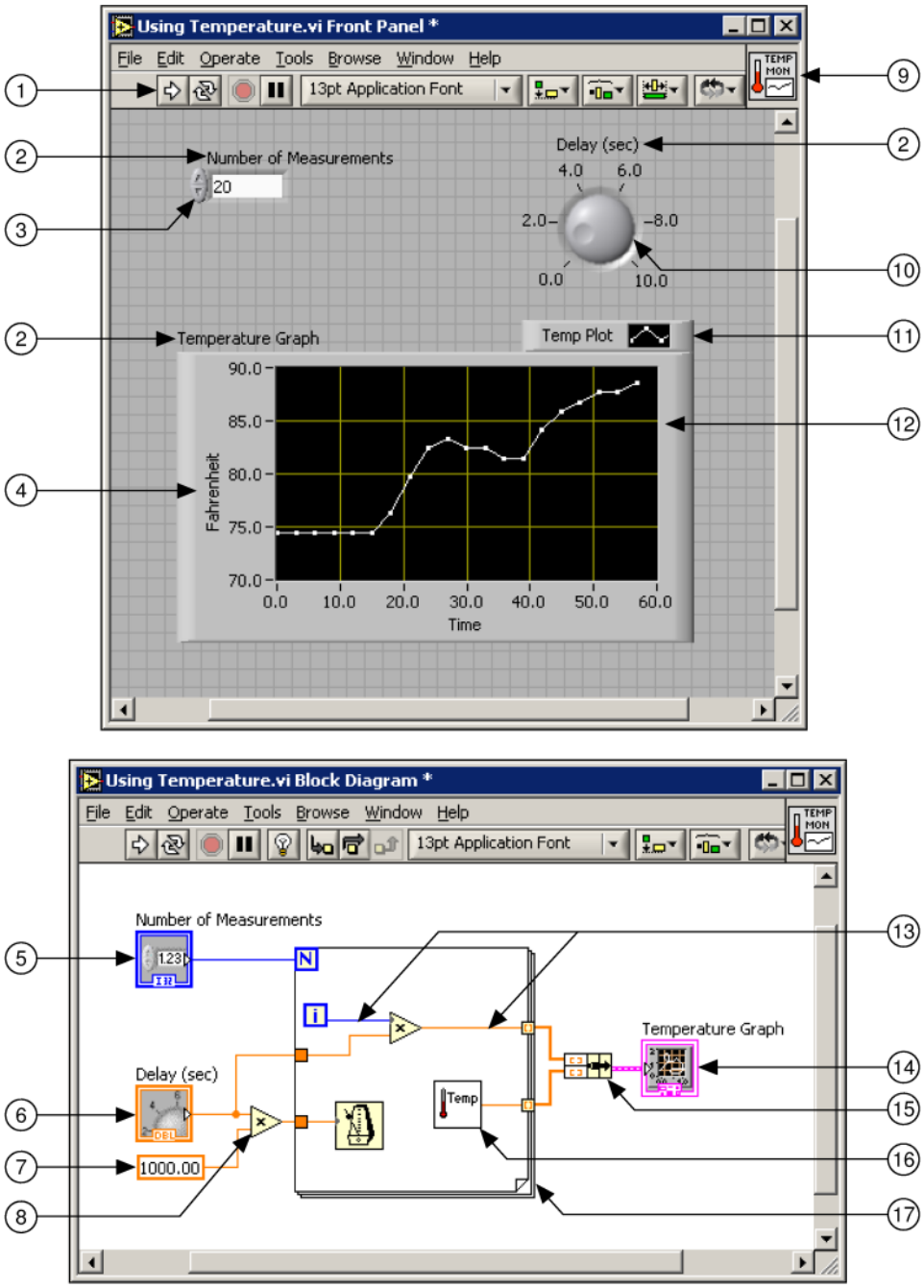


Abbildung 4.1: Schalttafel und Blockschaltbild in LabView: 1. Werkzeugleiste, 2. Beschriftung, 3. Numerischer Kontrollschalter mit Feld, 4. Freie Beschriftung, 5. Numerisches Steuerelement, 6. Knopf-Steuerelement, 7. Numerische Konstante, 8. Multiplikator-Funktion, 9. Icon, 10. Knopf, 11. Legende, 12. XY-Graph, 13. Verbindungen, 14. XY-Steuerelement, 15. Vereinigung-Funktion, 16. Unterprogramm (Visuelles Element), 17. For-Schleife (Reproduziert nach [LabView2007]).

Um ein erstelltes visuelles Instrument wieder verwenden zu können, ist es nötig, die Eingaben und Ausgaben, die

auf der Frontplatte erstellt wurden, über einen Verbinder als Anschlusspunkte an ein Icon zu binden. Das Icon kann anschließend in andere Instrumente eingefügt werden. An seinen Anschlusspunkten können verschiedene Daten angelegt bzw. von ihnen abgerufen werden. In Abbildung 4.1 ist unter Punkt 16 so ein visuelles Element abgebildet.

Schleifen

Das grau umrandete Rechteck (Punkt 17 in Abbildung 4.1) beschreibt eine For-Schleife, die ausdrückt dass alles im Rechteck so oft ausgeführt wird, wie im Feld unter Punkt 3 eingestellt wurde. Die Punkte an den Seiten des Rechtecks zeigen die Schnittstellen der Schleife. Innerhalb der For-Schleife befindet sich das oben erwähnte, erstellte visuelle Instrument, dass die Temperatur misst.

Die Verwendung der imperativen Elemente gestaltet sich aufgrund des Datenflusskonzeptes für den ungeübten Nutzer oft schwierig

Wie obiges Beispiel gezeigt hat stellt LabView nicht nur Datenflusselemente sondern auch Verzweigungen, Schleifen, Fallunterscheidungen, Sequenzen sowie lokale und globale Variablen zur Verfügung. Die Verwendung dieser imperativen Elemente gestaltet sich aufgrund des Datenflusskonzeptes für den ungeübten Nutzer allerdings oft schwierig. So kann etwa bei Schleifen auf Werte der letzten Iteration nur über Schieberegister zugegriffen werden. Rekursionen und Veränderungen des Programms zur Laufzeit sind nicht möglich.

Der Benutzer wird schon während der Programmerstellung über mögliche Probleme informiert

Während der Verkabelung sind alle Verbindungen die einen Fehler verursachen würden gestrichelt gekennzeichnet. Hierdurch wird der Benutzer schon während der Programmerstellung über mögliche Probleme informiert. Beim Testen des erstellten visuellen Instrumentes wird der Datenfluss sehr anschaulich mithilfe von Token visualisiert. Somit wird dem Benutzer die Fehlererkennung auf zwei Arten vereinfacht.

LabView verwendet das datenflussorientierte Modell für seine visuellen Elemente

LabView verwendet das datenflussorientierte Modell für seine visuellen Elemente. Die Ausführung eines Blocks erfolgt nur, wenn alle notwendigen Daten an Eingabeschnittstellen verfügbar sind. Nach der Abarbeitung wird das Ergebnis an Ausgabeschnittstellen, zur weiteren Nutzung, zur Verfügung gestellt. Die meisten textbasierten Programmiersprachen, wie Visua Basic, C++, Java, aber auch die weiter unten beschriebene visuelle Programmiersprache AgentSheets, verwenden das regelorientierte Modell bei der Programmabarbeitung. Im regelorientiertem Modell bestimmt die sequentielle Anordnung der Elemente, die Abarbeitungsreihenfolge der Befehle im Programm.

4.1.1.2. AgentSheets

AgentSheets [AgentSheets07] kombiniert Agenten, das sind kleine bewegliche Objekte mit speziellen Verhalten, mit

Worksheets, das ist die Umgebung, in der sich die Agenten bewegen können. Die Agenten können vom Programmierer mit „wenn–dann“ Regeln manipuliert werden. Somit können sie auf verschiedene Eingaben wie Mausklicks, Tastatureingaben, andere Agenten, Veränderungen von Attributen, Nachrichten oder sogar auf verschiedene Inhalte von Webseiten unterschiedlich reagieren. Die Reaktionen auf die Ereignisse können sehr verschieden sein. So können sie sich bewegen, ihr Aussehen und somit auch ihren Zustand ändern, neue Agenten erzeugen, andere löschen, Musik abspielen, Werte verändern, Nachrichten senden und Webseiten lesen.

AgentSheets kombiniert Agenten, die mit „wenn–dann“ Regeln manipuliert werden, mit Worksheets, das ist die Umgebung in der sich die Agenten bewegen können

Ein Worksheet ist eine Tabelle, die in jedem Feld einen oder mehrere Agenten übereinander aufnehmen kann. Die Agenten können sich in der Tabelle bewegen. Wenn die verschieden Agenten mit den charakteristischen Regeln entworfen sind, können sie frei im Worksheet platziert und eine Simulation gestartet werden. Es kann in verschiedenen Geschwindigkeiten sehr anschaulich beobachtet werden, wie die Agenten miteinander interagieren. Mit Ristretto wurde außerdem noch ein Tool implementiert, welches das entworfene AgentSheets-Projekt in ein Java Applet umwandelt. So können die entworfenen Simulationen sehr einfach im Internet einem breiten Publikum zur Verfügung gestellt werden.

Ristretto wandelt das entworfene AgentSheets-Projekt in ein Java Applet um

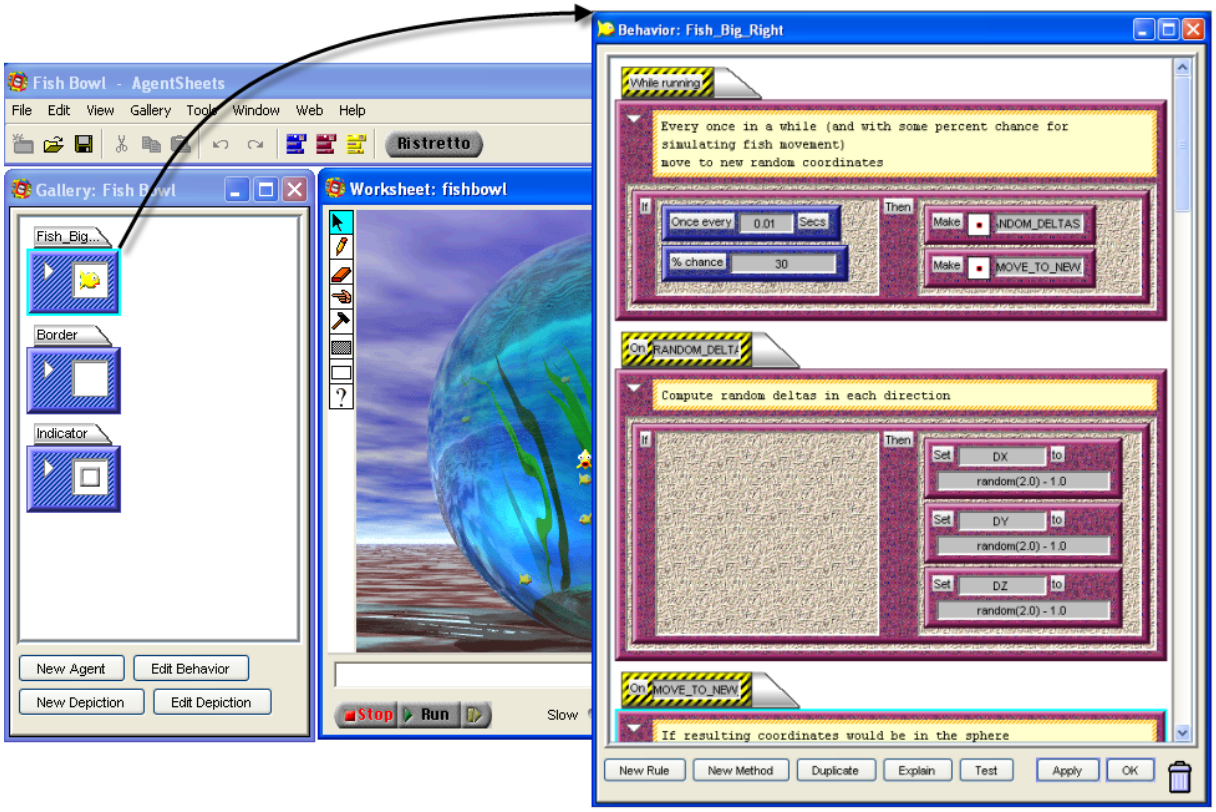


Abbildung 4.2: Die Entwicklungsumgebung von AgentSheets.

In Abbildung 4.2 ist ein kleines Projekt — Fish Bowl — zu sehen. Auf der linken Seite befindet sich die Gallery mit drei Agenten: Fish_Big_Right, Border und Indikator. Der Fisch hat drei Zustände, den normalen „Fisch schwimmt umher“-Zustand einen „Fisch schwimmt zur Glasscheibe“-Zustand und einen „Fisch schwimmt weg von der Glasscheibe“-Zustand. Auf der rechten Seite ist ein Worksheet zu sehen, auf dem in einem großen Fischglas einige Agenten angeordnet sind. Mit Symbolen wie Bleistift, Hammer und Radiergummi auf der linken Seite des Worksheets können die Agenten platziert, zu bestimmten Aktionen angeregt oder gelöscht werden. Mit dem Run Button wird die Simulation gestartet und an dem Schieberegler kann die Geschwindigkeit eingestellt werden.

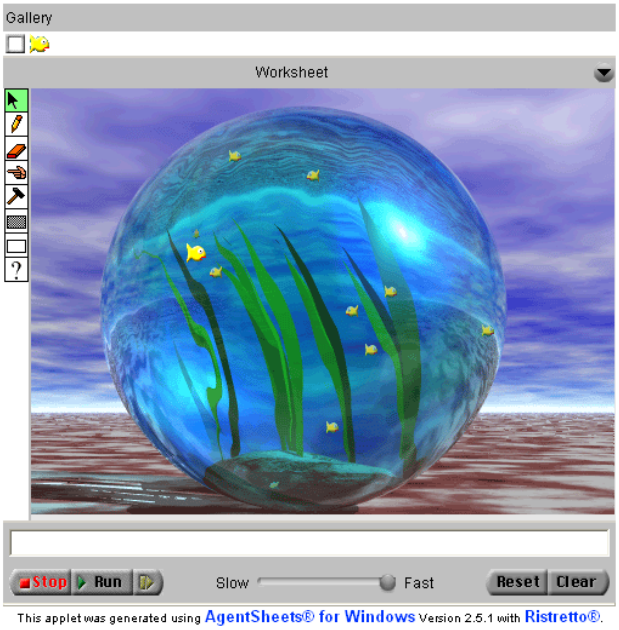


Abbildung 4.3: Mit Ristretto generiertes Applet für die Webseite.

Wenn die Simulation fertig entwickelt ist, kann - wie in Abbildung 4.3 zu sehen ist - mit wenigen Schritten ein Applet generiert werden. Bei der Interaktion des Benutzers mit dem Applet können auch Agenten in das Worksheet eingefügt bzw. aus diesem gelöscht werden. Die Agenten können mit dem Hammer Symbol zu einem bestimmten Verhalten angeregt werden. Somit besteht die Möglichkeit während der Laufzeit in das Programm einzugreifen. Eine Veränderung der Regeln zur Laufzeit ist allerdings nicht möglich.

4.1.2. Visuelle Modellierungssprachen

Visuelle Modellierung beinhaltet die Bedeutung der zwei Begriffe:

- visuell → grafisch, Diagramme
- Modellierung → Modelle

Visuelle Modellierungssprachen sind weit verbreitet

Im Gegensatz zu visuellen Programmiersprachen sind visuelle Modellierungssprachen deutlich weiter verbreitet. Möglicherweise liegt es daran, dass visuelle Modellierungssprachen mit Modellen arbeiten und diese wiederum mental leichter erfasst werden.

Modelle sind Abstraktionen, die notwendige Details eines komplexen Problems oder einer Struktur beschreiben, indem unwichtige Details herausgefiltert werden.

Modelle sind unter Anderem nützlich für das Problemverständnis, für die Kommunikation mit den Projektbeteiligten (Kunden, Experten, Analytiker, Designer, ...), für die Unternehmensmodellierung (Geschäftsprozessmodellierung), die Vorbereitung der Softwaredokumentation und für das Programm- und Datenbankdesign. Modellierung ermöglicht das bessere Verständnis der Anforderungen, ein exakteres Design und eine bessere Wartbarkeit des Systems. Modelle sind Abstraktionen, die notwendige Details eines komplexen Problems oder einer Struktur beschreiben, indem unwichtige Details herausgefiltert werden. Dadurch wird das Verständnis des Problems erleichtert. Die Konstruktion eines Modells erlaubt es, beispielsweise, dem Softwaredesigner, das Projekt aus verschiedenen Blickwinkeln abzubilden und den Detailgrad entsprechend der Bedürfnisse zu wählen. Dadurch kann man die Komplexität verringern und wesentlich schneller einen Einstieg auch in große Projekte gewinnen.

Anforderungen an die Modelle

Es gibt einige Anforderungen an die Verwendung von Modellen. Dazu gehören der Bau von Modellen unter Verwendung eindeutiger Notation, der Beweis, dass das Modell die Anforderungen an das System erfüllt sowie die stufenweise Transformation des Modells in eine Implementation [Frank2000].

Zwei der bekanntesten Vertreter der visuellen Modellierungssprachen — ER Diagramme und UML — werden anschließend vorgestellt.

4.1.2.1. ER-Diagramme

ER-Diagramme sind eine graphische Methode zur Darstellung komplexer Beziehungsgebilde der Realität

ER-Diagramme sind eine grafische Methode zur Darstellung komplexer Beziehungsgebilde der Realität. Die Realität ist gleichzeitig Grundlage für die logischen Beziehungen der entsprechenden Daten in einem relationalen Datenbanksystem. Die ER-Diagramme sind eine graphische Abbildung des ER-Modells.

ER-Modell, ist ein abstraktes Modell das in der Designphase einer Datenbank verwendet wird

Das Entity-Relationship-Model (ER-Modell) ist ein abstraktes Modell. Das Modell wird zwar nicht unmittelbar zur Implementierung eines Entwurfes verwendet, aber es wird gewöhnlich in der Designphase einer Datenbank verwendet da es weniger Einschränkungen und eine höhere Abstraktion als die meisten implementierten Datenbankmodelle bietet. Grundkonzepte des ER-Modells sind:

- Entität: Objekte oder Ereignisse der realen Welt
- Entitätstyp: Typisierung gleichartiger Entitäten
- Beziehung (Relationship): Beziehungen zwischen Entitäten
- Beziehungstyp: Typisierung gleichartiger Beziehungen
- Attribut: Eigenschaften von Entitäten
- Kardinalität: mögliche Anzahl der an einer Beziehung beteiligten Entitäten

ER-Diagramm beschreibt die Informationsstruktur einer Domäne im ER-Modell

Die Beschreibung der Informationsstruktur einer Domäne im ER-Modell wird Entity-Relationship-Diagramm (ER-Diagramm) genannt. Konventionen der Darstellung legen z.B. fest, dass ein Objekt als Rechteck, Beziehungen als Rauten und Merkmale (Attribute) als Kreise oder Ellipsen dargestellt und mittels Linien verbunden werden, an denen z.B. noch der Beziehungstyp notiert werden kann.

Eine Vielzahl unterschiedlicher Notationen beschreibt die gleiche Aussage

Es gibt heute eine Vielzahl unterschiedlicher Notationen, die sich in Klarheit, Umfang der grafischen Sprache, Unterstützung durch Standards und Werkzeuge unterscheiden. Im Folgenden finden sich einige wichtige Beispiele, die vor allem deutlich machen, dass bei allen grafischen Unterschieden die Kernaussage der ER-Diagramme nahezu identisch ist [WikiER07]:

- Chen-Notation von Peter Chen, dem Entwickler der ER-Diagramme.
- IDEF1X als langjähriger De-facto-Standard bei U.S. amerikanischen Behörden.
- Bachman-Notation von Charles Bachman als weit verbreitete Werkzeug-Diagramm-Sprache.
- Martin-Notation (Krähenfuß-Notation) als weit verbreitete Werkzeug-Diagramm-Sprache (Information Engineering).
- Min–Max-Notation als (kurzlebiger, siehe nächster Punkt) ISO-Standard.
- UML als heutiger Standard, den selbst ISO in eigenen Normen für ER-Diagramme verwendet.

Alle Notationen weiter unten drücken auf ihre Art die folgende Aussage aus:

Eine Person ist in maximal einem Ort geboren. Ein Ort ist Geburtsort von beliebig vielen Personen.

Ein Ort kann ein Geburtsort sein, muss es aber nicht sein.

Bis auf das Chen-Diagramm wird diese Aussage ergänzt um:

Eine Person muss in einem Ort geboren sein; bzw. ist in genau einem Ort geboren.

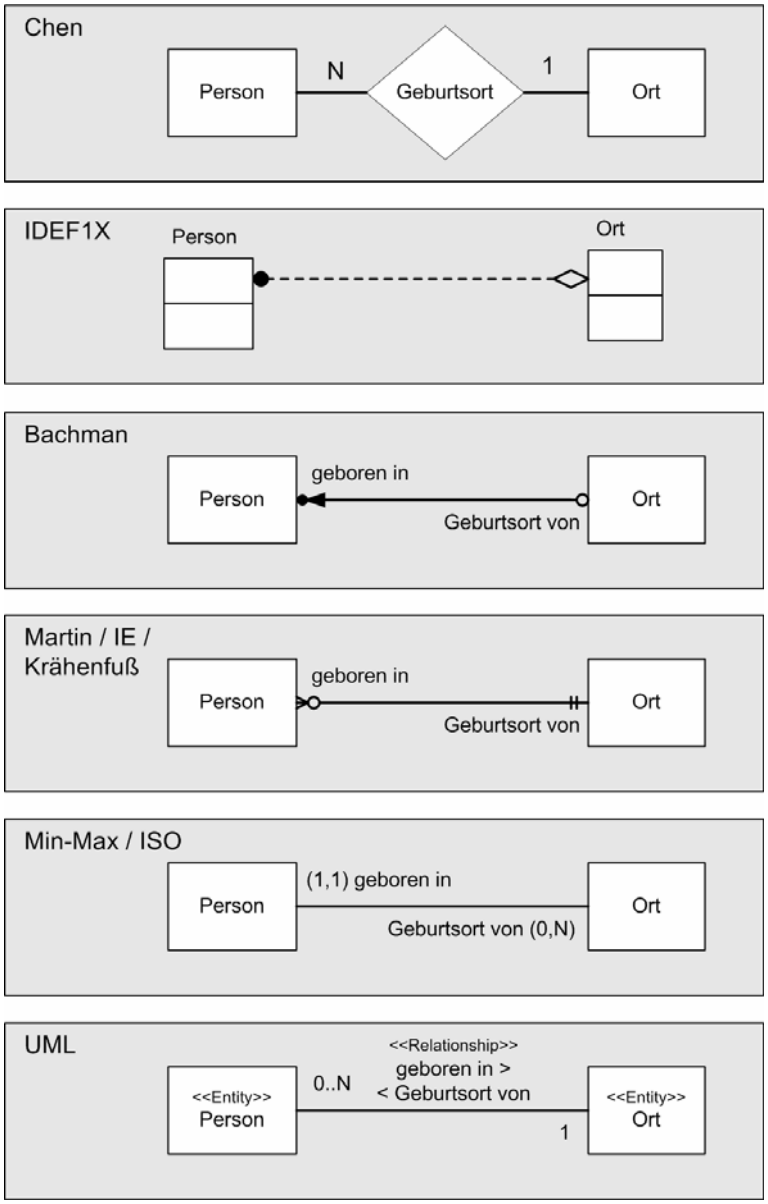


Abbildung 4.4: ER-Diagramm Beispiel in unterschiedlichen Notationen (aus [WikiER07])

Über das ER-Modell und die ER-Diagramme existiert sehr viel Literatur. Näheres kann z.B. in [Kemper2004] oder [Vossen2000] nachgelesen werden.

4.1.2.2. UML

UML ist von OMG standardisiert

Die Unified Modeling Language (UML) ist eine von der Object Management Group (OMG) standardisierte Sprache für die Modellierung von Software und anderen Systemen. UML definiert Bezeichner für die meisten Begriffe, die für die Modellierung wichtig sind und legt mögliche Beziehungen zwischen diesen Begriffen fest. UML definiert weiter graphische Notationen für diese Begriffe, sowie für Modelle von statischen Strukturen und dynamischen Abläufen, die man mit diesen Begriffen formulieren kann.

UML legt fest, mit welchen Begriffen und welchen Beziehungen zwischen den Begriffen Modelle spezifiziert werden

Die grafische Notation ist jedoch nur ein Aspekt, der durch UML geregelt wird. UML legt vor allem fest, mit welchen Begriffen und welchen Beziehungen zwischen diesen Begriffen Modelle spezifiziert werden. Die Diagramme der UML zeigen dabei nur eine grafische Sicht auf Ausschnitte dieser Modelle.

UML 1.x wurde 2005 durch UML2 ersetzt

Die erste Version der UML wurde in den 1990er Jahren von Grady Booch, Ivar Jacobson, James Rumbaugh sowie OMG entwickelt. Es war eine Zusammenfassung zahlreicher eigener Vorschläge für Modellierungssprachen, welche die zu dieser Zeit aufkommende objekt-orientierte Softwareentwicklung unterstützen sollten. Diese erste Folge von Sprachversionen, die unter dem Namen UML 1.x bekannt war, wurde 2005 durch eine grundlegend überarbeitete Version, unter dem Namen UML2, abgelöst.

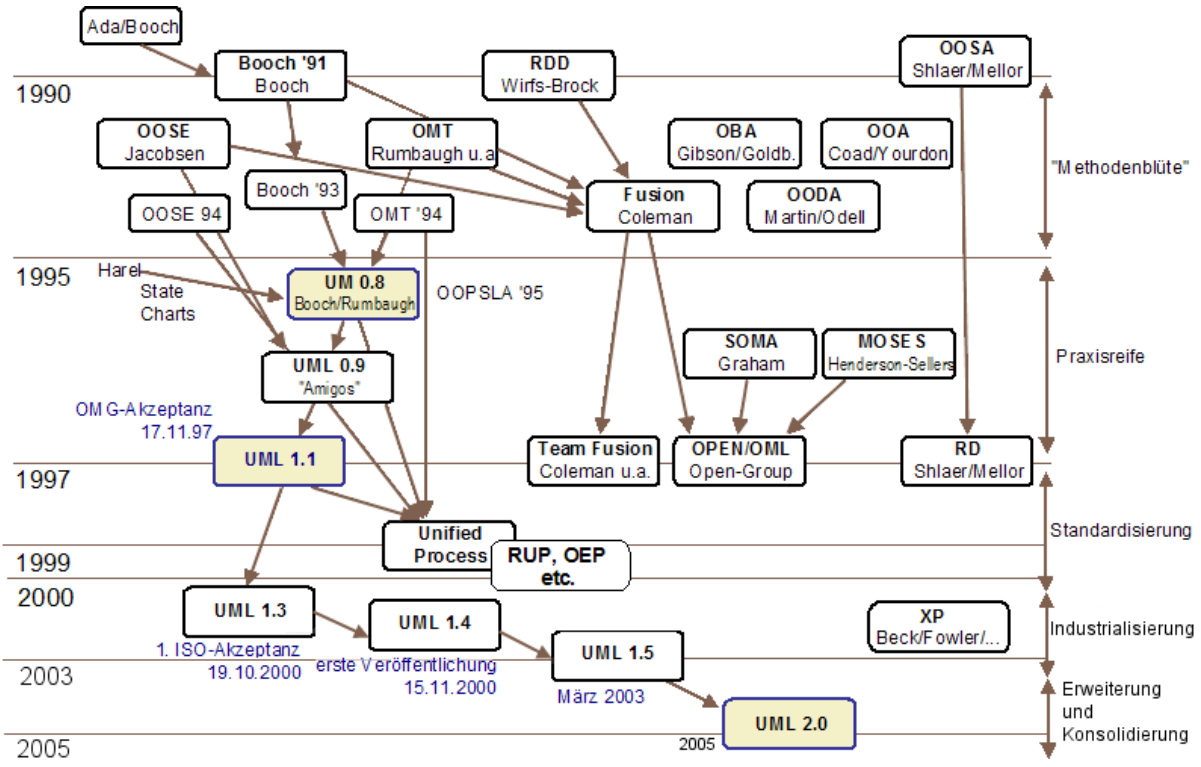


Abbildung 4.5: Entwicklungsschritte der Modellierungssprache UML

Eine sehr umfangreiche Materialsammlung zu UML findet sich auf den UML-Seiten von OMG unter der URL http://www.uml.org/. Weiters beschreibt [Hitz2005] ausführlich die Sprache UML 2.0.

4.1.2.2.1. Darstellung in Diagrammen

6 Strukturdiagramme

Die UML2 kennt *sechs* Strukturdiagramme:

- Klassendiagramm: Beschreibt den strukturellen Aspekt in Form von Klassen, Interfaces und Beziehungen
- Kompositionsstrukturdiagramm: Zeigt die interne Struktur von Modellelementen und ermöglicht dadurch eine hierarchische Dekomposition
- Komponentendiagramm: Zeigt interne und externe Sicht von Komponenten, sowie die Verknüpfung von Komponenten untereinander
- Verteilungsdiagramm: Zeigt die eingesetzte Hardware- und Software-Topologie in Form von Knoten und Kommunikationsbeziehungen sowie das zugeordnete Laufzeitsystem in Form von Artefakten
- Objektdiagramm: Beschreibt den strukturellen Aspekt in Form von Objekten und Links
- Paketdiagramm: Gruppiert Modellelemente auf verschiedenen Abstraktionsebenen zur Komplexitätsreduktion

7 Verhaltensdiagramme

Dazu kommen *sieben* Verhaltensdiagramme:

- Anwendungsfalldiagramm: Beschreibt die Funktionalität des zu entwickelnden Systems aus Benutzersicht
- Aktivitätsdiagramm: Dient zur Modellierung von prozeduralen Abläufen in Form von Kontroll- und Datenflüssen
- Sequenzdiagramm: Beschreibt komplexe Interaktionen zwischen Objekten in bestimmten Rollen, um eine konkrete Aufgabe zu erfüllen Fokus: Zeit als eigene Dimension
- Kommunikationsdiagramm: Beschreibt einfache Interaktionen Fokus: Strukturelle Beziehungen
- Interaktionsübersichtsdiagramm: Zeigt auf abstrakter Ebene den Kontrollfluss zwischen verschiedenen Interaktionsabläufen
- Zeitverlaufsdiagramm: Zeigt Zustandsänderungen der Interaktionspartner aufgrund von Zeitereignissen und Nachrichten

- Zustandsdiagramm: Beschreibt das erlaubte Verhalten von Modellelementen in Form von Zuständen und Zustandsübergängen

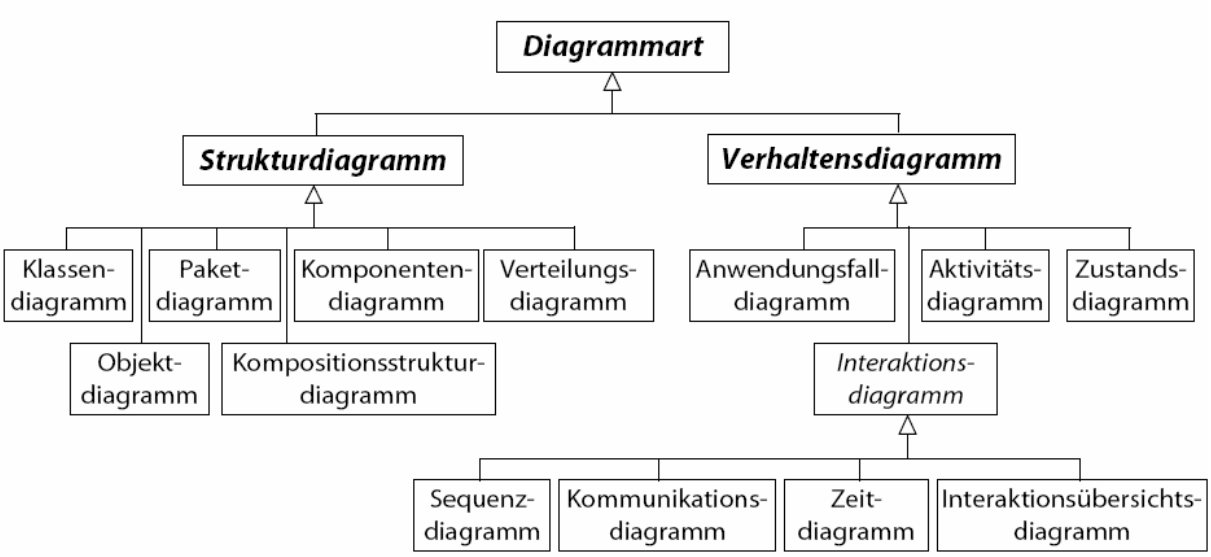


Abbildung 4.6: Die verschiedenen Diagrammtypen in UML 2.0

Die Grenzen zwischen den Diagrammtypen verlaufen nicht scharf

Die Grenzen zwischen diesen dreizehn Diagrammtypen verlaufen nicht sehr scharf. UML2 verbietet nicht, dass ein Diagramm graphische Elemente enthalten darf, die eigentlich zu unterschiedlichen Diagrammtypen gehören. Es ist sogar möglich, dass Elemente aus einem Strukturdiagramm und aus einem Verhaltensdiagramm auf dem gleichen Diagramm dargestellt werden, wenn damit eine besonders treffende Aussage zu einem Modell erreicht wird.

Austauschformat für Diagramme

UML2 definiert unter dem Namen UML 2.0 Diagram Interchange ein Austauschformat für Diagramme, so dass unterschiedliche Werkzeuge, mit denen Modelle basierend auf der UML2 erstellt werden, die Diagramme austauschen und wieder verwenden können. In der UML 1.x war das nur für die Repository-Modelle hinter den Diagrammen möglich, aber nicht für die eigentlichen Diagramme. Das Format basiert auf der Auszeichnungssprache XML und heißt XML Metadata Interchange (XMI). Die Grundlage für die Austauschbarkeit ist das MOF[30], auf dessen Konzept beide Sprachen, XMI und UML, beruhen.

Auf die jeweilige graphische Ausprägung und Details der oben besprochenen Diagramme wird nicht weiter eingegangen. Hierzu sei man auf einschlägige Literatur verwiese, wie etwa [Hitz2005].

[30] Meta-Object Facility (MOF) eine spezielle Metadatenarchitektur der Object Management Group. Die Spezifikation beschreibt eine abstrakte Sprache und ein Framework zur Verwaltung von plattformunabhängigen Metamodellen

4.1.2.2.2. Sprachspezifikation

Die UML Sprachspezifikation besteht aus einem Metamodell und zusätzlichen Integritätsbedingungen

UML ist eine sehr komplexe Sprache mit einer sehr umfangreichen Sprachbeschreibung. Die Sprachspezifikation besteht aus einem Metamodell und zusätzlichen Integritätsbedingungen. Das Metamodell stellt das Modell der Sprache dar. In Abbildung 4.7 sind die verschiedenen Sprachebenen dargestellt, die bei der Modellierung eines Ausschnitts der Realität zu beachten sind. UML stellt dabei das Metamodell mit Hilfe eines Klassendiagramms dar.

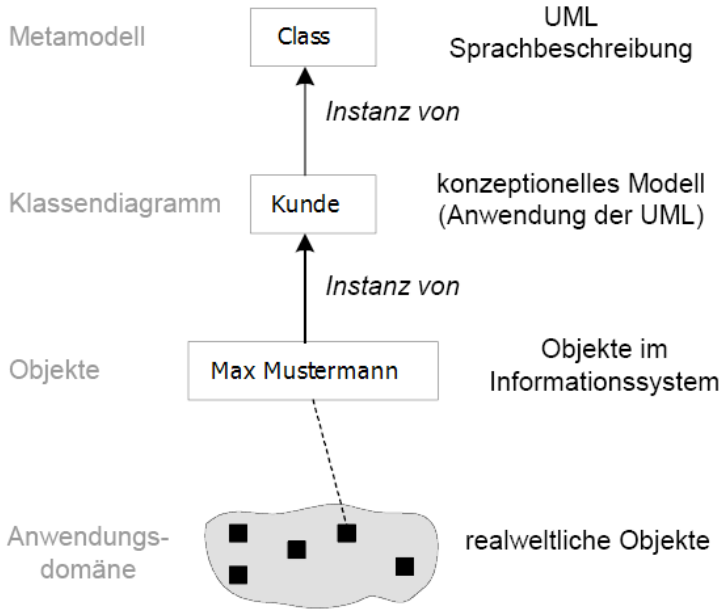


Abbildung 4.7: Ebenen der Modellierung (adaptiert nach [Frank2000])

Vorteile von Metamodellen

Die Verwendung von Metamodellen hat zwei Vorteile [Frank2000]:

- durch die Verwendung gleicher Konzepte für die Sprachbeschreibung und Sprachanwendung wird ein Paradigmenbruch in der Sprachspezifikation vermieden
- Metamodelle können einfach in Klassendiagramme überführt werden und stellen damit eine gute Grundlage für den Entwurf von Modellierungswerkzeugen.

In der Abbildung 4.8 wird ein Ausschnitt aus dem Metamodell des Klassendiagramms dargestellt. Eine Klasse („Class“) kann beliebig viele Eigenschaften („Feature“) haben, bei denen es sich entweder um Attribute oder Operationen handelt. Für die Assoziation zwischen „Konto“ und „Kontoinhaber“ wird die Klasse „Konto“, die eine Instanz von „Class“ ist, mit einer Instanz von „AssociationEnd“ verknüpft. Dieser Instanz ist die Kardinalität („multiplicity“) 1..* zuge-

ordnet. Zusätzlich ist sie mit einer Instanz von „Association“ verknüpft, die wiederum mit einer zweiten Instanz von „AssociationEnd“ (mit Kardinalität 1) verbunden ist. Diese Instanz von „AssociationEnd“ ist erneut mit einer Instanz von „Class“, nämlich der Klasse „Kontoinhaber“ verknüpft.

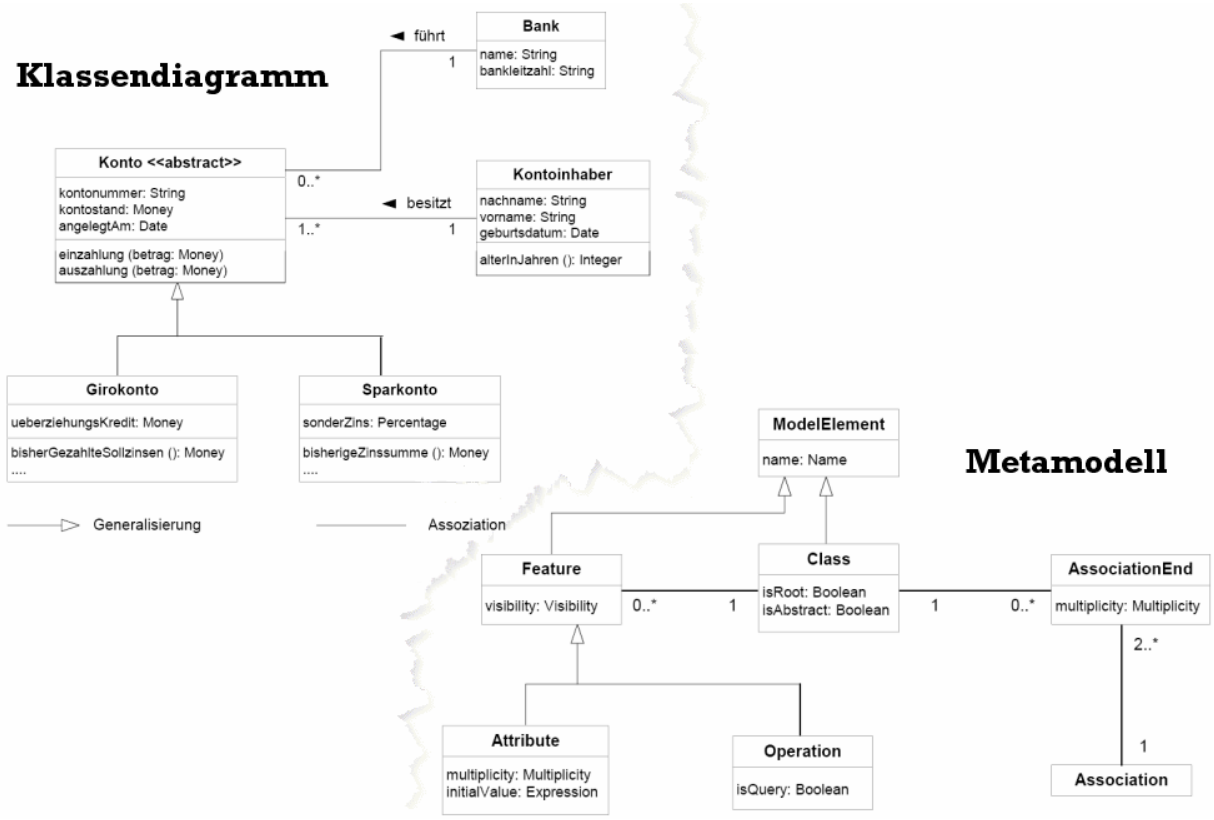


Abbildung 4.8: Zusammenhang zwischen Metamodell und Klassendiagramm (adaptiert nach [Frank2000])

UML erlaubt Round Trip Engineering

In UML2 sind alle Sprachelemente mit eindeutiger Semantik[31] versehen. Die Spezifikation ist in verschiedenen Detaillierungsgraden möglich – von sehr abstrakt bis sehr implementierungsnah. UML erlaubt Round Trip Engineering[32] bei der Abbildung von Modellen auf verschiedene Programmiersprachen.

[31] Die Semantik ist das Teilgebiet der Linguistik, das sich mit Sinn und Bedeutung von Sprache beziehungsweise sprachlichen Zeichen befasst

[32] Round Trip Engineering ist ein Begriff aus dem Gebiet der Softwaretechnik und sorgt für Konsistenz zwischen Implementierung und Diagrammen, es beinhaltet:
- Forward Engineering: Änderung der Diagramme führt zu einer Anpassung des Quelltexts
- Reverse Engineering: Änderung im Quelltext führt zur Anpassung der Diagramme
Einige Softwareentwicklungswerkzeuge (CASE) wie Together unterstützen simultanes Round Trip Engineering, d.h. die Aktualisierung erfolgt unmittelbar nach einer Veränderung.

4.1.3. Visuelle Anfragesprachen

VQS und VQL

Visuelle Anfragesprachen (VQL – Visual Query Language) sind einer der wichtigsten Bestandteile von visuellen Anfragesystemen (VQS – Visual Query System). In [Catarci93] heißt es:

> *"Visual Query Systems (VQSs) may be defined as query systems essentially based on the use of visual representations to depict the domain of interest and express the related requests. VQSs provide user-friendly query interfaces for accessing a database. They include both a language to express the queries in a pictorial form (i.e., a visual query language, VQL) and a variety of functionalities to facilitate man-machi-ne interaction. The VQSs are oriented to a wide spectrum of users who have limited technical skills and generally ignore the inner structure of the accessed database." (In [catarci93fundamental], S. 76)*

Und weiter:

> *"Within the class of visual programming languages, a subclass exists […] this subclass of languages is the one of visual query languages." (In [catarci93fundamental], S. 76)*

Eine der ersten visuellen Anfragesprachen war Query By Example (QBE). Manche heutigen kommerziellen Systeme basieren auf QBE, wie etwa Microsoft Access.

Visuelle Anfragesprachen sind auf den Gelegenheitsbenutzer ausgerichtet

Visuelle Anfragesprachen sind zumeist auf den Gelegenheitsbenutzer ausgerichtet. Das ist wenig verwunderlich, wenn man bedenkt, dass Gelegenheitsbenutzer einerseits meist keine Datenbankexperten sind, andererseits Datenbanken bequem abfragen wollen.

Zwei der Vertreter der visuellen Anfragesprachen – Kaleidoquery und QBE – werden nachfolgend vorgestellt.

4.1.3.1. Kaleidoquery

Kaleidoquery ist eine diagrammbasierte, visuelle Anfragesprache für objektorientierte Datenbanksysteme

Kaleidoquery ist eine diagrammbasierte, visuelle Anfragesprache für objektorientierte Datenbanksysteme. Kaleidoquery und seine Konstrukte werden im weiteren Verlauf des Kapitels in Anlehnung an [Murray1998] und [Murray2000] vorgestellt.

Kaleidoquery wird eine wichtige Basis für VISAmed darstellen, dementsprechend werden die einzelnen Konstrukte der Sprache im Folgenden genauer besprochen.

Die visuelle Syntax basiert auf OQL

Kaleidoquery hat eine visuelle Syntax, die auf der textuellen Anfragesprache OQL[33] (Object Query Language) basiert. Mit Kaleidoquery wurde versucht dem Gelegenheitsbenutzer ad-hoc Anfragen zu ermöglichen. Dem kommen die ikonbasierten Darstellungen der Anfragen entgegen, aus denen der Benutzer auf das dahinter liegende Datenbankschema intuitiv schließen kann.

Die "filter flow" Metapher

Die Grundidee hinter Kaleidoquery ist die "filter flow" Metapher, die bereits 1991 durch Ben Shneiderman vorgestellt wurde. Shneiderman verglich das Verarbeiten einer Anfrage mit fließendem Wasser, das durch etliche Rohre und Filter fließt. Die Rohre sind mit AND- und OR-Verbindungen verknüpft und die Filter lassen nur eine bestimmte Art von Durchfluss zu. Nach dem Prinzip des "filter flow"-Modells werden aus einer Menge von Informationen jene herausgefiltert, die für das Ergebnis der Anfrage relevant sind. Das steht im Gegensatz zum üblichen Vorgehen bei diagrammbasierten Anfragesprachen, bei denen der Anwender, ausgehend vom Datenbankschema, Teile des Schemas Auswählt und Anfragen darauf anwendet.

Einige Basissymbole von Kaleidoquery

Einige Basissymbole von Kaleidoquery werden in Abbildung 4.9 veranschaulicht. Das linke Symbol mit der Beschriftung Person bezeichnet eine Klasse Person. Das ovale Kästchen, inklusive Inhalt stellt die Wertemenge namens People der Klasse Person dar – People ist der Extensionsname der Klasse Person. Eine Extension einer Klasse ist die Menge, die zu jedem Zeitpunkt alle existierenden Ausprägungen (Objekte) der Klasse enthält. Analoges gilt für die Darstellung von Company und Companies.

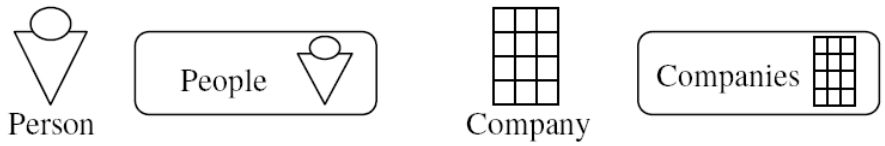


Abbildung 4.9: Darstellungssymbole in Kaleidoquery (aus [Murray2000]).

Einige Beispielanfragen

Drei Anfragen mit unterschiedlichen Projektionen werden in Abbildung 4.10 dargestellt. In der linken Anfragevisualisierung besteht das Ergebnis, durch das Anfügen der Attribute name und age hinter das Klassensymbol Person, aus-

[33] Die Object Query Language (OQL) ist eine stark an SQL angelegte Abfragesprache für Objektorientierte Datenbanken. Sie ist durch die Object Database Management Group (ODMG, http://www.odmg.org/) standardisiert. OQL wurde entwickelt, um die Interaktion zwischen objektorientierten Programmen und einer Datenbank zu vereinfachen, da der klassische relationale Ansatz für objektorientierte Programme ungeeignet ist und zu Brüchen in der Softwarearchitektur führt.

schließlich aus den Attributwerten zu name und age. Im charakteristischen OQL-Code sind diese Attribute in der select-Klausel angegeben. Die mittlere Darstellung repräsentiert eine Anfrage, in der keine speziellen Attribute zur Projektion ausgewählt werden. In OQL entspricht es dem Befehl *select**. Das Haken in der rechten Anfragevisualisierung zeigt, dass der Anwender, eine Menge von Attributen ausgewählt hat, deren Dokumentation er in der Visualisierung aufgrund der Menge nicht für sinnvoll hält.

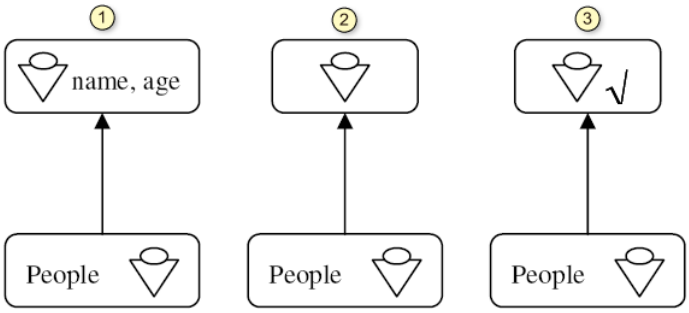


Abbildung 4.10: Visualisierung des Ergebnisses in Kaleidoquery. 1 und 3 können die gleiche Anfrage darstellen; 2 zeigt alle Attribute im Ergebnis an (nach [Murray2000]).

4.1.3.1.1. Ausdrücke, Vergleiche, Aggregatfunktionen

Ausdrücke

Einfache Ausdrücke werden in Kaleidoquery durch Anfügen von Attributen hinter das jeweilige Klassensymbol gestellt.

In Abbildung 4.11 wird das Gewicht einer Person mit 2,205 multipliziert um es von der Einheit Kilogramm in pounds (lbs) zu transformieren. Eine Benennung des Ausdrucks ist optional und kann durch Voranstellen des Namens und Klammerung des Ausdrucks erfolgen (im Beispiel: lbs(weight * 2.205)).

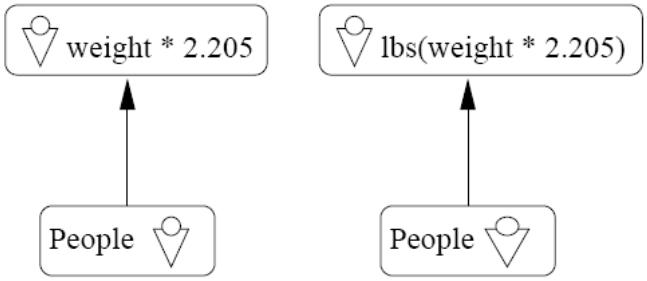


Abbildung 4.11: Ein einfacher Ausdruck in der Projektion (aus [Murray2000]).

Vergleiche

Kaleidoquery unterstützt auch Vergleiche von numerischen Datentypen, Zeichenketten und Mengen. Die Vergleichsoperatoren entsprechen denen von OQL (<=, <, >, >=, =, like, usw.). Abbildung 4.12 zeigt das Beispiel einer visuellen Repräsentation einer Vergleichsoperation und den dazugehörigen OQL-Code.

Abbildung 4.12: Ein einfacher Vergleich mit dazugehörigem OQL-Code (aus [Murray2000]).

Es werden alle Personen, jünger als 20 Jahre, ausgewählt. Da im Ergebnissymbol keine Attribute spezifiziert wurden, bezieht sich die Projektion auf alle Attribute der Klasse Person.

Aggregatfunktionen

Aggregatfunktionen wie COUNT, SUM, AVG, usw., werden in Kaleidoquery auf das Ergebnis der Anfrage angewendet. Die Aggregatfunktion wird dabei, wie in Abbildung 4.13 dargestellt, von einer weiteren Ergebnisbox umrandet.

Im Beispiel berechnet die erste Funktion die Anzahl der "People"; die zweite Funktion gibt das höchste Gehalt aus einer Menge von Gehältern aus.

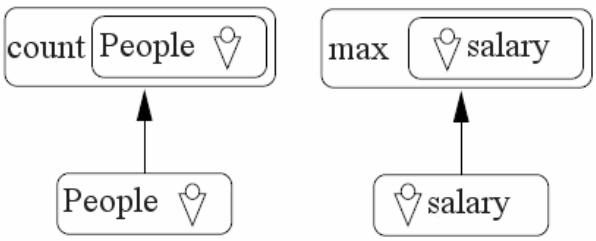


Abbildung 4.13: Aggregatfunktionen in Kaleidoquery (aus [Murray2000])

4.1.3.1.2. Boolesche Ausdrücke und Mengenausdrücke

AND, OR und NOT

Zur Verknüpfung von Bedingungen werden boolesche Ausdrücke (AND, OR und NOT) verwendet.

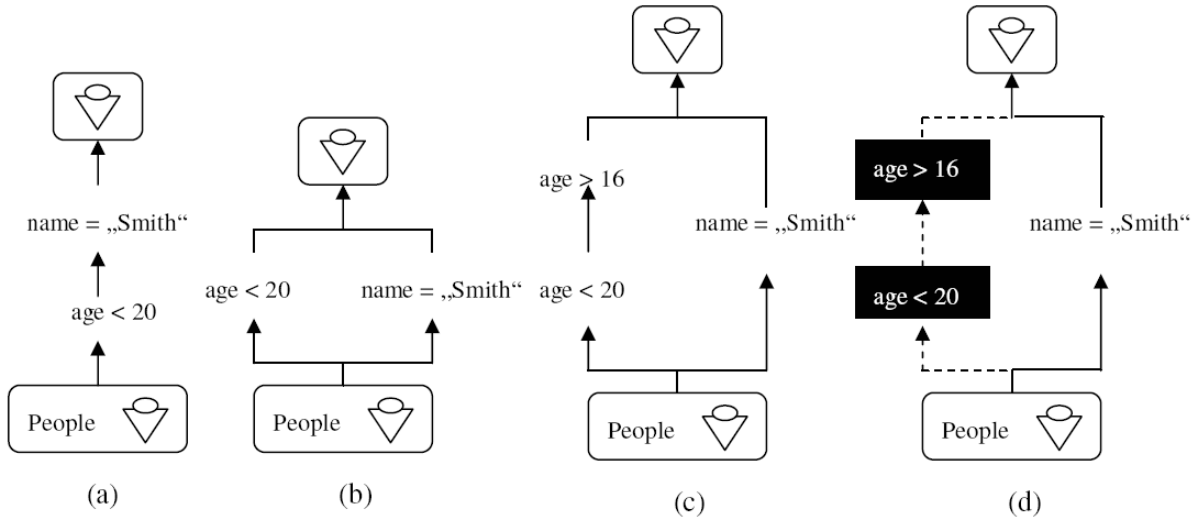


Abbildung 4.14: Visualisierung von AND, OR und NOT (aus [Murray2000]).

Darstellung (a) in Abbildung 6 zeigt die visuelle Repräsentation einer UND-Verknüpfung. Bedingungen werden in UND-Verknüpfungen durch Pfeile hintereinander geschaltet. Die dargestellte Verknüpfung liefert als Ergebnis alle Personen, die jünger als 20 Jahre alt sind und den Namen „Smith“ haben.

Darstellung (b) zeigt eine ODER-Verknüpfung. ODER-Verknüpfungen werden durch Verzweigen eines Pfeils visualisiert. Nach der Definition der Bedingungen werden die beiden Äste der Pfeile wieder zusammengeführt. Die dargestellte Verknüpfung liefert als Ergebnis alle Personen, die entweder jünger sind als 20 Jahre oder den Namen „Smith“ besitzen.

Darstellung (c) kombiniert UND- und ODER-Verknüpfungen. Es werden alle Personen ausgewählt, deren Alter entweder zwischen 16 und 20 Jahren liegt oder die „Smith“ heißen.

Darstellung (d) erweitert die Darstellung (c) um eine Negation (NOT) der UND-Verknüpfung. Negationen werden durch gestrichelte Linien und schwarz ausgefüllte Boxen dargestellt. Die dargestellte Verknüpfung liefert als Ergebnis alle Personen, deren Alter nicht zwischen 16 und 20 Jahren liegt oder deren Name „Smith“ ist.

INTERSECTION, UNION, EXCEPT

In Kaleidoquery werden die Mengenausdrücke (INTERSECTION, UNION, EXCEPT) mit Hilfe von Venn-Diagrammen repräsentiert. Der Vorteil dieser Art von Diagrammen liegt darin, dass sich mit ihnen alle Relationen zwischen den betrachteten Mengen übersichtlich darstellen lassen. Die Ergebnismenge der Ausdrücke ist jeweils grau

Venn-Diagramme

markiert. Der Ausdruck EXCEPT wird in Kaleidoquery als DIFFERENCE bezeichnet.

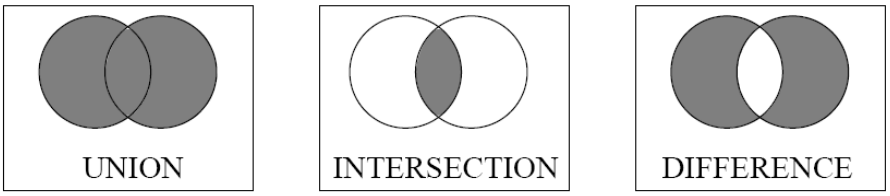


Abbildung 4.15: Mengenausdrücke, UNION, INTERSECTION, DIFFERENCE (aus [Murray2000]).

In der Abbildung 4.16 weiter unten ist der Durchschnitt von Leuten die bei IBM arbeiten mit denen die in London arbeiten gebildet — also alle Personen, die in London bei IBM arbeiten.

4.1.3.1.3. Zugriffspfade

Zugriffspfade

Zugriffspfade werden in Kaleidoquery durch Pfeile dargestellt. Über diese Pfeile kann von einer Klasse zu einer anderen Klasse navigiert werden.

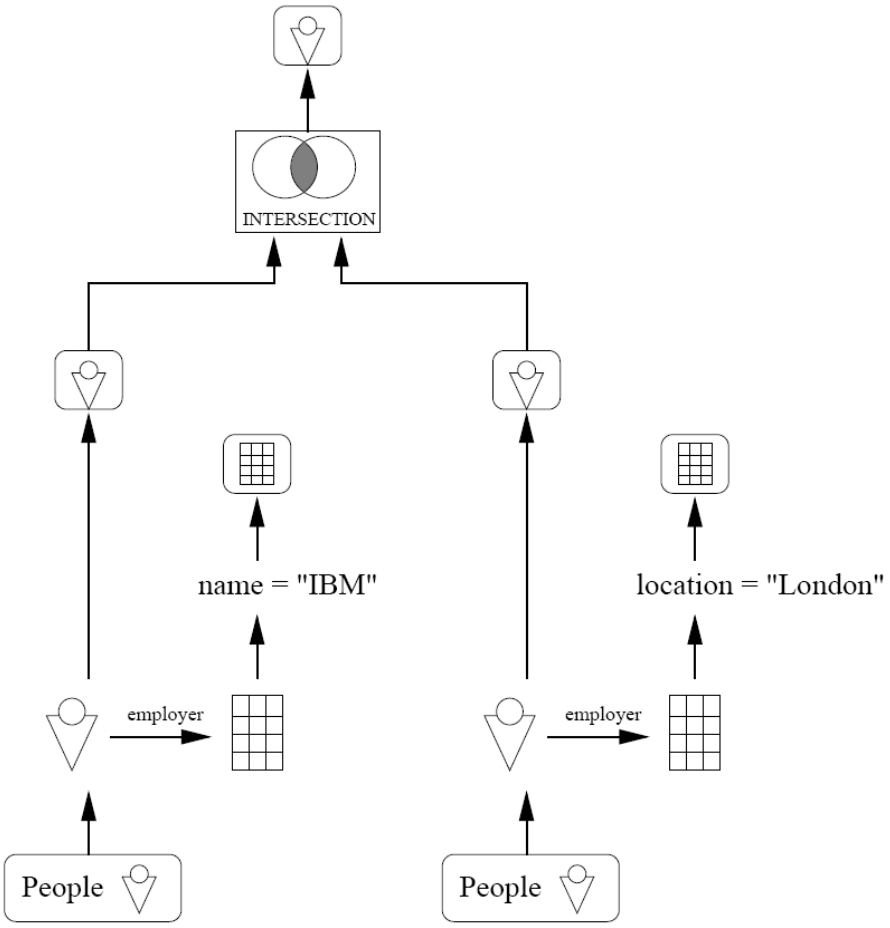


Abbildung 4.16: Ein Beispiel für eine Anwendung von Mengenausdrücken und Zugriffspfaden — die Bildung von der Durchschnittsmenge (aus [Murray2000]).

In Abbildung 4.16 wird eine Einschränkung bezüglich des Arbeitgebers erstellt, indem der Anwender von der Klasse People zu der Klasse Company navigiert und die entsprechenden Attribute — hier name = „IBM“, location = „London“ — setzt. Die Beziehung zwischen diesen beiden Klassen ist durch einen horizontalen Pfeil dargestellt, mit dem Namen der Beziehung über dem Pfeil.

4.1.3.1.4. FOR ALL, EXISTS, IN

FOR ALL, EXISTS und IN

Die Operatoren FOR ALL, EXISTS und IN werden in Kaleidoquery durch das Symbol dargestellt, wobei in der Schlaufe jeweils der Operator steht. In Kaleidoquery wird der Operator IN durch das Schlüsselwort MEMBER repräsentiert. Das Beispiel in Abbildung 4.18 auf Seite 90 visualisiert die Anfrage:

- IN: Welche Personen arbeiten in einem Unternehmen in England?
- FOR ALL: Welche Unternehmen beschäftigen nur Arbeitnehmer, die älter als 60 Jahre sind und ein Gehalt von min. 25000 haben?
- EXISTS Gibt es Unternehmen, die Arbeitnehmer älter als 60 Jahre und mit einem Gehalt von min. 25000, beschäftigen?

4.1.3.1.5. JOIN

JOIN und SELF-JOIN

In Kaleidoquery bilden bei einem Join von Attributen zweier Klassen die Extensionen beider Klassen die Startsymbole der visuellen Repräsentation. Im Gegensatz zur entsprechende OQL-Anfrage, wo nur auf eine Ergebnismenge, die sich auf eine der beiden Klassen bezieht projiziert wird, werden in Kaleidoquery die Ergebnismengen beider Klassen angezeigt. Der höchste Punkt der visuellen Repräsentation bildet dabei die Ergebnismenge der Anfrage.

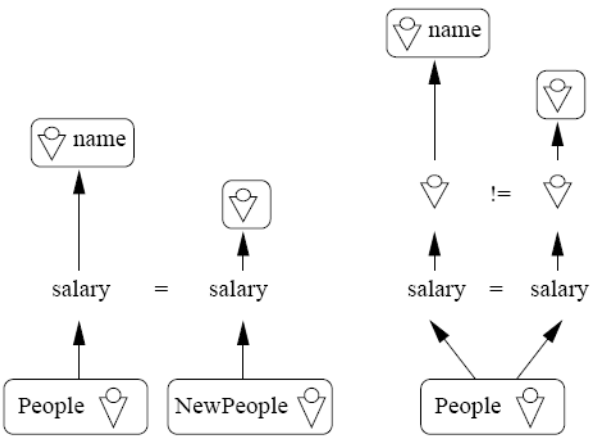


Abbildung 4.17: Ein JOIN und SELF JOIN (aus [Murray2000])

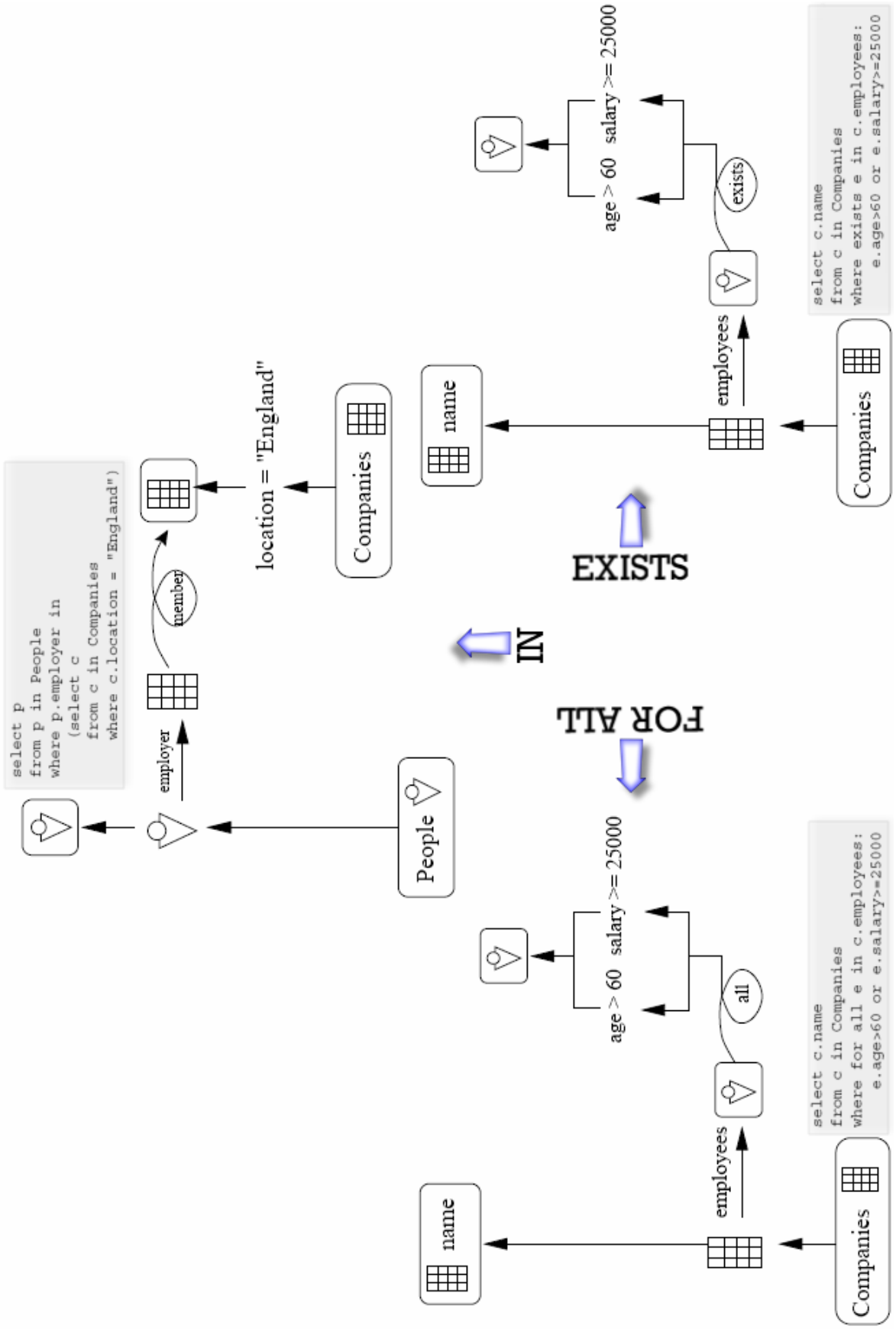


Abbildung 4.18: Die Operatoren FOR ALL, EXISTS und IN. Der OQL-Code ist bei der jeweiligen Grafik angeführt (adaptiert nach [Murray2000]).

4.1.3.1.6. GROUP BY, HAVING

GROUP BY und HAVING

Kaleidoquery unterstützt die Klauseln GROUP BY, zur Gruppierung von Ergebnissen, sowie HAVING, zur Filterung von gruppierten Ergebnissen. Durch eine Gruppierung wird das Ergebnis einer Anfrage in Partitionen (Untermengen) aufgeteilt. Nach der Gruppierung können durch die HAVING-Klausel Bedingungen an die Partitionen gestellt werden. Als Ergebnis werden, den definierten Bedingungen entsprechend gefilterte, Partitionen geliefert.

In Abbildung 4.19 werden Name, Alter und Gehalt von Person ausgewählt. Danach werden die Personen in zwei Gruppen partitioniert — eine Gruppe mit allen die „Smith“ heißen, und eine Gruppe mit Gehältern zwischen 30000 und 40000. Die beiden Partitionen werden abschließend durch die HAVING-Klausel gefiltert.

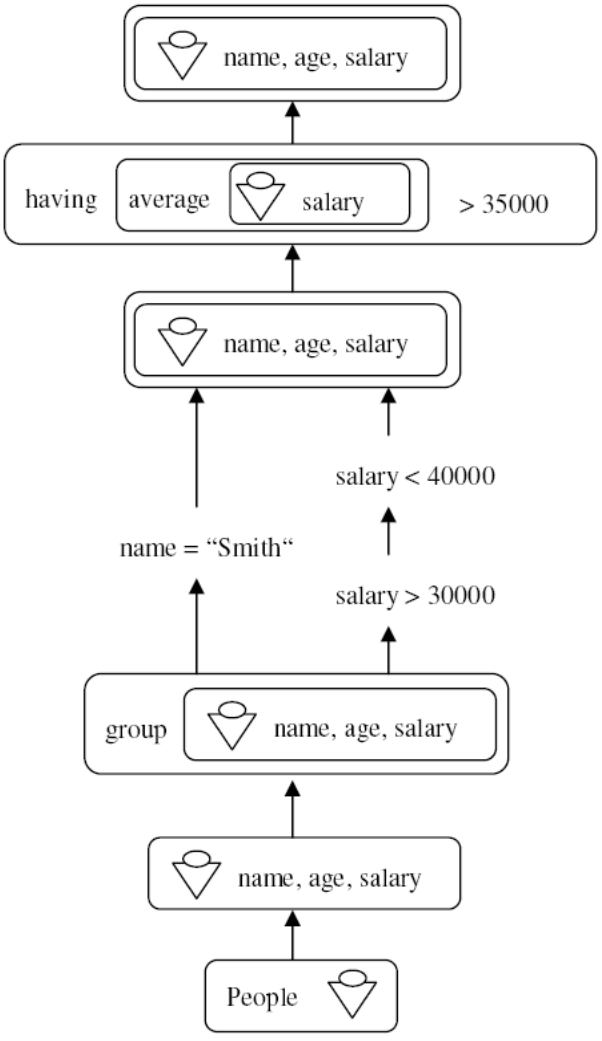


Abbildung 4.19: Gruppierung mit GROUP BY und HAVING (aus [Murray2000]).

4.1.3.1.7. Strukturieren und Ordnen

Strukturierungen der Anfrage und des Ergebnisses sind voneinander getrennt

Bei textuellen Anfragen hat die Strukturierung der Anfrage Einfluss auf die Strukturierung des Ergebnisses. In Kaleidoquery sind diese beiden Aufgaben unabhängig voneinander – zunächst wird die Anfrage modelliert, danach kann aus den verfügbaren Optionen entschieden werden wie das Ergebnis strukturiert werden soll. Die entsprechende Funktion in Kaleidoquery hat das Symbol  , wobei die Art der Strukturierung über dem Symbol vermerkt wird. In Kaleidoquery genügt es die Strukturierungsfunktion zu ändern um eine andere Strukturierung des Ergebnisses zu erhalten, die eigentliche Anfrage muss nicht geändert werden. In textuellen Anfragen sind immer Anpassungen an der Anfrage notwendig.

In Abbildung 4.20 (a) werden Paare von Leuten, mit gleichem Gehalt, generiert. In Abbildung 4.20 (b) dagegen werden Gruppen von Leuten mit gleichem Gehalt erzeugt. Für die Änderung der Sicht auf das Ergebnis genügt es die entsprechende Strukturierungsfunktion anzuwenden. Im Vergleich dazu sind die beiden OQL Anfragen sehr unterschiedlich.

```
(a) select tuple(pname:p.name, qname:q.name)
    from p in People, q in People
    where p.salary = q.salary

(b) select struct(p.name,
        (select struct (q.name)
        from q in People
        where p.salary = q.salary))
    from p in People
```

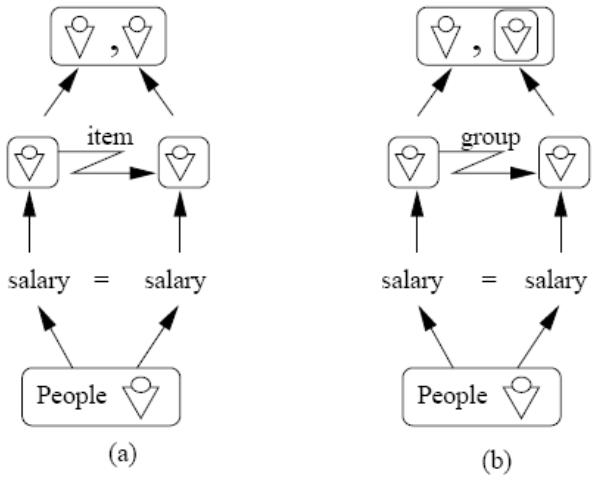


Abbildung 4.20:. Durch die voneinander unabhängige Strukturierung der Anfrage und des Ergebnisses, ist in Kaleidoquery, bei verschiedenen Strukturierungen des Ergebnisses, keine Änderung an der Anfrage notwendig (adaptiert nach [Murray2000]).

ORDER BY

In Kaleidoquery werden Sortierungen von Ergebnissen über ORDER BY realisiert. Dem Ergebnissymbol wird ein ORDER BY vorangestellt. Durch einen nach oben oder unten gerichteten Pfeil wird definiert, ob das Ergebnis aufsteigend oder absteigend sortiert werden soll. Abbildung 4.21 stellt ein Ergebnissymbol dar, in dem definiert ist, dass das Ergebnis aufsteigend nach dem Attribut name sortiert wird.

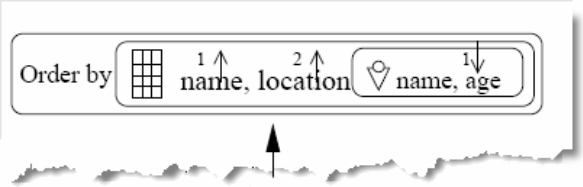


Abbildung 4.21: Sortierung erfolgt mit ORDER BY und Pfeilrichtung (adaptiert nach [Murray2000]).

4.1.3.2. Query By Example (QBE)

QBE – benutzerfreundliche interaktive Anfrageschnittstelle

Query by Example (QBE) ist eine visuelle Sprache zur Beschreibung von Datenbankabfragen in relationalen Datenbanken, die von Moshé M. Zloof bereits 1977 [Zloof1977] vorgestellt. QBE war als benutzerfreundliche interaktive Anfrageschnittstelle konzipiert. Ansatzweise findet sich QBE heute z.B. noch in der „Entwurfsansicht“ für Anfragen in MS Access.

In QBE wird die Abfrage nicht, wie in SQL, durch einen Text repräsentiert, sondern durch ein Tabellengerüst. Jedes Relationenschema der Datenbank lässt sich in einer Tabelle darstellen, deren Spalten den Attributen der Relation entsprechen.

QBE richtet sich an Gelegenheitsbenutzer

Query by Example richtet sich speziell an Gelegenheitsbenutzer, die mit SQL und anderen komplexen Abfragen wenig anfangen können. Das System stellt ein Tabellengerüst zur Verfügung. Die Tabellenpositionen können mit Werten, Ausdrücken oder Variablen belegt werden. Komplexere Bedingungen werden über eine zusätzliche Tabelle – die condition box – gebildet. Aggregatfunktionen können verwendet werden und Zeilen können negiert werden. Mit „_“ werden Beispielelemente begonnen, weiters dient es der Verknüpfung von Tabellen und der Angabe von Bedingungen in der condition box. Vergleichsoperationen werden in einer Zelle angegeben. Die Kommandos sind P., I. und D. und stehen für print, insert und delete. I. und D. werden immer in der ersten Spalte angegeben, P. kann in der ersten Spalte stehen, um die gesamte Zeile auszugeben, oder bei einem oder mehreren Attributen, um nur diese Attribute auszugeben. Durch ein Negationszeichen am Anfang einer Zeile können gültige Belegungen weiter eingeschränkt werden. Eine negierte Zeile darf kein P. enthalten.

Zur Veranschaulichung soll das Beispiel auf der nächsten Seite dienen. Die Abbildung 4.22 zeigt dabei an ein paar exemplarischen Anfragen, die grundlegende Vorgansweise in QBE bei Datenbankanfragen an:

Beispiel

In der Datenbank seien diese Relationen vorhanden:

Patient(patnr, pname, padresse, ort)
Aufenthalt (aufentnr, patnr, diagnart, dauer)
Diagnose(diagnart, dname, dprog)

Patient mit Patientennummer (patnr), Name (pname), Adresse (padresse) und Wohnort (ort). Stationärer Aufenthalt mit Aufenthaltsnummer (aufentnr), Patientennummer (patnr), Art der Diagnose nach ICD-10 (diagnart) und Dauer des Aufenthaltes in Tagen (dauer). Diagnose mit Art der Diagnose nach ICD-10 (diagnart), Bezeichnung der Diagnose (dname) und Prognose für den Verlauf (dprog).

Anfragen in QBE

In Query By Exampe werden Anfragen durch Befüllen der Tabellengerüste gestellt. In Abbildung 4.22 sind einige einfache Fälle dargestellt. In [Zloof1977] findet sich eine ausführliche Beschreibung der Sprache mit vielen weiteren Beispielen.

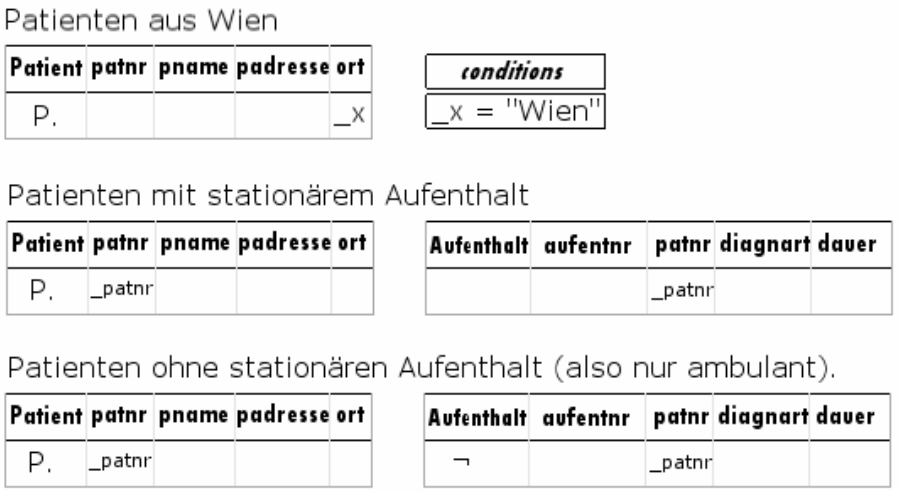


Abbildung 4.22: Diese Beispiele zeigen wie Projektion (Ausblenden von Spalten) und Selektion (Ausblenden von Zeilen), hier mit der condition box, sowie Verknüpfungen umgesetzt werden

Query By Example ist, trotz des hohen Alters, für diese Arbeit interessant, weil sie mit Tabellen arbeitet, für Gelegenheitsbenutzer konzipiert wurde und trotz der Einfachheit des Konzeptes mächtig Anfragen erlaubt.

4.1.4. RPDR Query Tool

Anfragetool für klinischen Datenbanken

RPDR Query Tool ist ein Akronym für Research Patient Data Registry Query Tool. Es ist ein Anfragetool für die klinischen Datenbanken der Partners Healthcare Inc., mit dem, forschende Mediziner ihre gewünschten Patientenkollektive

herausfiltern können. Das RPDR Query Tool ist seit 1999 am Partners Healthcare Inc.[34] in Boston, USA im Einsatz.

Unterstützung von datenbankunerfahrenen Medizinern bei Anfragen an die Datenbank

Abweichend von den weiter oben beschriebenen Anfragesprachen spezialisiert sich das RPDR Query Tool nur auf die Funktionalität, die für eine medizinisch–wissenschaftliche Anfrage relevant ist. Das besondere Augenmerk liegt in der Unterstützung von datenbankunerfahrenen Medizinern bei der Formulierung ihrer Anfragen an die Datenbank.

Alle weiteren Ausführungen zum RPDR Query Tool basieren auf den Arbeiten [Murphy2000] und [Murphy2003].

Das RPDR Query Tool ermöglicht dem forschenden Mediziner Anfragen ohne Expertenhilfe an die RPDR[35] zu stellen, dazu versucht es zwei grundlegende Probleme zu lösen:

- Blättern in Metadaten: Anwender werden nur mit vertrauten Begriffen konfrontiert.
- Bilden von logischen Konstrukten: durch Visualisierung von Zusammenhängen, wird der fehlerhaften Bildung von logischen Konstrukten entgegnet.

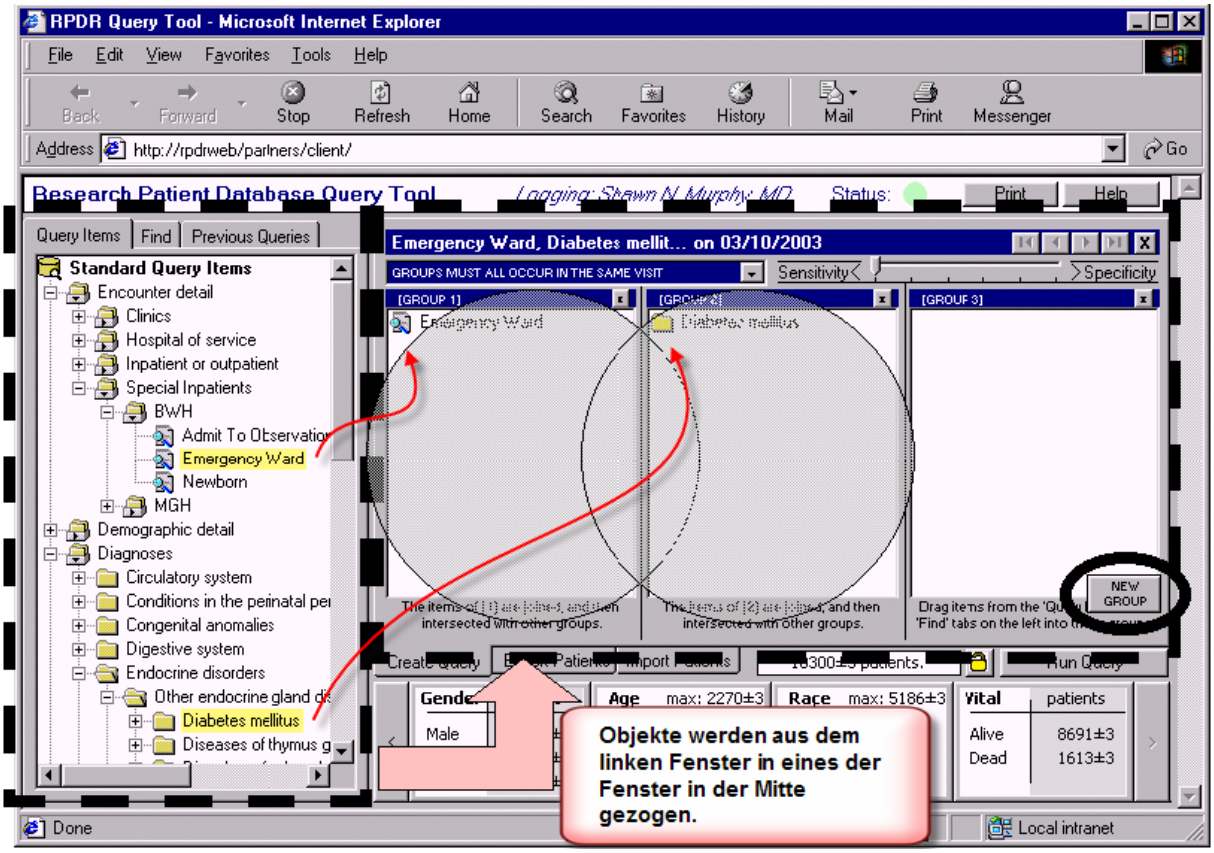


Abbildung 4.23: RPDR Query Tool Benutzeroberfläche (adaptiert nach [Murphy2003]).

[34] Partners Healthcare Inc. wurde 1994 durch das Brigham and Women's Hospital und Massachusetts General Hospital gegründet. http://www.partners.org/

[35] RPDR (Research Patient Data Registry) ist die analytische Datenbank der Partners Healthcare Inc.

Venn–Diagramme helfen bei der Visualisierung von logischen Zusammenhängen

In Abbildung 4.23 ist das Modell der Benutzeroberfläche von RPDR Query Tool dargestellt. Im linken Fenster kann der Anwender durch die Metadaten blättern. Die Präsentation ist so gewählt, dass der Mediziner nur ihm vertraute Begriffe sieht, ganz unabhängig von der darunter liegenden Organisation der Daten in der Datenbank. In der Mitte werden durch die Fenster, symbolisch, Venn–Diagramme modelliert. Der Anwender zieht ein Objekt vom linken Fenster in eines der Fenster in der Mitte. Wenn zum gewählten Objekt Werte definiert werden müssen, geht automatisch ein Fenster zum Setzen der Werte auf (siehe Abbildung 4.27). Objekte die in das gleiche Fenster abgelegt werden sind OR verknüpft, Objekte die in verschiedenen Fenstern abgelegt werden sind AND verknüpft. Die Invertierung (NOT) einer Gruppe, wird über das Auswahlmenü und „Invert All Items“ erreicht (siehe Abbildung 4.27). Durch den „New Group“ Knopf werden bis zu 25 neue Fenster geöffnet. Durch den „Run Query“ Knopf wird die Anfrage gestartet. Am Ende der Anfrage werden die Ergebnisse im unteren Fenster visualisiert.

Visualisierung von Ergebnissen

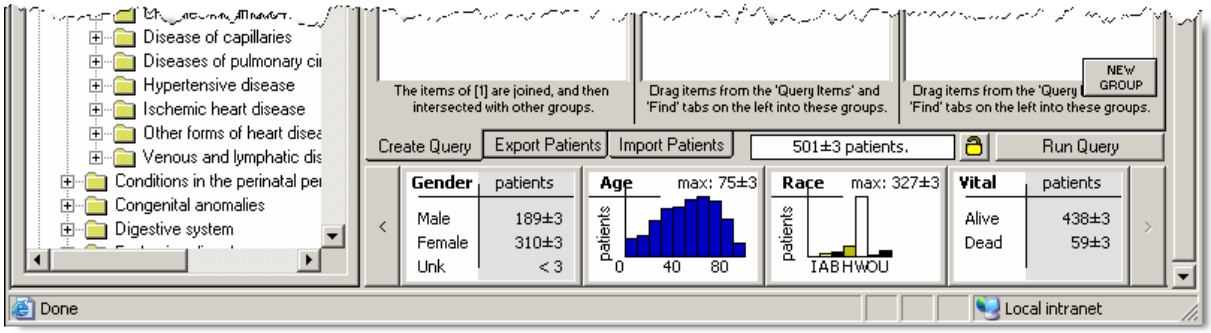


Abbildung 4.24: Visualisierung des Anfrageergebnisses (adaptiert nach [Murphy2003]).

Zeitliche Einschränkungen

Neben der üblichen zeitlichen Einschränkung einzelner Elemente mit Beginn– und Endzeit, kann für die gesamte Anfrage eine zeitliche Abstimmung nach Arztbesuch vorgenommen werden. Der Anwender kann wählen ob alle Gruppen, das sind die einzelnen verknüpften Fenster, im gleichen Arztbesuch oder in Beliebigen vorkommen. Ein Beispiel kann das verdeutlichen. Wenn jemand „Diabetes“ und „Notfallraum“ in verschiedene Fenster zieht und definiert, dass sie im gleichen Arztbesuch vorkommen sollen, dann muss der Patient wegen Diabetes im Notfallraum gewesen sein. Wenn definiert wird, dass sie in beliebigen Arztbesuchen vorkommen sollen, dann werden alle Patienten, die wegen beliebiger Gründe im Notfallraum waren und Diabetes haben aus der Datenbank gewählt.

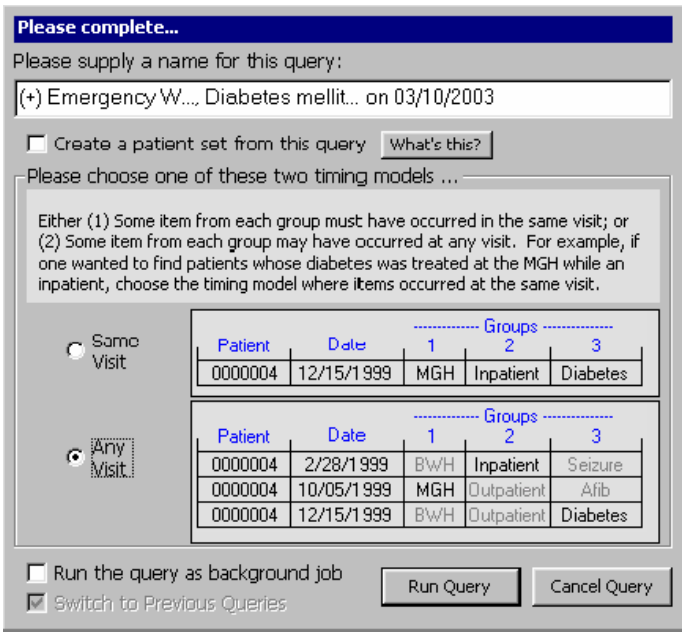


Abbildung 4.25: Fenster zur Verringerung von zeitlichen Abstimmungsfehlern. Der Anwender kann überprüfen ob seine Wahl – Same Visit oder Any Visit – richtig ist und bei Bedarf korrigieren (aus [Murphy2003]).

Datensicherheit

Das RPDR Query Tool bietet, aus Gründen der Datensicherheit, zwei unterschiedliche Sichten auf die Patientendaten an:

1. Ein Patientenkollektiv mit den gewünschten Eigenschaften, allerdings ohne einzelne Patienten identifizieren zu können.
2. Über die RPDR data acquisition engine können Details zu den ausgewählten Patienten abgerufen werden.

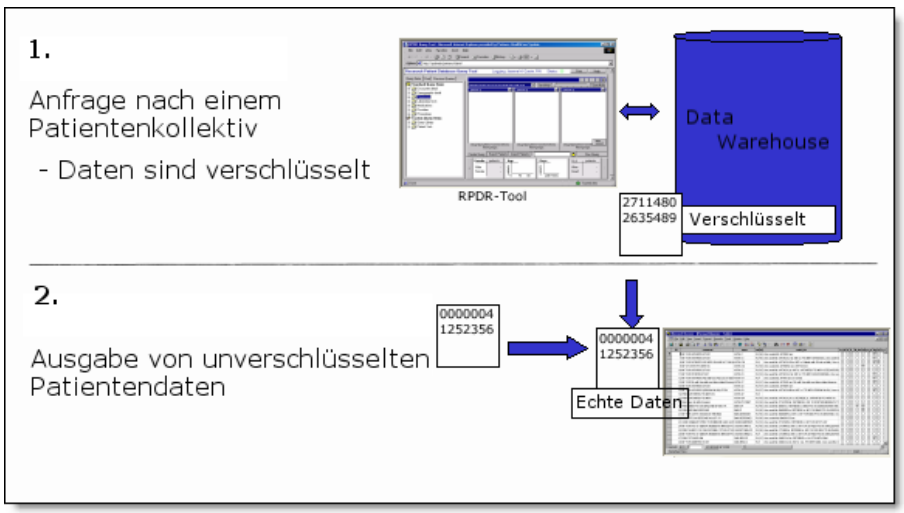


Abbildung 4.26: RPDR Query Tool – Patientendatenverarbeitung

Bei Anfragen an die Datenbank können die medizinischen Daten demographische Informationen, Labortestresultate, Entlassungsberichte, Operationsberichte und radiologische und pathologische Resultate enthalten. Die Anfrageresultate werden in einem textuellen Bericht und/oder einer Access–Datenbank abgelegt.

Alte Anfragen können wieder verwendet werden

Formulierte Anfragen können abgespeichert werden. Bei späteren Anfragen können sie als „Custom Queries“ direkt als Objekt ausgewählt und ins Anfragefenster gezogen werden.

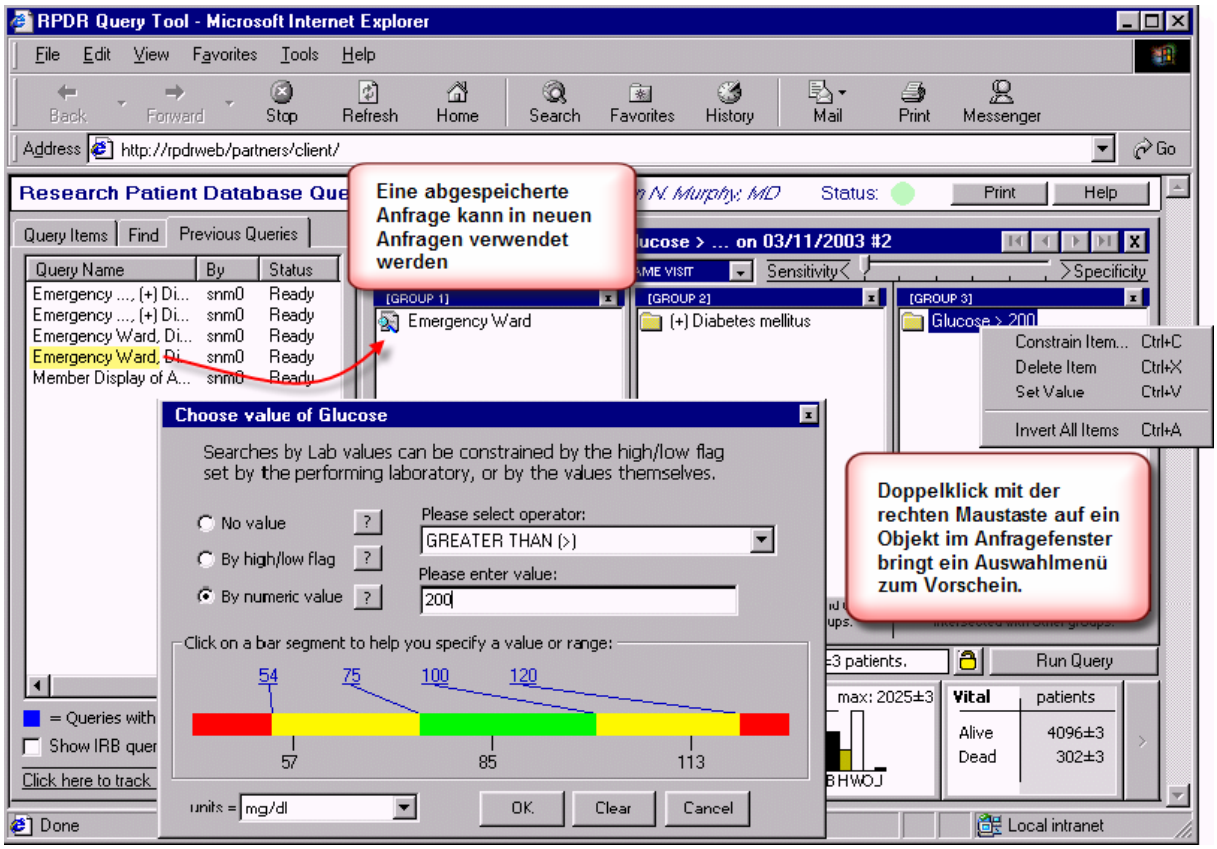


Abbildung 4.27: Der „Previous Queries“–Reiter ist ausgewählt und eine frühere Anfrage wird in der neuen Anfrage verwendet. Es ist auch das Optionenmenü aufgeklappt, sowie im Vordergrund ein Fenster zum Setzen der Werte für das ausgewählte Objekt (adaptiert nach [Murphy2003]).

4.2. Zusammenfassung

Nach der Betrachtung der visuellen Sprachen muss festegestellt werden, dass jede Sprache ihre speziellen Vorteile hat und nur eine Kombination der Eigenschaften verschiedener visueller Sprachen die Anforderungen an eine umfassende visuelle Anfragesprache erfüllen kann.

Diagrammbasierte visuelle Sprachen, wie Kaleidoquery, können einen guten Überblick über die Daten bieten. Die Verwendung aussagekräftiger grafischer Symbole, wie in

LabVIEW, in Verbindung mit einem Flussdiagramm, erhöht bei unerfahrenen Benutzern die Akzeptanz der Sprache, bietet eine intuitive Bedienung und einen guten Überblick über die gestellte Anfrage. Tabellenbasierte Sprachen, wie QBE, eignen sich für Anfragen, die vereinheitlicht werden können, die sich also nur in bestimmten Teilen verändern. Browser, wie zum Teil im RPDR Query Tool realisiert, erleichtern Anfragen, bei denen die gesuchten Daten nicht genau einzeln aufgeführt werden können. Die Verwendung von Metamodellen ist eine gute Grundlage für den Entwurf von Modellierungswerkzeugen, da sie einfach in Klassendiagramme übergeführt werden können.

Um die ideale Anfragesprache für eine Anwendung zu formulieren, müssen die anwendungsspezifischen Anforderungen ausgearbeitet werden und daraus die optimalen Methoden, die die Sprache unterstützen sollte extrahiert werden.

Aus dieser Betrachtung der verschiedenen visuellen Sprachen, werden, im nächsten Kapitel, Überlegungen für die Anforderungen an VISAmed angestellt, die später die Entwicklung visueller Konstrukte von VISAmed vorgeben werden.

KAPITEL

5

Anforderungen an VISAmed

Alles Sichtbare ist nur die äußere Gestalt,
in die sich ein Unsichtbares hüllt...

Gertrud von Le Fort

WIE bereits im Kapitel 4 dargestellt, besteht das Ziel dieser Arbeit in der Entwicklung eines Anfragegenerators, der mit seiner Benutzerschnittstelle und der durch sie repräsentierte Anfragesprache auch den unerfahrenen Anwendern erlaubt selbstständig sinnvolle Datenbankanfragen über multidimensionale Daten zu stellen. Weiters soll ein flexibles Interaktionsmodell unterstützt werden, das dem Anwender große Freiheiten bei der Datenerforschung erlaubt. Als Akronym für die Benutzerschnittstele soll VISAmed (**V**isual **I**nterface **S**upporting **A**d hoc queries in **med**icine) stehen.

5.1. Ableitung von Anforderungen an VISAmed

Medizinische Daten stellen zusätzliche Anforderungen an Datenbanken und Anfragesprachen.

Medizinische Daten haben im Vergleich zu Daten, wie sie in traditionellen Datenbanken vorkommen, zusätzliche Eigenschaften. Die Abbildung von räumlichen und zeitlichen Beziehungen in Anfragen, die geeignete Präsentation der Anfrageergebnisse und die Möglichkeit unscharfe Anfragen zu formulieren sind nur einige dieser Anforderungen. Eine ausführliche Betrachtung der Besonderheiten und der daraus abgeleiteten speziellen Kriterien für VISAmed erfolgt in Abschnitt 5.1.2.

5.1.1. Allgemeine Anforderungen an VISAmed

Allgemeine Anforderungen an VISAmed

Nachfolgend werden die wesentlichen allgemeinen Anforderungen an Anfragesprachen aufgeführt, wie sie in [Heuer1991] entwickelt und nach einer Verfeinerung in [Saake1997] von [Paskamp1999] kritisch untersucht wurden. Die, für diese Arbeit nützlichen, Ergebnisse dieser Untersuchung werden in Hinblick auf die Anforderungen von VISAmed diskutiert. Da die folgenden allgemeinen Anforderungen an eine Anfragesprache zum Teil gegensätzliche Ziele verfolgen, ist zu bedenken, dass eine Anfragesprache nicht alle Kriterien gleichermaßen gut erfüllen kann ([Saake1997], S. 236).

Die folgenden Abschnitte erläutern zunächst das jeweilige Kriterium allgemein und bewerten es anschließend bezüglich der Ansprüche von VISAmed.

5.1.1.1. Ad–hoc Formulierung

Mit einer Ad-hoc Anfrage, können ungeplante assoziative Fragen formuliert werden.

Neben dem Zugriff auf die Datenbank durch Anwendungsprogramme muss es auch eine Möglichkeit geben, Anfragen direkt über eine interaktive Benutzerschnittstelle zu formulieren. Mit einer Ad-hoc Anfrage, können in deklarativer Weise ungeplante assoziative Fragen formuliert werden, ohne ein vollständiges Programm schreiben zu müssen.

Eine Ad-hoc Formulierung sollte auch von VISAmed unterstützt werden. Es muss geeignete Methoden anbieten, wie etwa eine graphische Anfrageformulierung für zeitliche Zusammenhänge. Idealerweise sollten auch Basisfunktionalitäten zur Spezifikation der Ergebnispräsentation vorhanden sein.

5.1.1.2. Generische Operationen

Für semantisch gleiche Operationen, kann, auf verschiedenen Datenbankobjekten, die gleiche Syntax verwenden werden.

Durch generische Operatoren kann für semantisch gleiche Operationen auf verschiedenen Datenbankobjekten, die gleiche Syntax verwendet werden. Der Anwender wird dadurch entlastet, er muss nur wenige Sprachkonstrukte erlernen. Zusätzlich unterstützen die generischen Operatoren die Datenunabhängigkeit durch Abstraktion von den Implementierungsdetails und der Semantik der Anwendung.

In VISAmed ist es, z. Bsp. sinnvoll für die Beschreibung der zeitlichen und räumlichen Beziehungen zwischen Objekten innerhalb einer Anfrage, wie auch für die Beschreibung der Präsentation der Anfrageergebnisse einheitliche Attribute zu verwenden.

5.1.1.3. Anwendungsunabhängigkeit

Universelle Verwendung der Anfragesprache

Eine Anfragesprache soll für eine universelle – und keine spezielle – Verwendung entworfen werden. Diese Forderung trennt das DBMS von den Anwendungen. Es lassen sich

damit neue Anwendungen hinzufügen bzw. alte Anwendungen entfernen, ohne gleichzeitige Beeinflussung des DBMS. Die Anwendungsunabhängigkeit ist eine wichtige Forderung des Prinzips der Datenunabhängigkeit, das für eine langfristige Benutzung der Daten, durch verschiedene Anwendungen, von großer Bedeutung ist. Die Datenunabhängigkeit wird durch das 3–Ebenenmodell (S. 52, Abschnitt 3.1.3) und seine Unterteilung in interne Ebene, konzeptionelle Ebene und externe Ebene erreicht.

Für VISAmed kann die Anforderung nach Anwendungsunabhängigkeit übernommen werden. Obwohl der Einsatz primär für klinische Datenbanken gedacht ist, kann durch entsprechende Anpassung das Einsatzgebiet auf beliebige Datenbanken erweitert werden.

5.1.1.4. Deskriptivität

Der Benutzer formuliert was er haben will und nicht wie er das bekommt, was er haben will.

Der Zugriff auf Datenbankobjekte erfolgt durch die Beschreibung der Struktur und Inhalt der gesuchten Objekte. Es werden keine Berechnungsvorschriften und keine Abarbeitungsreihenfolge vorgegeben. Der Benutzer formuliert was er haben will und nicht wie er das bekommt, was er haben will. Diese Eigenschaft wird als Deskriptivität bezeichnet. Sie sichert die Datenunabhängigkeit und ist eine Vorraussetzung für die Optimierung.

Deskriptivität ist bei VISAmed im Zusammenhang mit der Präsentation von Anfrageergebnissen von besonderer Bedeutung. Die Konstrukte zur Präsentationsbeschreibung sollten nur den Inhalt und die Struktur der Präsentation beschreiben und nicht die Art wie diese zu erstellen ist. Weitere Aspekte der Deskriptivität, wie Optimierungen oder Änderungsunabhängigkeit der Implementierung werden in dieser Arbeit nicht weiter betrachtet.

5.1.1.5. Mengenorientiertheit

Klassische Anfragen arbeiten auf Mengen von Daten.

Klassische Anfragen arbeiten auf Mengen von Daten. Dabei arbeitet jede Operation auf Mengen von Daten gleichzeitig und nicht navigierend nur auf einzelne Elemente. Das System liefert das Ergebnis in einer beliebigen Reihenfolge. Eine Optimierung der Anfragebearbeitung kann hier zu einer höheren Effizienz im Anfrageprozess führen.

Die Mengenorientiertheit ist auch für VISAmed wünschenswert, jedoch müssen einige Besonderheiten beachtet werden. In einer medizinischen Datenbank ist es sinnvoll für die Suche nach komplexen Daten Ähnlichkeitsanfragen zuzulassen. Es werden also nicht nur exakte sondern auch unscharfe Anfragen formuliert. Dadurch liegen statt klassischen Mengen Fuzzy– Mengen vor. Die Ergebnismenge besteht da–

bei aus Daten, die ähnliche Eigenschaften aufweisen, wie sie in der Anfrage formuliert wurden. Als Maß für die Ähnlichkeit zwischen der Anfrage und den zurück gelieferten Daten wird eine Gewichtung, auch Ranking bezeichnet angeben. Die einzelnen Mengenoperationen, wie Vereinigung, Durchschnitt, usw. müssen um Konzepte zur Behandlung von Relevanzwerten erweitert werden. Die Präsentation der Ergebnisse dieser unscharfen Anfragen wird entsprechend dem Ranking angeordnet.

Die Behandlung der Fuzzy–Mengen ist nicht Bestandteil dieser Arbeit, stellt aber eine mögliche sinnvolle Erweiterung von VISAmed dar.

5.1.1.6. Orthogonalität

Sprachkonstrukte sind in ähnlichen Situationen auch ähnlich anwendbar.

Die Operationen müssen frei und beliebig kombinierbar sein. Das bedeutet, dass die Sprachkonstrukte in ähnlichen Situationen auch ähnlich anwendbar sind. Die Orthogonalität ermöglicht das Schachteln von Anfragen und ist damit Vorraussetzung für komplexe Anfragen.

VISAmed soll die Kombination von Konstrukten erlauben.

5.1.1.7. Effizienz

Kurze Antwortzeiten erhöhen die Benutzerfreundlichkeit.

Kurze Antwortzeiten sind für die Benutzerfreundlichkeit wesentlich. Der akzeptable Aufwand liegt bei $O(n^2)$, n Anzahl der betrachteten Tupel. Daher sollen die Operationen effizient realisierbar sein.

Im Allgemeinen muss das auch für VISAmed gelten, jedoch können durch die hohen Datenvolumina medizinischer Datenbanken und in die eigentliche Anfragesprache der Datenbank, die hinter VISAmed liegt, Effizienzprobleme auftreten. Bezüglich der Ergebnispräsentation kann allerdings, durch zweckmäßige Parametrisierung der Ausgabe, die Effizienz des Systems optimiert werden.

5.1.1.8. Erweiterbarkeit

Erweiterung um anwendungsspezifische Datentypen und Methoden.

Datenbankmanagementsysteme folgen dem Prinzip der Erweiterbarkeit und lassen sich um anwendungsspezifische Datentypen und Methoden erweitern.

Die einfache Erweiterbarkeit um Funktionen ist eine wichtige Forderung bei VISAmed. Es soll eine beliebige Erweiterbarkeit des bestehenden Systems möglich sein, ohne grundlegende Änderungen an ihm vornehmen zu müssen.

5.1.1.9. Abgeschlossenheit

Das Ergebnis ist wieder eine Relation.

Die Abgeschlossenheit fordert, dass jedes Ergebnis einer Anfrage in dem zugrunde liegenden Datenmodell darstellbar ist. Das Ergebnis ist wieder eine Relation und kann wieder als Eingabe für eine neue Anfrage genutzt werden.

Für VISAmed kann diese Forderung übernommen werden. Erforderlich dabei ist, dass die ermittelten Ergebnisrelationen wieder Eingangsrelationen für künftige Anfragen darstellen können.

5.1.1.10. Adäquatheit

Anfragesprache nutzt alle Modellkonstrukte des Datenbankmodells.

Die Adäquatheit verlangt, dass alle Modellkonstrukte des Datenbankmodells durch die Anfragesprache ausgenutzt werden können.

VISAmed soll dadurch die zeitlichen und räumlichen Beziehungen der medizinischen Daten sowie die Metadaten zur Formulierung von Anfragen nutzen. Außerdem sollen Konstrukte zur Präsentation der Anfrageergebnisse vorhanden sein.

5.1.1.11. Sicherheit

Keine Endlosschleifen und immer endliche Ergebnisse.

Keine Anfrage, die syntaktisch korrekt ist, darf in eine Endlosschleife geraten oder ein unendliches Ergebnis liefern. Die Anfragesprache darf daher nicht berechnungsvollständig sein, weil sonst z. Bsp. Endlosschleifen formuliert werden könnten.

VISAmed soll diese Forderung unterstützen.

5.1.1.12. Optimierbarkeit

Optimierbarkeit

Die Sprache besteht aus wenigen Operationen, für die es Optimierungsregeln gibt.

Optimierung von VISAmed wird in dieser Arbeit nicht behandelt.

5.1.1.13. Eingeschränktheit

Die Anfragesprache ist keine vollständige Programmiersprache.

Aus der Sicherheit, Optimierbarkeit und Effizienz folgt die Eingeschränktheit und besagt, dass die Anfragesprache keine vollständige Programmiersprache sein darf.

Die Eingeschränktheit gilt auch für VISAmed.

5.1.1.14. Vollständigkeit

Vollständigkeit

Die Anfragesprache muss mindestens die Anfragen einer Standardsprache ausdrücken können. Damit wird von relationalen Anfragesprachen z. Bsp. die gleiche Ausdrucksfähigkeit, wie sie relationale Algebra aufweist, verlangt.

Aufgrund des zugrunde liegenden Datenmodells, sollte VISAmed mindestens die Mächtigkeit der relationalen Algebra besitzen.

5.1.2. Spezielle Anforderungen an VISAmed

Besondere Eigenschaften von medizinischen Daten erfordern zusätzliche Eigenschaften der Anfragesprache.

Für den speziellen Bereich der medizinischen Auswertungen, ergeben sich, aufgrund der besonderen Eigenschaften von medizinischen Daten, zusätzliche Anforderungen die über den traditionellen Ansatz hinausgehen. Neben exakten Anfragen auf numerischen und alphanumerischen Daten müssen weitere Konzepte berücksichtigt werden. In Anlehnung an Anforderungen an Multimedia–Anfragesprachen wie in ([Paskamp1999], S. 46ff) und ([Schulz2004a], S. 38f) beschrieben, die recht gut auch die Anforderungen an Anfragesprachen für medizinische Auswertungen abdecken und ergänzt um weitere Aspekte, sind das:

- Inhaltsbezogene Anfragen: Inhaltsbezogene bzw. inhaltsorientierte Anfragen suchen nach Gruppen inhaltlich ähnlicher Daten.
- Raumbezogene Anfragen: Anfragen an die räumlichen Beziehungen von Daten. Die Darstellung geographischer Daten spielt bei epidemiologischen Daten eine wichtige Rolle.
- Zeitbezogene Anfragen: Anfragen und Suche über zeitbezogene Zusammenhänge innerhalb oder zwischen den betrachteten Daten. Dabei können sowohl Zeitpunkte wie auch Zeitintervalle angegeben werden.
- Präsentation von Anfrageergebnissen: Ergebnisse werden entsprechend den vom Nutzer gegebenen Vorgaben aufbereitet und Ausgegeben.
- Natürliche Metapher: Es sollen Operatoren verwendet werden die für den Anwender eine natürliche Bedeutung und Verständlichkeit haben. Solche Operatoren sind etwa Selektion, Projektion, Navigation und Vereinigung.
- Dynamische Anfragegenerierung: Diese Technik erlaubt oft eine schnellere und einfachere Abfrage der Datenbank. Sie stellt ein wichtiges Interaktionsinstrument dar, da Dynamische Anfragegenerierung zu einer unmittelbaren Aktualisierung der dargestellten Daten führt.
- Patientendatenschutz: Echte Patientendaten müssen, bei Auswertungen, anonymisiert werden um Missbrauch vorzubeugen.

Diese Anforderungen werden in folgenden Abschnitten in Form von weiteren Kriterien beleuchtet.

Eine Auflistung von Anforderungen an ein medizinisches Auswertesystem findet sich auch in [Dorda2002], auf Seite 95f. Diese Auflistung deckt sich zum Teil mit dem hier bereits Gesagtem und ist auch für VISAmed gültig,

Nicht betrachtete Anforderungen …

Weitere Anforderungen, die jedoch in dieser Arbeit nicht betrachtet werden, sind:

- Unscharfe Anfragen: Unscharfe Formulierung einer Anfrage unter Verwendung von Methoden des Information Retrieval. Dabei können für die Formulierung der Anfragen auch Fuzzy–Prädikate genutzt werden. Dazu müssen die Mengenoperationen um Konzepte zur Behandlung von Fuzzy–Mengen erweitert werden.
- Gewichtete Anfragen und Präferenzanfragen: Bei der Anfrageformulierung können durch Gewichtung von Bedingungen oder Operatoren zusätzliche Präferenzen des Benutzers abgebildet werden. Analoges gilt für die Formulierung von Präferenzanfragen.

5.1.2.1. Inhaltsbezogene Anfragen

Inhaltsbezogene Anfragen

Bei inhaltsbezogenen Anfragen erfolgt die Suche auf Basis der inhaltsabhängigen und inhaltsunabhängigen Metadaten[36]. Diese Anfragen können auch unscharf, beispielsweise unter Verwendung eines Ähnlichkeitsoperators, oder unter Verwendung von Fuzzy–Ausdrücken wie *viel* oder *gering*, formuliert werden.

Welche Art der Suche, bei inhaltsbezogenen Anfragen, gemeint ist, soll durch folgendes Beispiel gezeigt werden:

> *Suche nach Patienten die in den letzten zehn Jahren an AIDS erkrankt sind und noch leben und bei denen keine wesentliche Verschlechterung des Gesundheitszustandes eingetreten ist.*

Das Beispiel zeigt, dass die weiter unten diskutierten zeitbezogenen und raumbezogenen Anfragen genau genommen auch inhaltsorientierte Anfragen sind, da Zeit und Raum ebenfalls Eigenschaften darstellen, die medizinische Daten beschreiben.

Die Unterstützung von inhaltsbezogenen Anfragen ist eine wichtige Forderung an VISAmed. Die Anfragen müssen auch zu komplexen Anfragen zusammengesetzt werden können.

[36] Inhaltsabhängige Metadaten beschreiben die Datensätze, geben Auskunft über Datentypen, usw., im Gegensatz zu inhaltsunabhängigen Metadaten, welche die Daten identifizieren.

5.1.2.2. Raumbezogene Anfragen

Raumbezogene Anfragen

In epidemiologischen Studien spielen Anfragen bei denen räumliche Trends beobachtet werden eine wichtige Rolle. Wietek beschäftigt sich in ([Wietek2000], S. 5ff) ausführlich mit dem Anwendungsgebiet der Epidemiologie. Näheres kann dort nachgelesen werden. Für diese Arbeit sei nur so viel festgehalten – Epidemiologie verzichtet meist auf ein experimentelles Vorgehen und beschränkt sich auf die Beobachtung der Verteilung von Krankheitshäufigkeiten in menschlichen Populationen, unter Zusammenarbeit mit anderen Fachgebieten, wie Sozialwissenschaften, Statistik, Biometrie oder Geographie. Dabei sollen bevölkerungsbezogen Inzidenz, Prävalenz und Mortalität nach zeitlichen und insbesondere *räumlichen* Trends beobachtet werden.

VISAmed soll den Benutzer bei der Bildung von interessierenden Patientenkollektiven unterstützen. Somit soll es auch zur Bereitstellung von Daten für epidemiologische Studien (Fall-Kontroll-Studien, Kohorten-Studien) herangezogen werden können.

5.1.2.3. Zeitbezogene Anfragen

Zeitbezogene Anfragen

Im medizinischen Umfeld spielt Zeit eine kritische Rolle. So stützen sich Entscheidungen für medizinische Behandlungen auf der entsprechenden Beachtung der individuellen Krankengeschichte. In der Medizin werden Daten von Individuen meist über eine lange Zeit gesammelt, um anschließend analysiert und interpretiert zu werden. Bei zeitbezogenen Anfragen können sowohl Zeitpunkte, Zeitintervalle wie auch Zeitspannen betrachten werden.

Ein Zeitintervall ist ein Intervall im mathematischen Sinne und Teilmenge der Menge Zeit. Es ist durch einen Anfangszeitpunkt und einen Endzeitpunkt bestimmt. Ein Zeitpunkt ist ein Sonderfall eines Zeitintervalls bei dem die Intervalluntergrenze gleich der Intervallobergrenze ist. Der Abstand zwischen der Ober– und der Untergrenze eines Zeitintervalls ist eine Zeitspanne. ([Paskamp1999], S. 31)

Zwischen zwei Zeitintervallen gibt es nach [Allen1983] dreizehn verschiedene Relationen. Diese Beziehungen können als Grundlage für die Modellierung von zeitlichen Anfragen in VISAmed verwendet werden. Dazu werden zeitbezogene Operationen benötigt. In Tabelle 5.1 und Tabelle 5.2 sind die wichtigsten zeitbezogenen Operationen, wie sie in ([Paskamp1999], S. 32) beschrieben werden aufgelistet.

Zeitrelationen

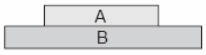
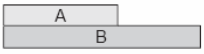
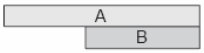
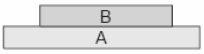
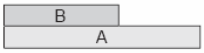
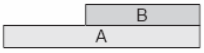
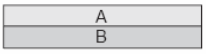
Relation	Definition	grafische Darstellung
A *before* B	$A_e < B_s$	
A *meets* B	$A_e = B_s$	
A *overlaps* B	$A_s < B_s < A_e < B_e$	
A *during* B	$B_s < A_s < A_e < B_e$	
A *starts* B	$A_s = B_s \wedge A_e < B_e$	
A *finishes* B	$B_s < A_s \wedge A_e = B_e$	
A *before* B	$A_s > B_e$	
A *meets* B	$A_s = B_e$	
A *overlaps* B	$A_e > B_e > A_s > B_s$	
A *during* B	$A_s < B_s < B_e < A_e$	
A *starts* B	$A_s = B_s \wedge A_e > B_e$	
A *finishes* B	$B_s > A_s \wedge A_e = B_e$	
A *equal* B	$A_s = B_s \wedge A_e = B_e$	

Tabelle 5.1: 13 Zeitrelationen nach [Allen1983] (adaptiert nach ([Paskamp1999], S. 2)). s und e stehen für Startzeitpunkt und Endzeitpunkt.

Zeitbezogene Operationen

Name	Operand 1	Operand 2	Ergebnis
obere Intervallgrenze	Zeitintervall		Zeitpunkt
untere Intervallgrenze	Zeitintervall		Zeitpunkt
Intervalllänge	Zeitintervall		Zeitspanne
before, meets, starts, finishes, during, overlaps, equal	Zeitintervall	Zeitintervall	Boole'scher Wert
Durchschnitt, Vereinigung, Differenz	Zeitintervall	Zeitintervall	Zeitintervall
Intervall kürzen/strecken	Zeitintervall	Zeitspanne	Zeitintervall
$<;>;\leq;\geq;=$	Zeitpunkt	Zeitpunkt	Boole'scher Wert
+ oder -	Zeitpunkt	Zeitspanne	Zeitpunkt
Abstand	Zeitpunkt	Zeitpunkt	Zeitspanne
Durchschnitt	Zeitpunkt	Zeitintervall	Zeitpunkt
$<;>;\leq;\geq;=$	Zeitspanne	Zeitspanne	Boole'scher Wert
+ oder -	Zeitspanne	Zeitspanne	Zeitspanne

Tabelle 5.2: Zeitbezogene Operationen (nach ([Passkamp1999], S. 32)).

5.1.2.4. Präsentation von Anfrageergebnissen

Die Visualisierung von Daten kann die Handhabung großer Datenmengen und das Erkennen von Zusammenhängen darin erleichtern.

Neben der Ausgabe von Ergebnissen in Form von, z. Bsp. Tabellen, sollten in medizinischen Anfragesystemen, allein schon aufgrund der enormen Datenmengen, auch Mechanismen für ihre graphische Präsentation vorhanden sein. Die Visualisierung von Daten kann die Handhabung großer Datenmengen und das Erkennen von Zusammenhängen darin erleichtern. Zugrunde liegende Prozesse können besser verstanden werden und Anwender können das System früher eigenständig benutzen, da sie Zusammenhänge in den Daten ohne langwierige Schulung sehen.

Visualisierung von Zwischenergebnissen

Oft ist es wünschenswert, die Zwischenergebnisse ebenfalls zu visualisieren, weil dadurch das Vertrauen und Verständnis des Anwenders in die Ergebnisse der Datenselektion verbessert werden kann. Ein gutes Beispiel stellt das RPDR Query Tool, mit seiner Visualisierung der Anfrageergebnisse, dar.

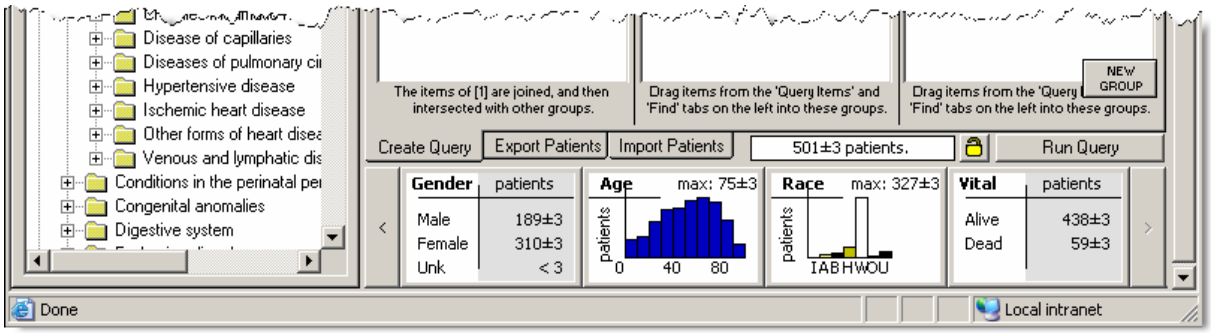


Abbildung 5.1: Visualisierung des Anfrageergebnisses (adaptiert nach [Murphy2003]).

VISAmed soll, ähnlich dem RPDR Query Tool, die Zwischenergebnisse wie auch die Ergebnisse visualisieren und damit dem Anwender ein ständiges Feedback über seine Anfragetätigkeit liefern.

5.1.2.5. Natürliche Metapher

Eine Anfrage an eine Datenbank wird bei Relationalen Datenbanken als SQL-Anfrage an das Datenbankmanagementsystem gesendet. Hier wird die Anfrage von einem Parser in die relationalen Operationen zerlegt, die nötig sind um die Anfrage zu beantworten.

Logische Operatoren in der relationalen Algebra.

Die Relationale Algebra definiert die folgenden logischen Operatoren, mit denen alle weiteren Operatoren gebildet werden können ([Kemper2004], S. 82ff):

- Projektion
- Selektion
- Kreuzprodukt

- Umbenennung
- Vereinigung
- Differenz

Metapher für die Navigation.

In VISAmed sollen für die Operatoren Projektion, Selektion, Vereinigung und zusätzlich Navigation Metapher verwendet werden die für den Anwender eine natürliche Bedeutung und Verständlichkeit haben. Die Navigation kann etwa durch einen Dateiauswahlbaum dargestellt werden, die Vereinigung z. Bsp. durch Venn–Diagramme.

5.1.2.6. Dynamische Anfragegenerierung

Dynamic Query – Alternative zur SQL. Anfrage.

Als visuelle Alternative zur SQL-vermittelten Datenbankabfrage können so genannte Dynamic Queries (vgl. [Card1999], [Fishkin1995], [Oellien2002]) verwendet werden. Im Gegensatz zum SQL-basierten Datenbanksuchen erfordern Dynamic Queries kein Spezialwissen von Anwenderseite. Weiters erlaubt diese Technik oft eine schnellere und einfachere Abfrage der Datenbank. Dynamic Queries führen zu einer unmittelbaren Aktualisierung der dargestellten Datenlandschaft und stellen daher ein wichtiges Interaktionsinstrument dar. Im Prinzip können alle graphischen Standardeingabeelemente wie Schieberegler, Checkboxes und Radiobuttons, aber auch andere applikationsspezifischen Eingabeelemente der graphischen Benutzerschnittstelle als Dynamic Query-Werkzeuge verwendet werden. Diese Auswahlelemente der Benutzeroberfläche werden mit Datendimensionen verknüpft, wobei die einzelnen Variablenwerte auf den Auswahlelementen abgebildet werden. Die Selektion von Werten führt schließlich zur Aktualisierung der graphischen Darstellung. Dem Anfänger eröffnen sich somit Wege, um auch komplexe Datenbankrecherchen auf Basis visueller Suchstrategien durchzuführen. Benutzer, die bereits über SQL- bzw. Datenbankerfahrung verfügen, können mit Hilfe von verschiedenen Kombinationen der dynamischen Filter schnell und einfach komplizierte Suchanfragen realisieren. Die Dynamic Query-Technik erfüllt somit eine Reihe der weiter unten erwähnten Richtlinien für die Realisierung von Interaktionswerkzeugen, wie beispielsweise das unmittelbare Feedback.

5.1.2.7. Patientendatenschutz

Sorge über die Sicherheit von Patientendaten ist durchaus berechtigt.

Der Zugang zum Anfragetool für medizinische Datenbanken steht häufig einer großen Menge an Personen offen. Denkbar ist eine unbefugte Benutzung, etwa durch Einbruch in das System oder sorgloser Umgang mit den Zugangsdaten. Die Sorge über die Sicherheit von Patientendaten ist also durchaus berechtigt. Es muss unbedingt

sichergestellt werden, dass individuelle Patientendaten nicht über das Anfragetool abgefragt werden können bzw. dass entsprechende Versuche entdeckt werden.

Ansätze zum Patientendatenschutz.

Es werden verschiedene Ansätze zur Lösung dieser Problematik verfolgt. In [Murphy2002] wird eine Methode beschrieben, bei der beide Aspekte aus dem letzen Absatz berücksichtigt werden.

Der Patientendatenschutz ist für VISAmed ein wichtiges Thema, würde jedoch den Rahmen dieser Arbeit sprengen und wird deswegen nicht weiter behandelt.

5.2. Probleme bei Anfragesprachen

Probleme bei Anfragesprachen.

In [Murray2000] werden einige Probleme identifiziert, die im Zusammenhang mit Anfragesprachen auftreten. Murray differenziert in Probleme

- der Sprache selbst:
  - AND und OR: ihre, von der booleschen Logik abweichende, umgangssprachliche Verwendung führt zu Verwirrungen, denn das umgangssprachliche UND entspricht dem booleschen OR.
  - SUM und COUNT: Probleme die unterschiedliche Bedeutung zu erfassen
  - GROUP BY: mangels umgangssprachlicher Verwendung dieses Terminus, wird seine Notwenigkeit ignoriert.
  - Klammerungen: hohe Verschachtelungstiefe bedingt hohe Komplexität der Anfrage und führt zu Verständnisproblemen
  - FOR ALL, EXISTS, IN: mangelnde Erfahrung in ihrer Verwendung erschwert ihre korrekte Anwendung
- der Benutzerschnittstelle der Sprache:
  - Mangelnde Unterstützung bei der Bildung von JOINS: unterschiedliche Benennung gleicher Attribute erschwert das Verständnis für Joins.
  - Mangelnde Unterstützung bei der Anwendung von Operatoren: unerfahrene Benutzer haben eine hohe Lernkurve und machen Fehler in der Syntax und Semantik der Anfrage.
  - Mangelnde Visualisierung des Datenbankschemas: der Anwender bekommt meist zu wenig Information über die Datenbankinterna.
  - Fehlende History-Funktion: Anwender greifen, bei der Formulierung neuer Anfragen, gern auf alte,

funktionierende Anfragen zurück. Diese sollten im System verfügbar sein.

Bei der Realisierung von VISAmed erscheint eine entsprechende Beachtung der oben erwähnten Probleme sinnvoll. So soll von der deskriptiven Anfragesprache der Datenbank zu einer interaktiven visuellen Darstellung für die Anwender abstrahiert werden.

Abstraktion von deskriptiv nach interaktiv visuell.

5.3. Richtlinien für Interaktionswerkzeuge

Richtlinien für Interaktionswerkzeuge

Das Vertrauen und Verständnis des Anwenders in die Ergebnisse der Datenselektion kann durch geeignete intuitive Interaktionswerkzeuge maßgeblich erhöht werden. Dabei macht es Sinn einige Richtlinien einzuhalten:

- Realisierung einfacher Interaktionssequenzen
- Vermeidung von unübersichtlichen Funktionen bzw. auch zu vieler verschiedener Funktionen
- Ständiger Zugriff auf alle wichtigen Funktionen zu jedem Zeitpunkt der Interaktion
- Verfügbarkeit geeigneter Feedback-Mechanismen
- Undo-Funktionalität bei irrtümlichen Benutzereingaben

Konzept der Direct Manipulation

Die visuelle Schnittstelle von VISAmed soll dem Konzept von Direct Manipulation [Shneiderman1983] folgen. Die Befehle für die Anfragegenerierung werden nicht über Kommandos, wie etwa Eintippen von Befehlen oder Ausfüllen von Dialogboxen, sondern durch direktes Anklicken der entsprechenden Werkzeuge ausgeführt. Die direkte Manipulation erlaubt so ein natürlicheres und einfacheres Arbeiten.

5.4. Einbettung in elementare Abbildungsschichten

Schicht des visuellen Anfragesystems

Im Abschnitt 3.1.3 auf Seite 54 wurde das 5-Schichtenmodell vorgestellt. Dort wurde dem Schichtenmodell eine weitere Schicht, die Schicht des visuellen Anfragesystems, aufgesetzt.

Drei Abstraktionsschritte notwendig

Für die Realisierung dieser Schicht muss von der deskriptiven Anfragesprache der vorangegangenen Schicht zu einer visuellen Darstellung für den Anwender abstrahiert werden. Es sind drei Abstraktionsschritte notwendig, wie in Abbildung 5.2 weiter unten dargestellt. Auf der untersten Ebene stellt die Datenbankschnittstelle die Abstraktion der konkreten Einbettung in die Programmiersprache dar. Hier können Methoden der Programmiersprache aufgerufen wer–

den, die ihrerseits SQL-Anweisungen ausführen. Auf der nächsten Ebene stellt der SQL-Translator, unter Verwendung von SQL-Methoden der visuellen Anfragesprache Operatoren zur Verfügung. Die visuelle Anfragesprache wird schließlich dem Anwender, über die graphische Oberfläche, zur Verfügung gestellt.

VISAmed, so wie es in dieser Arbeit betrachtet wird, stellt dabei die graphische Oberfläche im visuellen Anfragesystem dar.

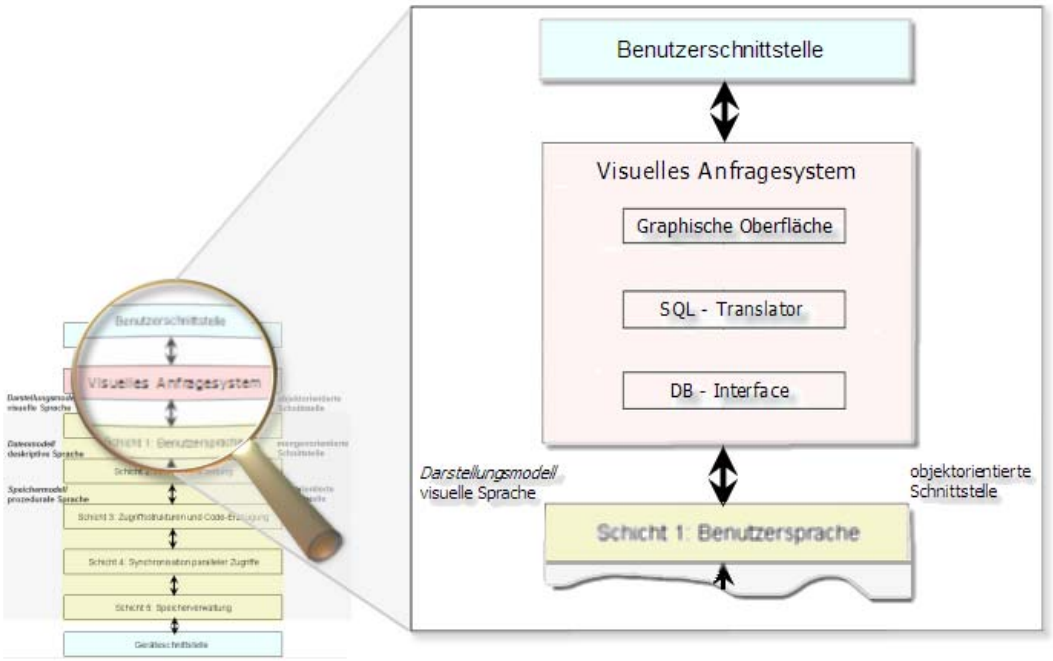


Abbildung 5.2: Betrachtung des visuellen Anfragesystems im 5-Schichtenmodell.

5.5. Zusammenfassung

Der Anwender von VISAmed hat in der Regel keine Kenntnisse über den Datenbestand der Datenbank noch ist er mit datenverarbeitungstechnischen Begebenheiten vertraut. Vorhandene Datenbestände sollen jedoch auch von unerfahrenen Anwendern selektiert, kombiniert und parametrisiert werden können. VISAmed zielt auf die entsprechende Unterstützung dieser unerfahrenen Anwender ab, indem:

- sie weitgehend intuitiv zu nutzen ist aber von technischen Details abstrahiert
- genügend Freiräume zur Konfiguration und Manipulation anfragespezifischer Einzelschritte bietet

Demnach sind neben der „ad–hoc“-Anfragemöglichkeit, die Minimierung der linguistischen, strukturellen und

intentionalen Forderungen die wichtigsten Kriterien, die VISAmed erfüllen soll.

In Folge dieser Kriterien wird VISAmed das Prinzip der direkten Manipulation verfolgen, das bedeutet:

- Anfragekomponenten und Ergebnisse werden visuell dargestellt.
- während der inkrementellen, reversiblen Anfrageformulierung (mittels Anklicken und nicht mittels Eintippen) erfolgt unmittelbares und ständiges Feedback.

VISAmed soll vom Anwender minimales Wissen erfordern. Daher soll in verschiedenen Abstraktionsstufen nur der notwendige Überblick über die vorhandenen Daten und deren Zusammenhänge gegeben werden. Das bedeutet aber auch, dass einige Bereiche dem Anwender verborgen bleiben.

KAPITEL

6

Das VISAmed Tool

Nimm das an, was nützlich ist...
Lass das weg, was unnütz ist...
Füge das hinzu, was dein Eigenes ist...

Bruce Lee

Die vollständige Implementierung eines einsetzbaren Anfragegenerators geht über den Rahmen dieser Arbeit hinaus. Die Erstellung eines einsatzfähigen Systems erfordert noch eine umfassende Betrachtung bezüglich der Konzeption und Realisierung der Schnittstellen zum Datenbankmanagementsystem und statistischem Auswertesystem. Dieses Kapitel geht auf die Gesamtarchitektur von VISAmed ein, ohne jedoch alle Details auszuarbeiten, und liefert gewissermaßen eine Grundlage für Anschlussarbeiten.

6.1. VISAmed

Wie bereits in den vorangegangenen Kapiteln dargestellt, besteht das Ziel dieser Arbeit in der Konzeption eines Anfragegenerators, genauer einer visuellen Anfrageumgebung, die ein flexibles Interaktionsmodell unterstützt, das vor allem dem unerfahrenen Benutzer eine Datenanalyse ermöglicht und den interaktiven Umgang mit Datenbankanfragen vereinfacht.

Spracheigenschaften von VISAmed.

In Anlehnung an die Einteilung der visuellen Anfragesprachen in [Batini1991] und [Catarci1997], soll VISAmed folgende Spracheigenschaften aufweisen:

- Browserfunktionalität
- Menübasiertheit
- Iconbasiertheit
- Diagrammbasiertheit

Eine weitere von Batini et al. angeführte Spracheigenschaft, die Formularbasiertheit, soll nicht verwendet werden. Das Ausfüllen der Formulare setzt Kenntnisse voraus, welche Werte wo eingesetzt werden müssen bzw. dürfen und erhöht damit die Anforderungen an den Anwender.

VISAmed soll besonders auf die Konzepte vom RPDR Query Tool, sowie vom Kaleidoquery zurückgreifen.

Anfrageformulierung läuft in mehreren Phasen ab.

Die Anfrageformulierung soll in mehreren Phasen ablaufen. VISAmed startet mit einem Datenüberblick, damit sich der Anwender zunächst in der Datenbank orientieren kann. Mit Hilfe dieses Datenüberblicks wählt er die für die Anfrage relevanten Daten aus. Der Überblick über die ausgewählten Daten bleibt immer erhalten – die ausgewählten Daten werden in weiteren Fenstern visualisiert (siehe Abbildung 6.1). An die selektierten Daten können verschiedene Bedingungen gestellt werden. Nachdem die Anfrage durchgeführt worden ist, erfolgt die Datenausgabe. Für den Anwender sind also Datenüberblick, Anfrage und Anfrageergebnis immer gleichzeitig sichtbar.

Direkte Manipulation minimiert die Anforderungen an den Anwender.

Um die strukturellen Anforderungen an den Anwender zu minimieren, d.h. die Forderung nach Kenntnis der Sprachsyntax zu verringern, soll der Anwender mittels direkter Manipulation durch die Formulierung der Anfragebedingungen geführt werden. Der Anwender erstellt die Anfrage demnach intuitiv durch Interaktionen, wie etwa durch Selektion verschiedener Symbole.

6.2. Visuelle Konzepte in VISAmed

Die Komponenten in VISAmed stellen die Konstrukte dar, die dem Anwender ermöglichen ad–hoc–Anfragen in VISAmed zu bilden. Im Folgenden werden Konzepte der Anfragephasen – Datenüberblick, Anfrage und Anfrageergebnis – betrachtet.

6.2.1. Visualisierung des Datenüberblicks

Datenüberblick zur Orientierung des Anwenders.

Die Visualisierung des Datenüberblicks hilft dem Anwender sich in der Datenbank zu orientieren und die für die Anfrage relevanten Daten auszuwählen.

In VISAmed wird der Datenüberblick durch zwei Auswahlfenster dargestellt. Das erste Fenster stellt die verfügbaren Formulare dar, im zweiten Fenster werden die verfügba–

ren Variablen dargestellt (Abbildung 6.1). Vergleiche dazu auch ArchiMed, das im Kapitel 2 beschrieben wurde.

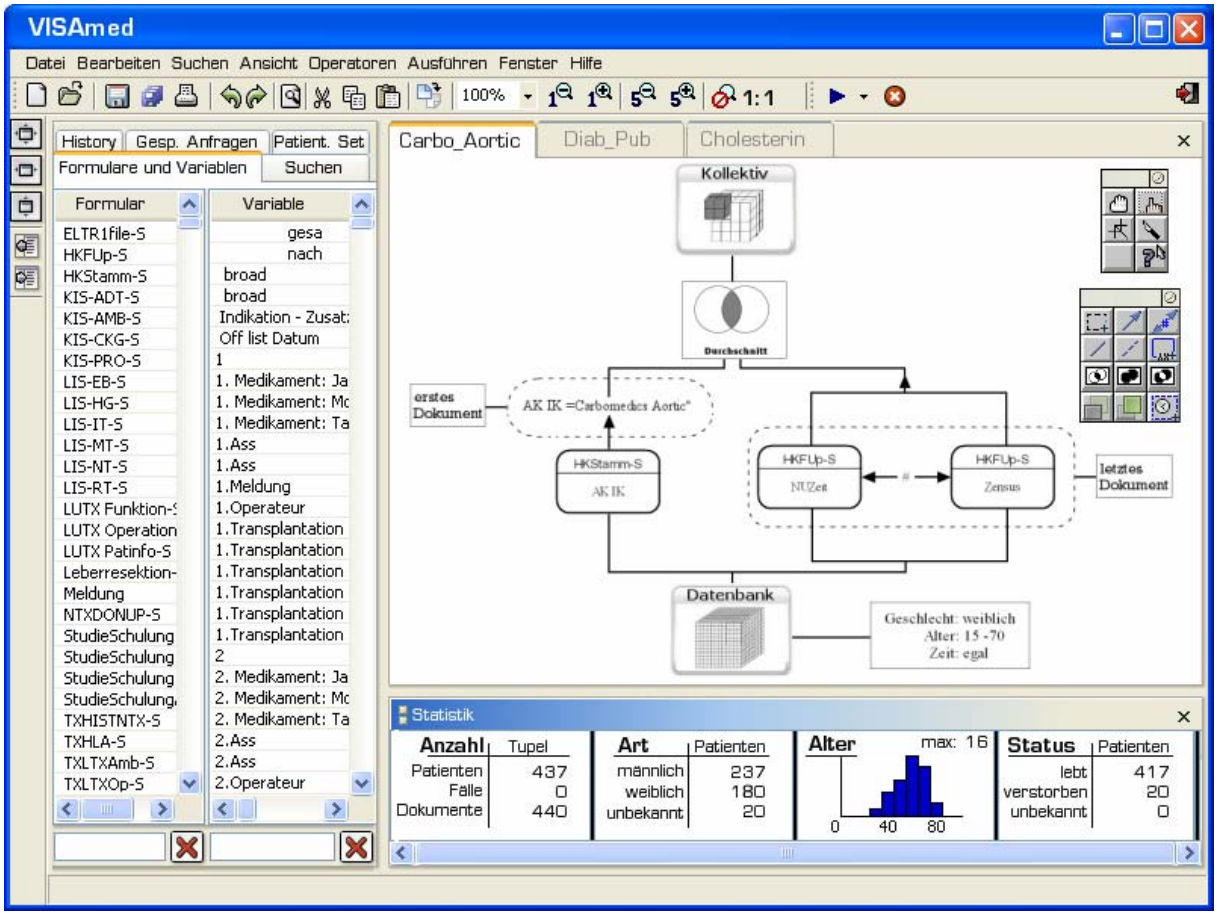


Abbildung 6.1: Die VISAmed Benutzeroberfläche: links ist das Auswahlfenster „Formulare und Variablen“ zu sehen.

Benutzerdefinierte–Anfrageelemente

Gespeicherte Anfrage

Patienten Set

In Anlehnung an das RPDR Query Tool [Murphy2003] werden neben diesen Standardanfrageelementen Benutzerdefinierte–Anfrageelemente verwendet. Benutzerdefinierte–Anfrageelemente bestehen aus gespeicherten Anfragen und Patienten Sets. Alle definierten Anfragen können als benutzerdefinierte Anfragen und Patienten Sets abgespeichert werden und als Elemente in neuen Anfragen verwendet werden. Nach Aktualisierung der Datenbank liefern gespeicherte Anfragen neue Resultate, bei Patienten Sets ändert die Aktualisierung nicht das ursprüngliche Resultat.

Suchen

Eine Suchen-Funktion erlaubt eine einfache alphanummerische Suche. Die Suche kann nach Formularen oder Variablen durchgeführt werden.

History

Alte Datenbankanfragen sind unter den Punkten History und gespeicherte Anfragen zu finden. Wie in [Murphy2003] bemerkt wird, verwendet sie der Anwender in etwa der Hälfte der Fälle als Schablone für neue Anfragen.

6.2.2. Visualisierung der Anfragebedingungen

VISAmed hat eine visuelle Syntax, die auf der textuellen Anfragesprache SQL basiert.

"filter flow" Metapher als Grundidee.

Die Grundidee hinter VISAmed ist die "filter flow" Metapher, die bereits im Abschnitt **4.1.3.1** auf Seite 83 kurz erwähnt wurde. Das Verarbeiten einer Anfrage wurde mit fließendem Wasser verglichen, das durch etliche Rohre und Filter fließt. Die Rohre sind mit AND– und OR–Verbindungen verknüpft und die Filter lassen nur eine bestimmte Art von Durchfluss zu. Nach dem Prinzip des "filter flow"-Modells werden aus einer Menge von Informationen jene herausgefiltert, die für das Ergebnis der Anfrage relevant sind.

Basissymbole von VISAmed basieren auf Kaleidoquery.

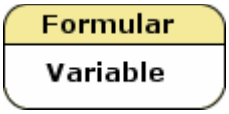
Die Basissymbole von VISAmed basieren grundsätzlich auf denen von Kaleidoquery (vergleiche Abschnitt **4.1.3.1**). Dort werden Entities auf Grund von booleschen Prädikaten selektiert. Diese booleschen Prädikate werden auf die Attributwerte eines Entities oder auf die Attributmenge von zueinander in Beziehung stehender Entities angewendet. Das Ergebnis einer solchen Anfrage sind ausgewählte Attribute und Beziehungen jedes selektierten Entities. In medizinischen Datenbanken beinhaltet die Selektionsbedingung zumeist auch temporale Bedingungen und es ist oft wünschenswert die Ausgabe auf einen bestimmten Zeitraum einzuschränken. Zur Konstruktion solcher temporaler Anfragen werden neue Anfragekonstrukte benötigt. Die dazu angestellten Überlegungen basieren auf [Kuhn1997].

In weiterer Folge werden zunächst die Konstrukte von VISAmed benannt, anschließend folgt die Betrachtung der temporalen Anfragen.

6.2.2.1. Konstrukte von VISAmed

Die Konstrukte von VISAmed können in Objekte, Zugriffspfade und Klammerungen unterteilt werden.

Objekte sind:



Variablen: Eine Variable bezeichnet einen Platz auf einem Formular und kann bestimmte Werte annehmen. Der graphische Variablenname setzt sich aus dem Namen des Formulars und dem Namen der Variable zusammen.

Konstanten: Eine Konstante hängt mit einer Variablen zusammen und kann Werte wie Zahl, Datum, Uhrzeit, Datum–Uhrzeit–Dupel oder Text darstellen.

Einschränkungen: Eine Einschränkung reduziert die zu berücksichtigenden Werte aufgrund von UND–verbundenen Patienten–/Dokumenteneigenschaften,

die ihrerseits durch Bedingungen und Zugriffspfade beschrieben werden können. Bedingungen werden durch *Funktionen* (Minimum, Anzahl, Summe, die ersten/letzten n Fälle), *Gruppierungen* (Patient, Fall, Dokument) und *Objekte* (Variable, Fall, Dokument) limitiert. Einschränkungen können über Klammerungen mit mehreren Variablen gepaart sein.

Zugriffspfade sind:

Zugriffspfade mit und ohne Operatoren: Operatoren können ein *Textvergleich* (IST [NICHT], ENTHÄLT [NICHT], eine *Rechenanweisung* (PLUS, MINUS, MULTIPLIZIERT MIT, DIVIDIERT DURCH), ein *arithmetischer Vergleich* (KLEINER [ODER GLEICH], [NICHT] GLEICH, GRÖSSER [ODER GLEICH], INNERHALB INKLUSIV/EXKLUSIV, AUSSERHALB EXKLUSIV/INKLUSIV) oder eine *logische Verknüpfung* (UND, ODER) sein. Nach UND können Zusammenhänge wie, [NICHT] AM SELBEN DOKUMENT, [NICHT] BEIM SELBEN FALL, [NICHT] IN DER /IM/AM SELBEN (Jahr, Tag, Minute, Sekunde), [WENIGER/MEHR ALS (Anzahl + Einheit] DAVOR/DANACH und WENIGER/MEHR ALS (Anzahl + Einheit). Mehrere Operatoren werden mit einer ODER–Verknüpfung verbunden.

Klammerungen sind:

Kasten: Eine Klammer wird durch gestrichelten Kasten dargestellt. Ein Klammer umschließt mehrere Objekte, dabei können Klammern, Klammern umschließen aber nicht überschneiden. Temporale Bedingungen werden ebenfalls durch Klammerung ausgedrückt.

6.2.2.2. Vergleiche und Aggregatfunktionen

Vergleiche

VISAmed unterstützt Vergleiche von numerischen Datentypen, Zeichenketten und Mengen. Die Vergleichsoperatoren entsprechen denen von SQL (<=, <, >, >=, =, like, usw.). Abbildung 6.2 zeigt das Beispiel einer visuellen Repräsentation einer einfachen Vergleichsoperation.

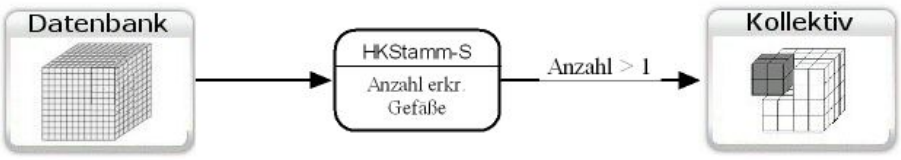


Abbildung 6.2: Ein einfacher Vergleich. Es werden alle Patienten ausgegeben mit mehr als einem erkrankten Gefäß.

Aggregatfunktionen

Aggregatfunktionen wie COUNT, SUM, AVG, MAX, MIN, werden in VISAmed auf das Ergebnis der Anfrage angewendet. Das Ergebnis wird dabei in einem neuen Fenster am unteren Rand der Arbeitsoberfläche angezeigt. Die Aggregatfunktion wird, wie in Abbildung 6.3 dargestellt, von einer weiteren Ergebnisbox umrandet. Die Funktion mit dem Attribut wird in der Umrandung angegeben. Aggregatfunktionen werden zusammen mit GROUP BY (siehe Abschnitt 6.2.2.5, weiter unten) verwendet.

VISAmed verwendet für die Benennung der Aggregatfunktionen umgangssprachliche Begriffe. Der Vergleich mit SQL Begriffen ist in Tabelle 6.1 dargestellt.

Aggregatfunktionen	
VISAmed Funktionen	**SQL Funktionen**
Anzahl ohne Leereinträge (Anzahl)	COUNT
Anzahl mit Leerenträgen (Anzahl L)	COUNT *
Anzahl unterschiedlicher Werte (Anzahl U)	COUNT DISTINCT
Durchschnitt aller unterschiedlichen Werte (Durchschnitt U)	AVG DISTINCT
Durchschnitt gesamt (Durchschnitt)	AVG
Maximum	MAX
Minimum	MIN
Summe	SUM
Standardabweichung	STDDEV
Varianz	VARIANCE

Tabelle 6.1: Gegenüberstellung Aggregatfunktionen in VISAmed und SQL.

Im Beispiel unten berechnet die Aggregatfunktion die Anzahl der Patienten mit mehr als einem erkrankten Gefäß.

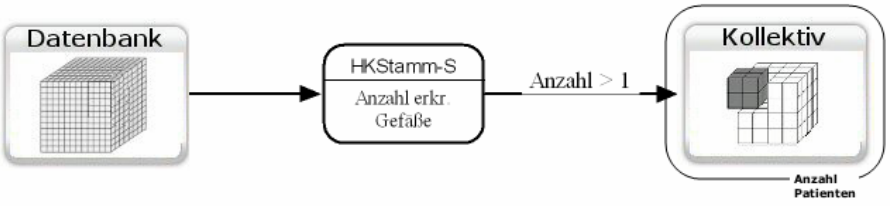


Abbildung 6.3: Aggregatfunktion in VISAmed.

6.2.2.3. Boolesche Ausdrücke und Mengenausdrücke

Zur Verknüpfung von Bedingungen werden boolesche Ausdrücke (AND, OR und NOT) verwendet.

AND, OR und NOT

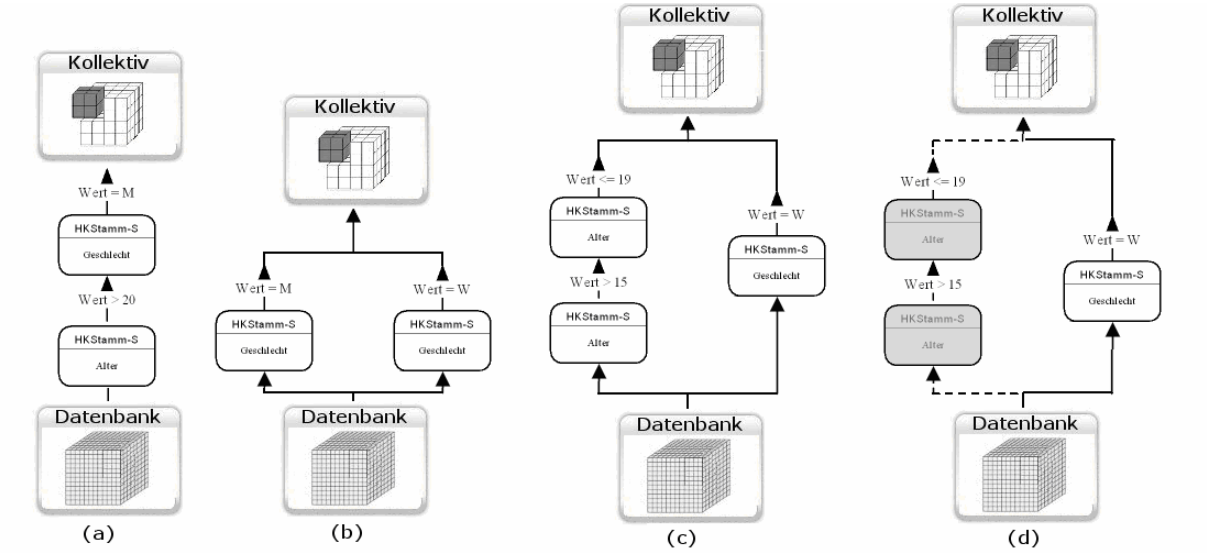


Abbildung 6.4: Visualisierung von AND, OR und NOT (adaptiert nach [Murray2000]).

Darstellung (a) in Abbildung 6.4 zeigt die visuelle Repräsentation einer UND-Verknüpfung. Bedingungen werden in UND-Verknüpfungen durch Pfeile hintereinander geschaltet. Die dargestellte Verknüpfung liefert als Ergebnis alle Personen, die älter als 20 Jahre alt und männlich sind.

Darstellung (b) zeigt eine ODER-Verknüpfung. ODER-Verknüpfungen werden durch Verzweigen eines Pfeils visualisiert. Nach der Definition der Bedingungen werden die beiden Äste der Pfeile wieder zusammengeführt. Die dargestellte Verknüpfung liefert als Ergebnis alle Personen, die entweder männlich oder weiblich sind.

Darstellung (c) kombiniert UND- und ODER-Verknüpfungen. Es werden alle Personen ausgewählt, deren Alter entweder zwischen 15 und 19 Jahren liegt oder die weiblich sind.

Darstellung (d) erweitert die Darstellung (c) um eine Negation (NOT) der UND-Verknüpfung. Negationen werden durch gestrichelte Linien und grau ausgefüllte Boxen dargestellt. Die dargestellte Verknüpfung liefert als Ergebnis alle Personen, deren Alter nicht zwischen 16 und 19 Jahren liegt oder die weiblich sind.

In VISAmed werden die Mengenausdrücke (VEREINIGUNG, DURCHSCHNITT, DIFFERENZ) mit Hilfe von Venn-Diagrammen repräsentiert. Der Vorteil dieser Art von Dia–

VEREINIGUNG, DURCHSCHNITT, DIFFERENZ

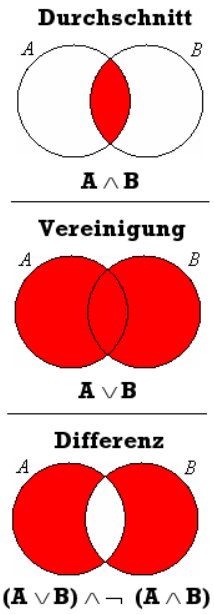


Abbildung 6.5: *Zusammenhang zwischen Venn–Diagrammen und booleschen Ausdrücken.*

grammen, gegenüber den booleschen Ausdrücken, liegt darin, dass sich mit ihnen alle Relationen zwischen den betrachteten Mengen übersichtlich darstellen lassen. Die Ergebnismenge der Ausdrücke ist jeweils grau markiert.

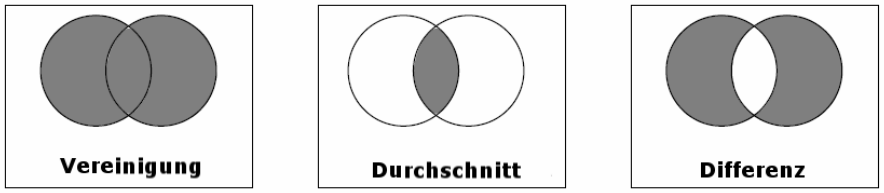


Abbildung 6.6: Mengenausdrücke, Vereinigung, Durchschnitt, Differenz (adaptiert nach [Murray2000]).

In der Abbildung 6.7 weiter unten werden Mädchen selektiert die unter Diabetes leiden und eine verzögerte Pubertät aufweisen.

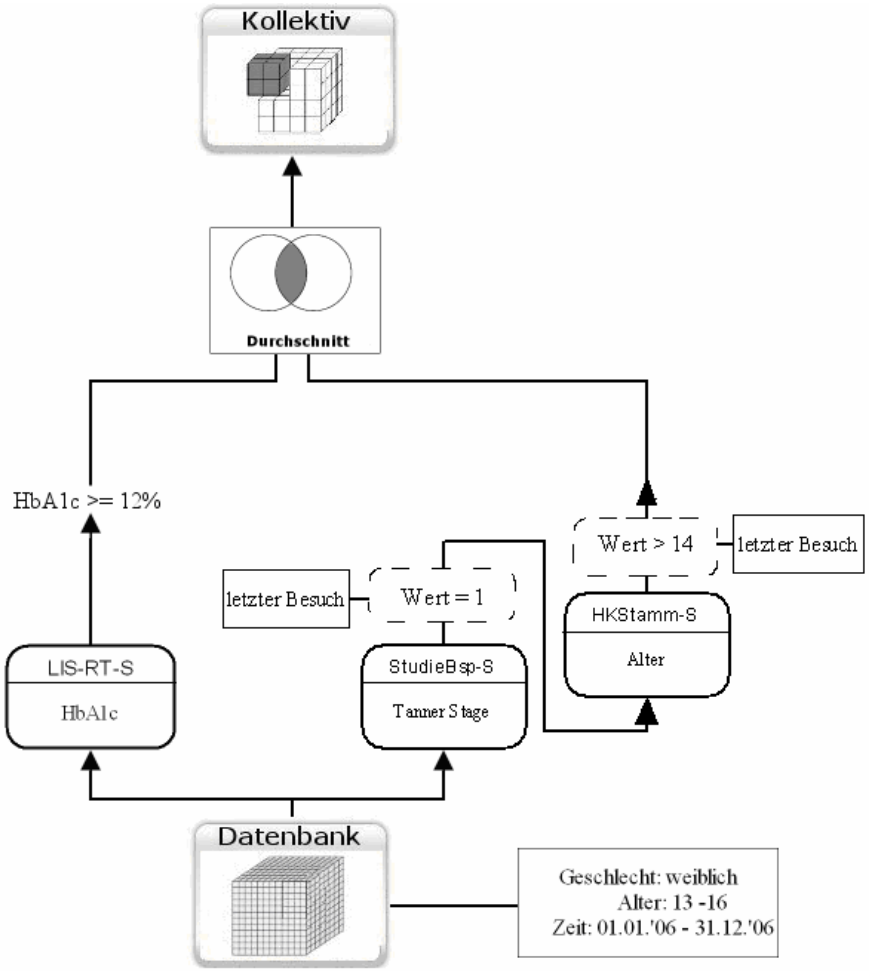
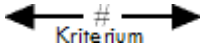


Abbildung 6.7: Ein Beispiel für eine Anwendung von Mengenausdrücken, booleschen Ausdrücken und Zugriffspfaden – Mädchen mit Diabetes und verzögerter Pubertät. Diese Anfrage ließe sich auch nur mit booleschen Ausdrücken realisieren, allerdings auf Kosten der Übersichtigkeit.

In Abbildung 6.7 werden zwei Einschränkungen vorgenommen. Zunächst werden die Daten aus der Datenbank bezüglich *Geschlecht*, *Alter* und *Dokumentationszeit* global eingeschränkt, danach wird eine Einschränkung bezüglich der Zeit der Befunderhebung des Tanner Stagings erstellt, indem die Variablen *Alter* und *Tanner Stage* gemeinsam beim letzten Arztbesuch erhoben werden sollen. Die Beziehung zur Einschränkung ist durch einen horizontalen Pfeil dargestellt. Die zeitlich eingeschränkten Variablen werden mit einem gestrichelten Kasten markiert.

6.2.2.4. JOIN

JOIN

Der JOIN–Operator wird typischerweise verwendet, um Beziehungen zu bilden, die schon implizit zwischen den Relationen bestehen. In VISAmed hat der entsprechende Operator das Symbol 

. Das JOIN–Kriterium gibt an welche Variablenwerte in einer Zeile kombiniert werden. Das JOIN–Kriterium kann Patient, Fall, Dokument und eine logische oder zeitliche Bedingung sein.

JOIN–Kriterium

In Abbildung 6.8 werden die Variablen *NUZeit* und *Zensus* miteinander verknüpft. Zu beachten ist dabei, dass die Zeilen der beiden Operanden so verknüpft werden, dass beide Werte in die Ergebniszeile kommen – es können Missingwerte[37] entstehen. Folgendes theoretische Beispiel soll das verdeutlichen.

Gegeben sind die Relationen R und S:

R

A	B	C
1	2	3
4	5	6

S

C	D	E
3	7	8
9	10	11

Ergebnisrelation:

A	B	C	D	E
1	2	3	7	8
4	5	6	*null*	*null*
null	*null*	9	10	11

In der Ergebnisrelation bleiben die Tupel beider Relationen erhalten. Das ist ein wichtiger Punkt bei medizinischen Daten, da Information nicht verloren gehen darf.

Weiterverknüpfen mit Attributen des zweiten Operanden

In VISAmed werden weitere Operationen mit den Attributen des zweiten Operanden verknüpft.

[37] Missingwerte (missing values) stellen unvollständige Information dar und werden im relationalem Datenmodell durch den NULL–Wert repräsentiert.

Regeln für Umgang mit Missingwerten

Das Ergebnis bei Anfragen beim Vorliegen von Missingwerten ist häufig überraschend, denn Daten könnten nicht berücksichtigt werden. In ([Kemper2004], S. 120f) werden dafür folgende Regeln für den Umgang mit Missingwerten genannt:

1. Wenn ein Operand *null* ist, wird auch das Ergebnis *null*.
2. SQL kennt die Werte *true*, *false* und *unknown*. Vergleichsoperationen liefern immer *unknown* zurück, wenn mindestens einer ihrer Argumente *null* ist.
3. Die Berechnung logischer Ausdrücke erfolgt nach folgenden Tabellen:

Logische Verknüpfungen in SQL

NOT	
true	false
unknown	unknown
false	true

AND	true	unknown	false
true	true	unknown	false
unknown	unknown	unknown	false
false	false	false	false

OR	true	unknown	false
true	true	true	true
unknown	true	unknown	unknown
false	true	unknown	false

Tabelle 6.2: Logische Verknüpfungen in SQL (nach [Kemper2004])

4. In einer where–Bedingung werden Tupel, für die die Bedingung zu *unknown* auswertet, nicht ins Ergebnis aufgenommen.
5. Bei einer Gruppierung wird *null* als eigenständiger Wert aufgefasst und in eine eigene Gruppe eingeordnet.

Aus den obigen fünf Regeln lässt sich ableiten, dass wenn ein Ausdruck zu *null* oder *unknown* auswerten kann, er nicht in die Ergebnismenge aufgenommen wird und damit möglicherweise zu falschen Ergebnissen führt, wenn auf das Auftreten von Missingwerten nicht geprüft wird. Mit der Bedingung *is null* bzw. *is not null* lässt ich überprüfen ob ein Ausdruck zu null auswerten kann. Weiters können logische Ausdrücke mit *is unknown* bzw. *is not unknown* getest werden.

Existenz von Missingwerten

Die Existenz von Missingwerten[38] wird in VISAmed bei der Anfrageformulierung immer implizit berücksichtigt.

Beispiel zu Missingwerten

Ein Beispiel soll den Zusammenhang von Missingwerten und JOIN–Verknüpfungen verdeutichen (entnommen aus [Gall2001a]):

[38] Missingwerte = NULL–Werte

Beispiel zu JOIN und Missingwerten

Ein Patient besucht mehrmals eine medizinische Einrichtung. Beim ersten Aufenthalt werden zwei Dokumente (Doku1 und Doku2) generiert. Beim zweiten Aufenthalt wird das Dokument (Doku3) generiert. Die Variablen Var1, Var2 und Var3 kommen auf den Dokumenten vor, wobei nur das erste Dokument (Doku1) alle Variablen enthält.

Patient	Aufenthalt	Dokument	Variable	Wert
Pat1	A1	Doku1	Var1	Wert1
			Var1	Wert2
			Var2	Wert3
			Var3	Wert4
		Doku2	Var1	Wert5
			Var1	Wert6
	A2	Doku3	Var2	Wert7
			Var3	Wert8

Tabelle 6.3: Patientendokumentation

Im Beispiel sollen diese drei Variablen ausgewertet werden. Das Ergebnis ist abhängig davon welche Werte in einer Zeile verlinkt werden sollen, dabei spielen Missingwerte eine wichtige Rolle. Eine Verknüpfung über das JOIN–Kriterium *Dokument* produziert folgende Ergebnisse.

Ergebnis ohne Missingwerte

Ergebnis ohne Betrachtung der Missingwerte:

JOIN–Kriterium	Kriterium	Variable1	Variable2	Variable3
Dokument	Doku1	Wert1	Wert3	Wert4
	Doku1	Wert2	Wert3	Wert4

Tabelle 6.4: JOIN ohne Missingwerte

Ergebnis mit Missingwerten

Ergebnis mit Betrachtung der Missingwerte:

JOIN–Kriterium	Kriterium	Variable1	Variable2	Variable3
Dokument	Doku1	Wert1	Wert3	Wert4
	Doku1	Wert2	Wert3	Wert4
	Doku2	Wert5	*null*	*null*
	Doku2	Wert6	*null*	*null*
	Doku3	*null*	Wert7	Wert8

Tabelle 6.5: JOIN mit Missingwerten

Eine Verknüpfung über das JOIN–Kriterium *Patient* würde sogar eine Tabelle mit 16 Zeilen (4-mal Var1 verlinkt mit 2-mal Var2 verlinkt mit 2-mal Var3) produzieren.

Information auf mehreren Tabellen ⇨ JOIN..

Ein JOIN wird immer dann verwendet, wenn die gesuchte Information auf mehr als einer Tabelle zu finden ist. Für den ungeübten Anwender stellt das ein schwieriges Konstrukt dar. Die Schwierigkeit besteht darin, den Anwender zu motivieren, die Semantik der impliziten Beziehungen zwischen Relationen zu erkennen und durch den JOIN–Operator

explizit zu formulieren. Durch die fixe Attributmenge bei Operanden als Ergebnis von Operationen (siehe Abschnitt **2**.1.3.3) ist es in VISAmed immer möglich einen JOIN zwischen Variablen, über die Patient–ID, Aufenthalt–ID und Dokument–ID durchzuführen.

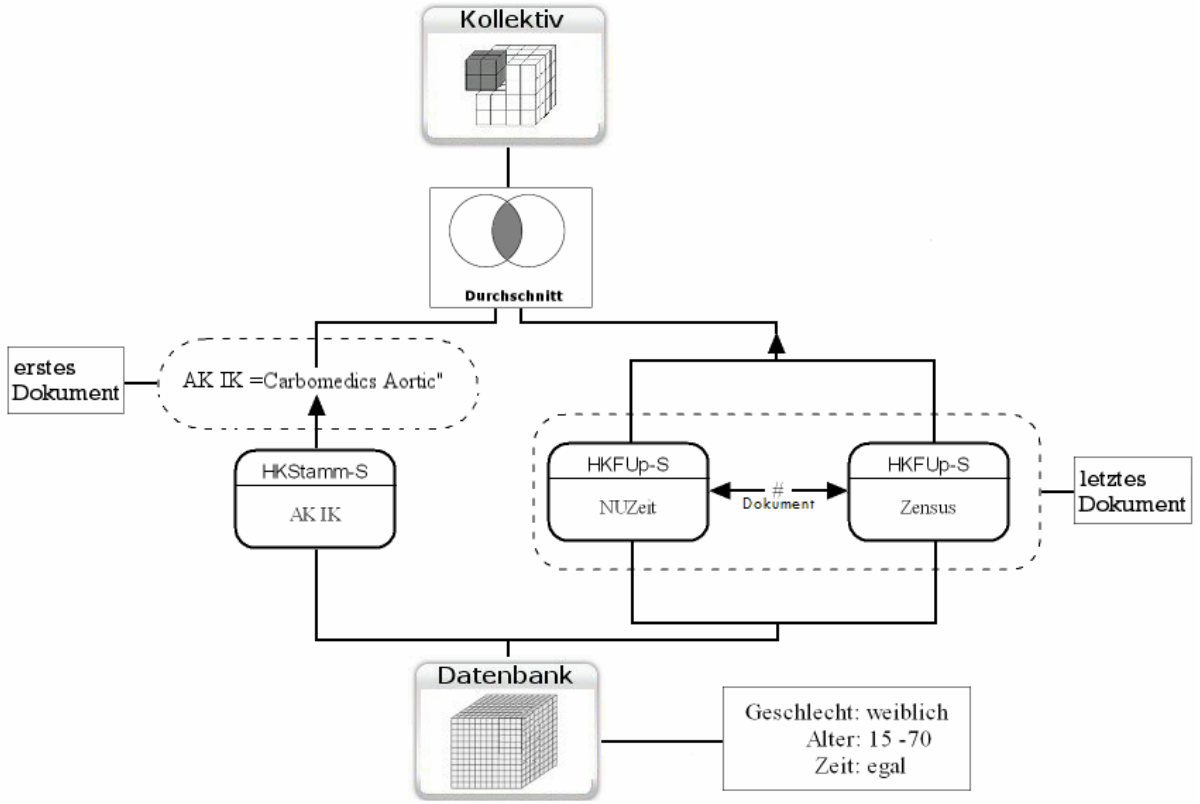


Abbildung 6.8: Ein Beispiel für eine Anwendung von Joins, Zugriffspfaden, globalen Einschränkungen und Zeiteinschränkungen.

6.2.2.5. GROUP BY

GROUP BY HAVING

Nur zusammen mit Aggregatfunktionen

VISAmed unterstützt die Klausel GROUP BY, zur Gruppierung von Ergebnissen, sowie HAVING zur Filterung der gruppierten Ergebnisse. Diese zwei Klauseln können nur in Verbindung mit Aggregatsfunktionen verwendet werden (siehe Abschnitt 6.2.2.2, weiter oben). Durch eine Gruppierung wird das Ergebnis einer Anfrage in Partitionen (Untermengen) aufgeteilt. Gruppiert kann nach Patient, Fall oder Dokument werden. Der GROUP BY – Operator wird wie in Abbildung 6.9 dargestellt, als Einschränkung zu einer Aggregatfunktion angegeben. Wonach gruppiert wird, gibt der Operator mit dem Attribut in der Box an. Optional kann noch ein HAVING – Operator angegeben werden.

VISAmed verwendet für die Benennung des GROUP BY – Operators den Ausdruck *Gruppiert nach* (vergleiche [Welty1981], S. 644). HAVING wird mit *mit* ausgedrückt.

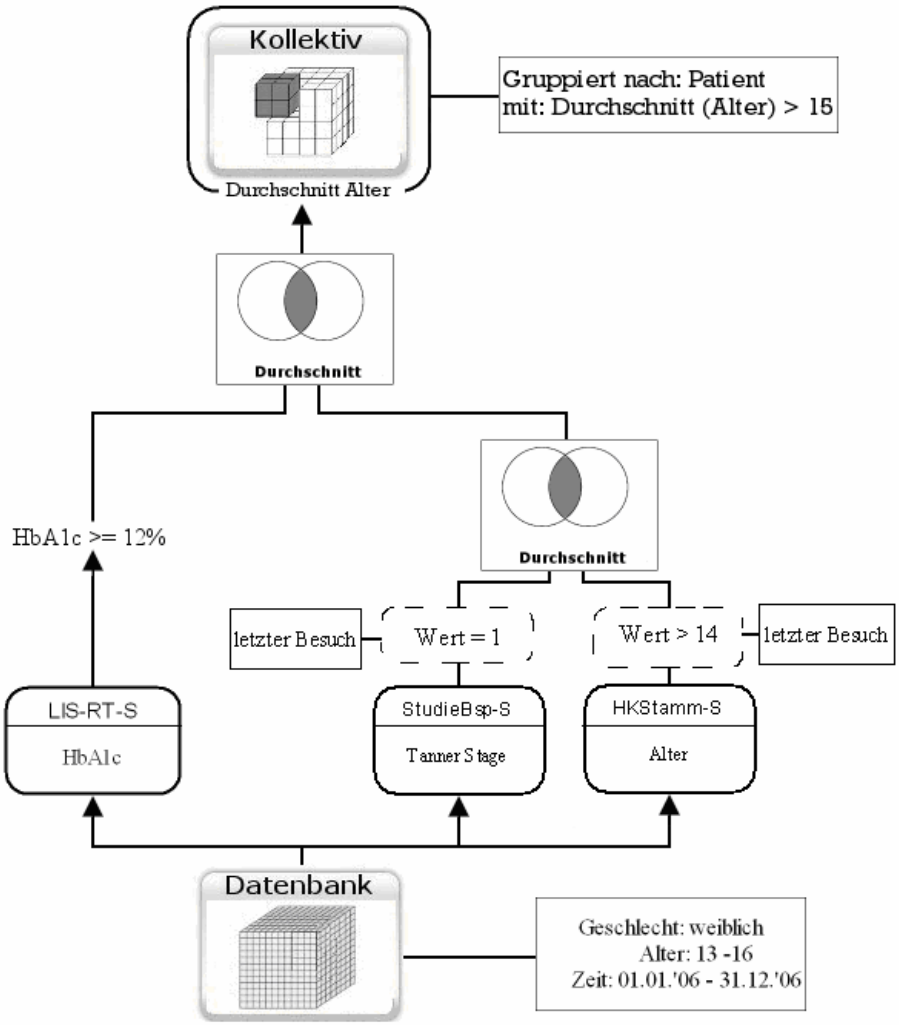


Abbildung 6.9: Ein Beispiel für eine Anwendung von Mengenausdrücken und Zugriffspfaden erweitert um die Anwendung der Gruppierung– Mädchen mit Diabetes und verzögerter Pubertät.

6.2.2.6. Zeitliche Einschränkung

Temporale Einschränkung

In VISAmed werden die zeitlichen Abhängigkeiten als Zeitpunkte, im Gegensatz zu Zeitintervallen, betrachtet (vergleiche dazu [Dorda2002], S. 90ff). Weitere Überlegungen zum Zeitoperator basieren auf den Arbeiten [Fails2005] und [Kuhn1997].

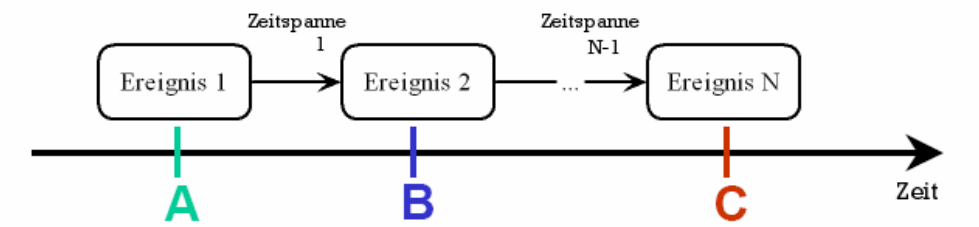


Abbildung 6.10: Veranschaulichung der Zeitpunktsemantik beim betrachten zeitlicher Abhängigkeiten. (A vor B) UND (B vor C).

Ein zeitliches Ereignis besteht aus einer Abfolge von Ereignissen und Zeitspannen zwischen den Ereignissen.

Temporaler Projektionsoperator

Der *temporale Projektionsoperator* ermöglicht die zeitliche Einschränkung aller Entities auf einen bestimmten Zeitraum und schränkt somit die Ausgabemenge auf diesen bestimmten Zeitraum ein. In VISAmed wird der temporale Projektionsoperator mit einem festen, vom Benutzer angegebenen Zeitraum — genauer mit einem Start– und Endzeitpunkt — bei der globalen Einschränkung angegeben und wirkt auf das gesamte Arbeitsdiagramm, also global.

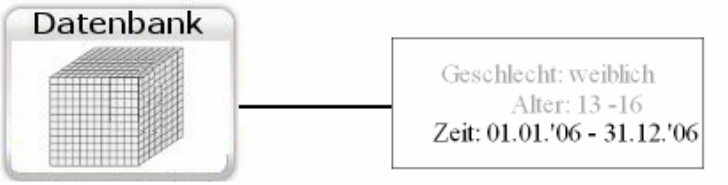


Abbildung 6.11: Der temporale Projektionsoperator schränkt die Ausgabemenge auf den angegebenen Zeitraum ein.

Temporaler Selektionsoperator

Der *temporale Selektionsoperator* wird nicht auf das gesamte Arbeitsdiagramm angewendet, sondern auf Teile davon. Mit seiner Hilfe werden Entities aufgrund zeitlicher Bedingungen an ihre Attribute ausgewählt.

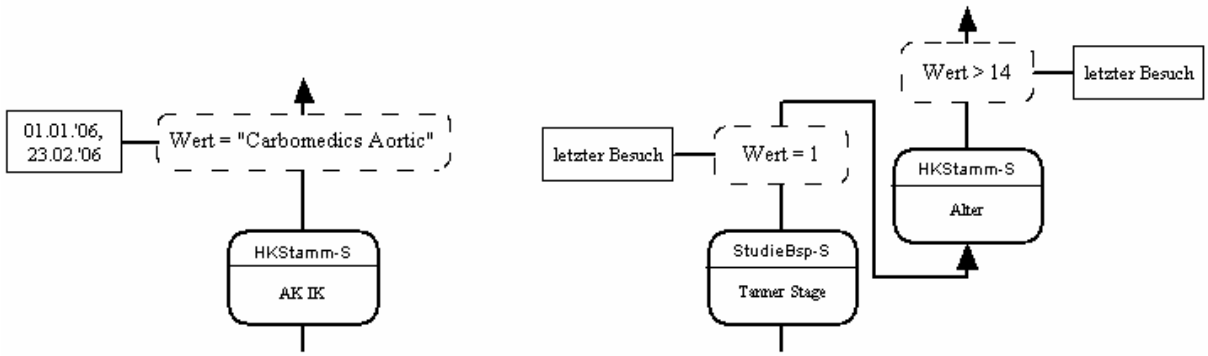


Abbildung 6.12: Der temporale Selektionsoperator schränkt die Auswahl der Entities aufgrund ihrer temporalen Bedingungen ein.

Attribute, auf die eine temporale Selektion ausgeführt werden soll, werden mit Hilfe eines gestrichelten Kastens umrandet. Eine Linie ordnet diesem Kasten die temporale Einschränkung zu.

Time–span–Operator

Mit Hilfe des *Time–span–Operators* wird ein temporales Element berechnet. Es wird mit einem oder mehreren weiteren temporalen Elementen verbunden. Graphisch besteht dieser Operator aus einem gestrichelten Kasten, der weitere

gestrichelte Kästen umfasst, sowie Kanten mit Benennung der Einheit (Jahre, Tage, Stunden, ...) und der Werte.

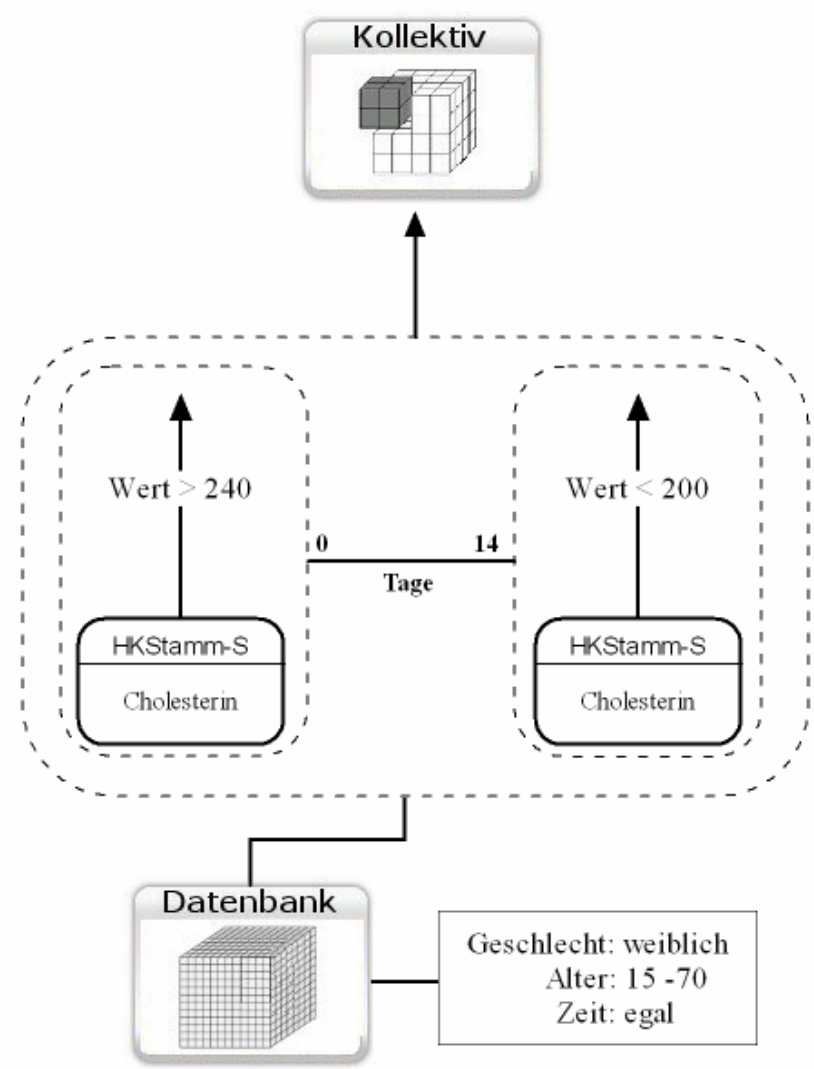


Abbildung 6.13: Zeitliche Abfrage mit variabler Zeitspanne. Es werden alle Patienten ausgegeben, deren Cholesterinwert über 240 lag, aber innerhalb von 14 Tagen unter 200 sank.

Ausprägungen des Time–span–Operator.

Abbildung 6.14 gibt eine Aufstellung der Ausprägungen des Time–span–Operator an. Für die Beschreibung der Ausprägung werden jeweils nur die ersten zwei Ereignisse betrachtet.

In Ausprägung (a) der Abbildung 6.14 tritt das Ereignis 1 in einem Zeitraum von 21 Tagen (Jahren, Minuten, ...) laufend auf.

Beispiel: Finde alle Patienten deren Cholesterinspiegel innerhalb von 21 Tagen zwischen 200 und 220 lag.

Ausprägung (b).zeigt eine Aneinanderreihung von Ereignissen ohne Zeitspanne. Von links nach rechts treten die Ereignisse eine *beliebige Zeit später* auf.

Beispiel: Finde alle Patienten bei denen es bei der Operation zu Komplikationen kam.

Ausprägung (c).beschreibt eine fixe Zeitspanne. Die Ereignisse treten zu den angegeben Zeitpunkten auf.

Beispiel: Finde alle Patienten deren Cholesterinspiegel bei 240 lag, aber 14 Tage später unter 200 gefallen ist.

Ausprägung (d) beschreibt eine variable Zeitspanne. Die Ereignisse treten im angegeben Zeitraum auf.

Beispiel: Finde alle Patienten deren Cholesterinspiegel bei 240 lag, aber innerhalb von 60 Tagen unter 200 gefallen ist.

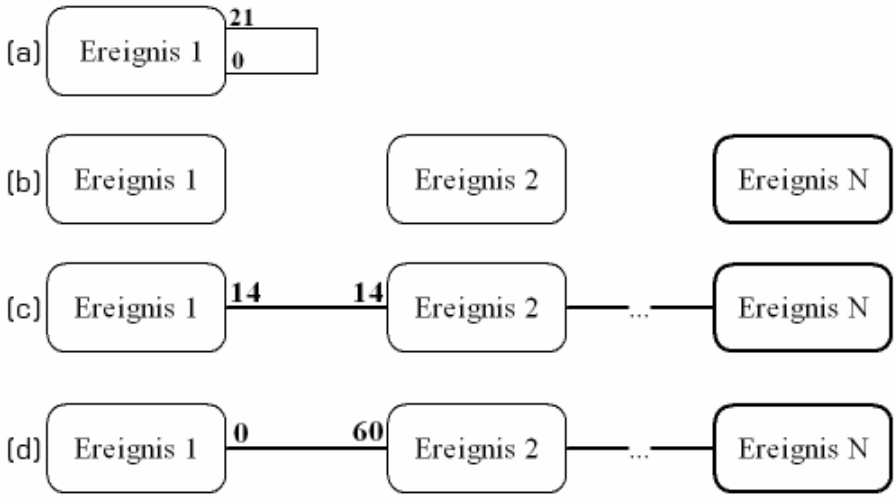


Abbildung 6.14: Ausprägungen des Time–span–Operators.

6.2.3. Visualisierung der Ausgabe

Visualisierung von Zwischenschritten und des Anfrageergebnisses.

Im Abschnitt 5.3 auf Seite 113 heißt es, dass das Vertrauen und Verständnis des Anwenders in die Ergebnisse der Datenselektion durch geeignete intuitive Interaktionswerkzeuge maßgeblich erhöht werden kann. Ein Punkt dabei ist die Verfügbarkeit geeigneter Feedback-Mechanismen. Ein Aspekt davon ist die Visualisierung von Zwischenschritten und des Anfrageergebnisses.

Anwender hat ständiges Feedback über den Anfrageprozess.

In VISAmed wird, in Anlehnung an das RPDR Query Tool [Murphy2003] eine Auswahl an Daten ständig im unteren Teil der Anfrageoberfläche eingeblendet und abhängig von den Abfragebedingungen aktualisiert. Der Anwender hat somit ein ständiges Feedback über den Anfrageprozess. Die eingeblendeten Daten sind:

- Anzahl der gefundenen Ergebnistupel. Der Anwender kann diese Zahl als Kontrolle für seine Anfrage benutzen – steigt die Anzahl der Ergebnistupel nach einer weiteren Anfrage an, muss von einem Fehler in der Anfrage ausgegangen werden.
- Geschlecht der Ergebnistupel, aufgeteilt in männliche und weibliche Patienten.
- Statistische Verteilung des Alters der Ergebnistupel in Form eines Histogramms.
- Verteilung der Patienten in lebende und verstorbene.

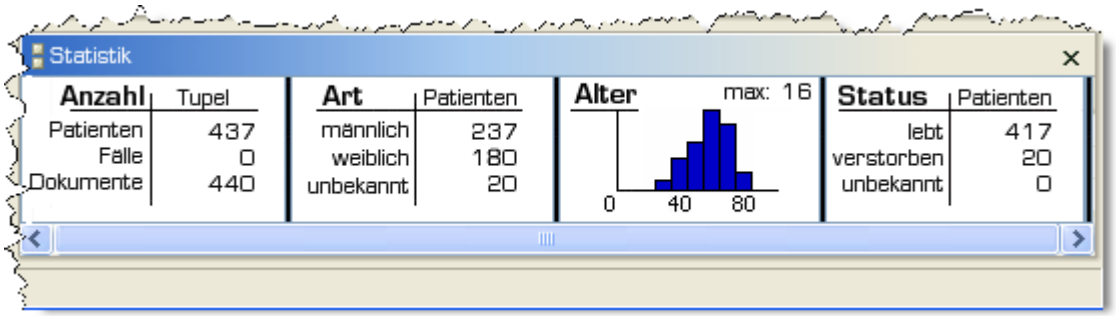


Abbildung 6.15: Die Visualisierung der Zwischen– und Endergebnisse erfolgt in VISAmed in einem eigenem Bereich.

6.3. Die Konstrukte im VISAmed Prototyp

In den vergangenen Kapiteln wurden ausführlich die Grundlagen von graphischen Sprachen, insbesondere von graphischen Anfragesprachen besprochen. Es wurden verschiedene visuelle Sprachen vorgestellt und daraus die Anforderungen für einen graphischen Anfragegenerator für medizinische Datenbanken abgeleitet. Aus den gefundenen Anforderungen wurden Konzepte ausgearbeitet und vorgestellt. In diesem Absatz werden die Konzepte in einem Prototypen umgesetzt.

Der Anwender wird mittels direkter Manipulation durch die Formulierung der Anfrage geführt.

Um die Anforderungen an den Anwender zu minimieren, wird ihm eine intuitive Benutzeroberfläche zur Verfügung gestellt (Abbildung 6.16). Der Anwender wird mittels direkter Manipulation durch die Formulierung der Anfrage geführt. Die Befehle für die Anfragegenerierung werden nicht über Kommandos, wie etwa Eintippen von Befehlen oder Ausfüllen von Dialogboxen, sondern durch direktes Anklicken der entsprechenden Werkzeuge ausgeführt. Die direkte Manipulation erlaubt so ein natürlicheres und einfacheres Arbeiten

Die Anfrageformulierung erfolgt, wie bereits erwähnt, in mehrere Phasen. VISAmed startet mit einem Datenüber–

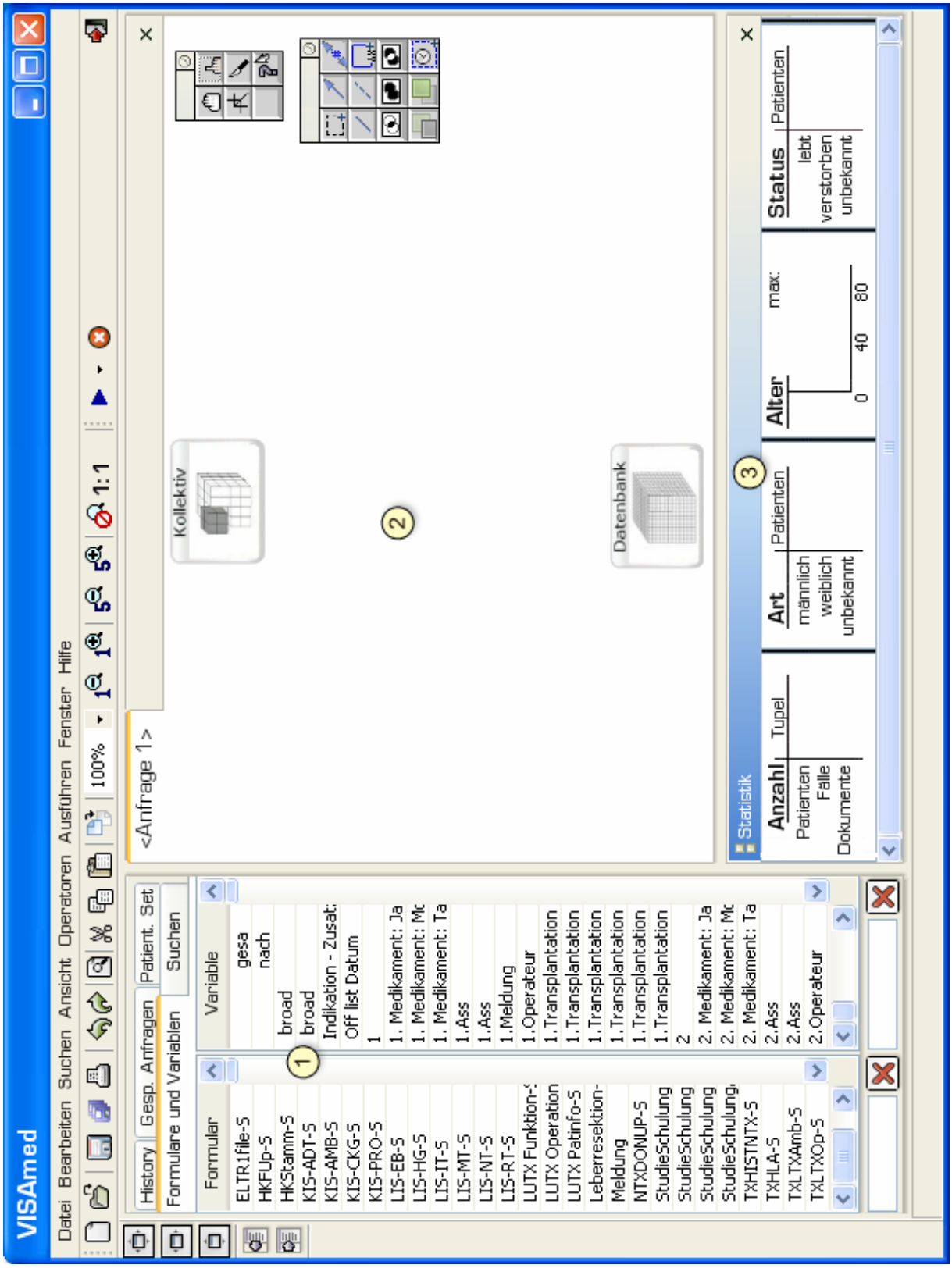


Abbildung 6.16: VISAmed Tool: die Benutzeroberfläche mit den interaktiven Bedienungselementen. Sie vereinigt die Attribute Menübasiertheit, Iconbasiertheit, Browserfunktionalität und Diagrammbasiertheit.

Datenüberblick

blick, damit sich der Anwender zunächst in der Datenbank orientieren kann. In VISAmed erfolgt dieser Datenüberblick mit dem Darstellen von Formularen und Variablen im linken Teil der Anfrageoberfläche (Punkt 1 in Abbildung 6.16). Mit Hilfe dieses Datenüberblicks wählt er die für die Anfrage relevanten Daten aus. Dabei werden die interessierenden Variablen einfach in das rechte Arbeitsfenster gezogen (Punkt 2 in Abbildung 6.16).

Datenüberblick bleibt erhalten

Der Überblick über die ausgewählten Daten bleibt immer erhalten – die ausgewählten Daten werden in weiteren Fenstern unterhalb des Arbeitsfensters visualisiert. Die ausgegebenen Werte werden automatisch aktualisiert, sobald die Anfrage durchgeführt wird. (Punkt 3 in Abbildung 6.16)

Interaktionsmöglichkeiten

An die selektierten Daten können verschiedene Bedingungen gestellt werden. Abhängig davon um welches Konstrukt es sich handelt, stehen verschiedene Interaktionsmöglichkeiten zur Verfügung.

Im Folgenden sind die einzelnen Konstrukte aufgeführt. Ein Klick mit der Maus bringt die abgebildete Toolbox zum Vorschein, von der aus, ganz im Sinne einer direkten Manipulation, die zur Verfügung gestellten Funktionen ausgeführt werden können.

Datenbank – Widget[39]

Datenbank – Widget

Das Datenbankwidget symbolisiert den gesamten Datenbestand. Neben den Zugriffspfaden ist die *globale Einschränkung* die wichtigste Funktion die auf den gesamten Datenbestand definiert werden kann.

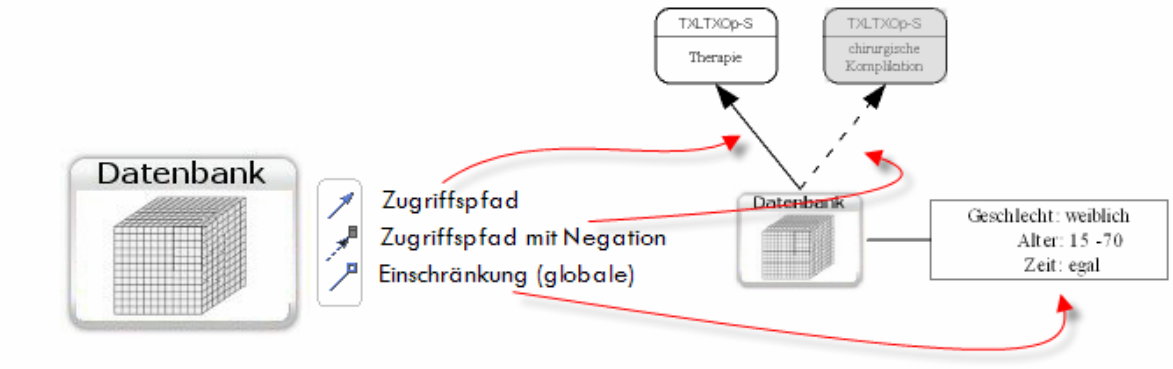


Abbildung 6.17: Widget für den gesamten Datenbestand. Es stehen drei Funktionen zur Verfügung.

[39] Widget: graphisches Symbol das eine Interaktion zwischen Computer und Benutzer erlaubt (Knopf usw.).

Variable – Widget

Variable – Widget

Das Widget für die Variable symbolisiert die ausgewählte Variable. Neben den Zugriffspfaden stehen noch Join und lokale Einschränkung zur Verfügung. Weitere Funktionen auf Variablen – Klammerungen, Aggregatfunktionen und temporale Einschränkungen – können aus der Tool–Palette gewählt werden.

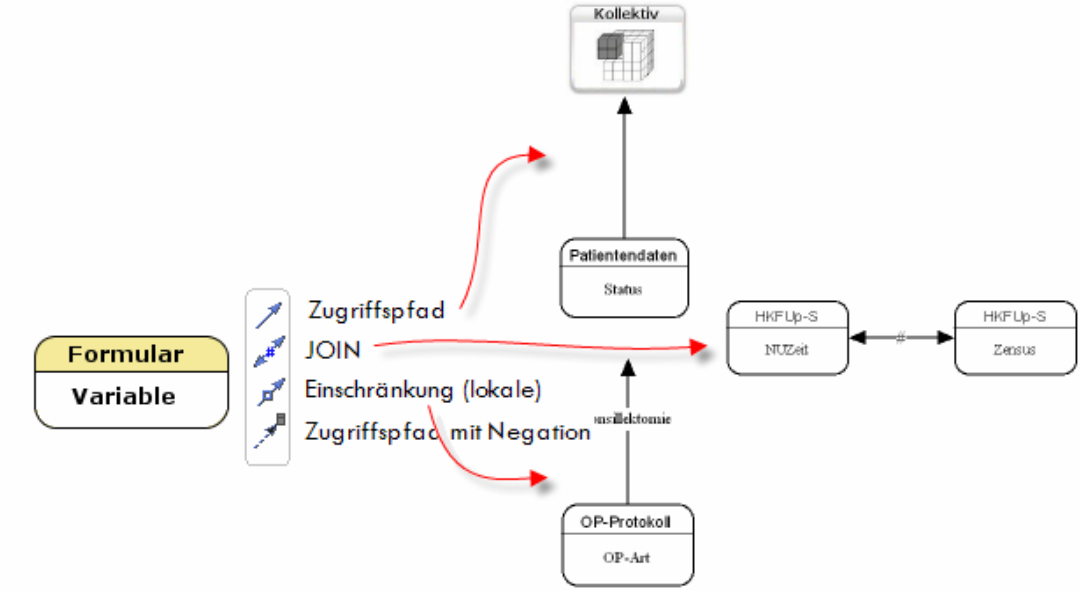


Abbildung 6.18: Widget für die interessierende Variable. Es stehen direkt vier Funktionen zur Verfügung. Andere Funktionen können über die Tool–Palette gewählt werden.

Gespeicherte Anfragen – Widget

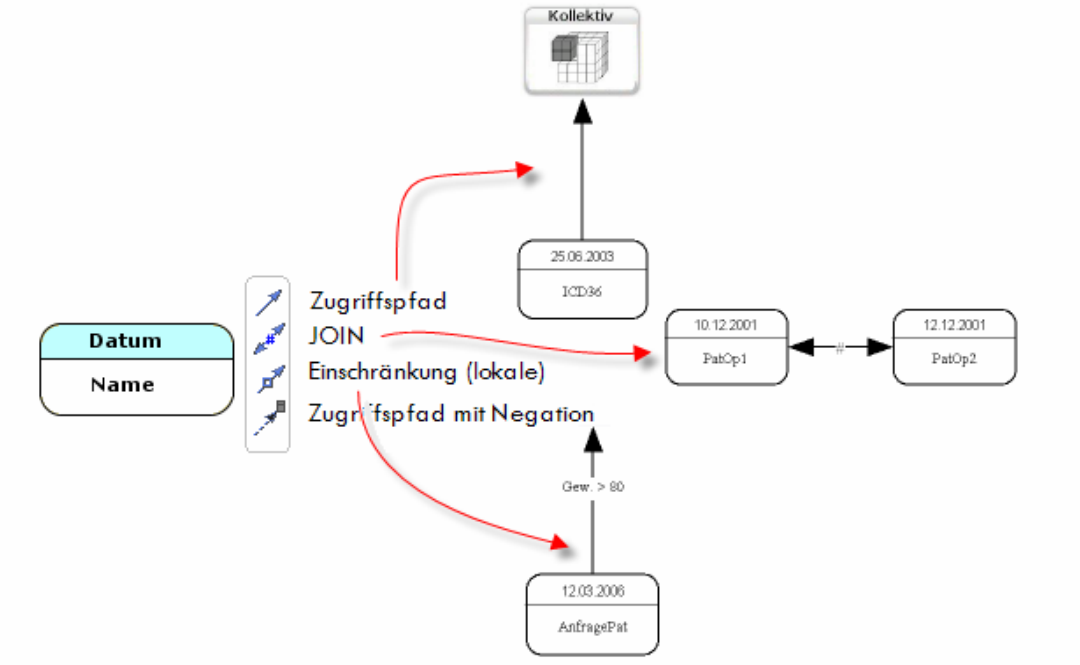


Abbildung 6.19: Widget für gespeicherte. Anfragen. Sie können wie Variablen verwendet werden.

Patienten Sets – Widget

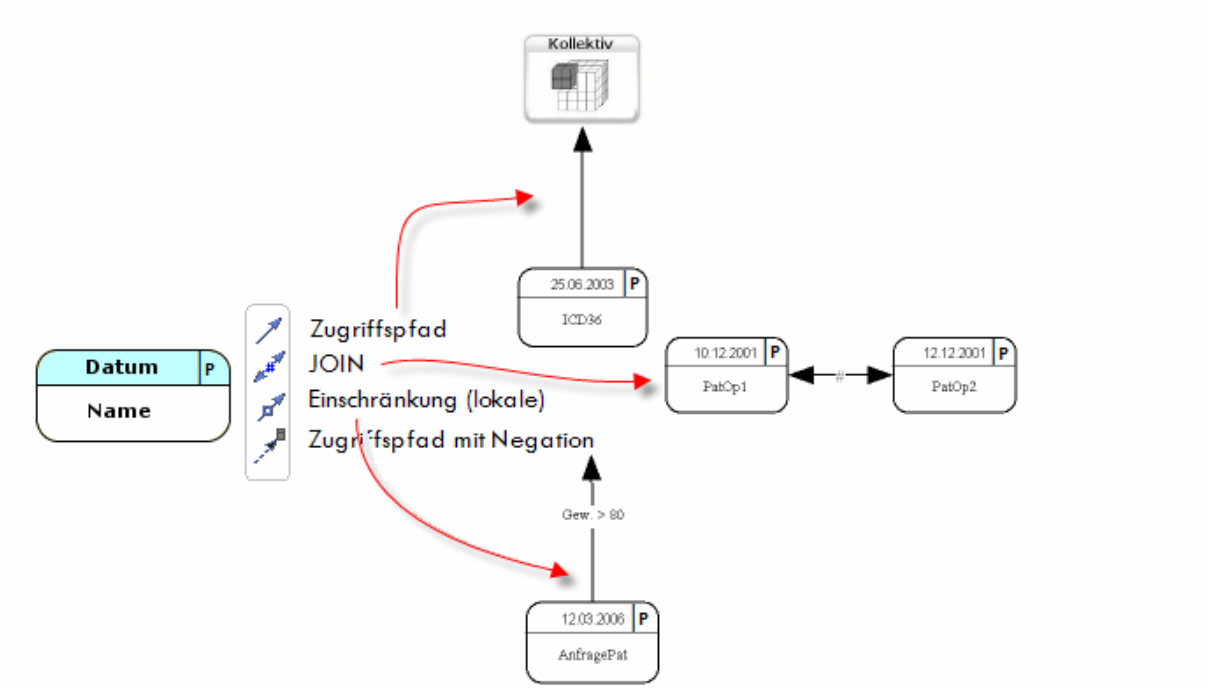


Abbildung 6.20: Widget für Patienten Sets. Sie können wie Variablen verwendet werden.

Für die Widgets, gespeicherte Anfrage und Patienten Sets, gilt das Gleiche wie für das Variable – Widget.

Mengenausdruck – Widget

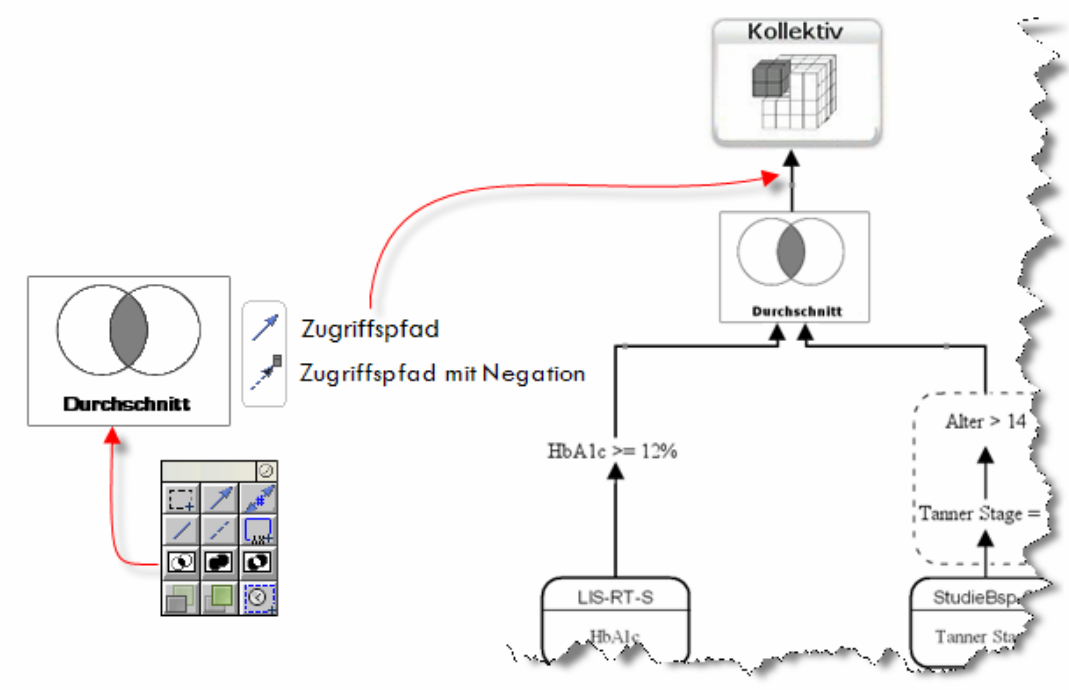


Abbildung 6.21: Widget für Mengenausdrücke mit den zwei möglichen Zugriffspfaden.

Vom Widget für Mengen (Durchschnitt, Vereinigung, Differenz) können nur Pfade zu anderen Widgets gelegt werden. Einschränkungen sind nicht möglich.

Klammer- und Zeitklammer – Widget

Klammer – Widget

Ein Klammer umschließt mehrere Objekte, dabei können Klammern, Klammern umschließen aber nicht überschneiden.

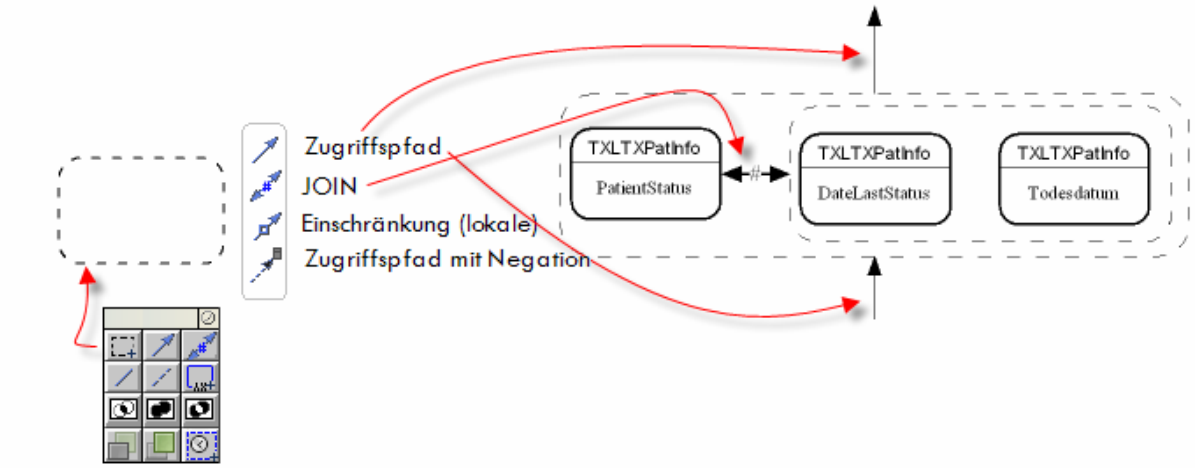


Abbildung 6.22: Widget für Klammerungen mit dem Symbol in der Tool–Patelle.

Zeitklammer – Widget

Temporale Bedingungen werden ebenfalls durch Klammerung ausgedrückt.

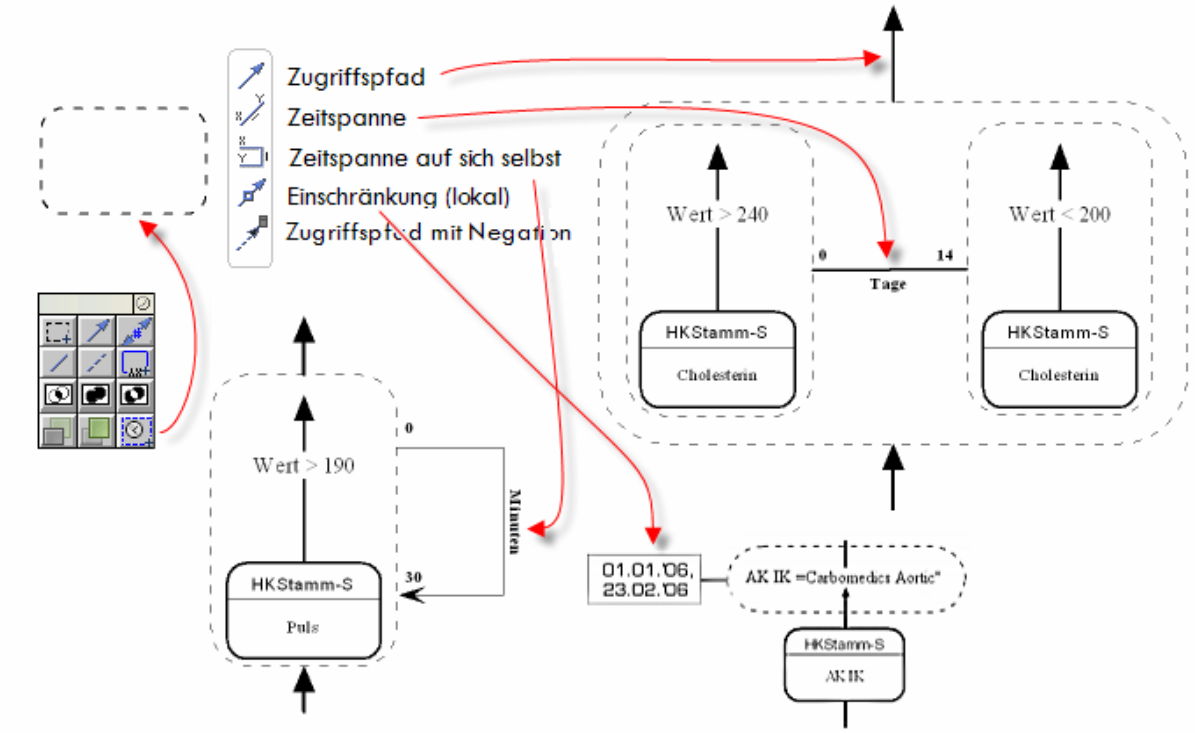


Abbildung 6.23: Widget für temporale Klammerungen mit dem Symbol in der Tool–Patelle.

Pie Menue

Die beschriebenen Widgets verwenden für die Werkzeugleiste sog. Pie Menues. Pie Menues wurden von Don Hopkins entwickelt und in [Callahan1988] im Vergleich zu linearen Menüs diskutiert. Im Allgemeinen ist die Auswahl mit Pie Menues schneller und zuverlässiger als mit linearen Menüs. Auch verschachtelte Pie Menues sind intuitiver zu bedienen als lineare Menüs, können aber mit diesen kombiniert werden. Pie Menues werden nur bei Bedarf angezeigt und führen so zu geringerer Ablenkung des Benutzers bzw. zu geringerer Überladung der Benutzeroberfläche durch ständig angezeigte Werkzeug– und Menüleisten.

Aggregatfunktion – Widget

Eine Aggregatfunktion kann nur auf das Ergebnis angewendet werden. In Verbindung mit Aggregatsfunktionen können die Klauseln GROUP BY, zur Gruppierung von Ergebnissen, sowie HAVING zur Filterung der gruppierten Ergebnisse verwendet werden

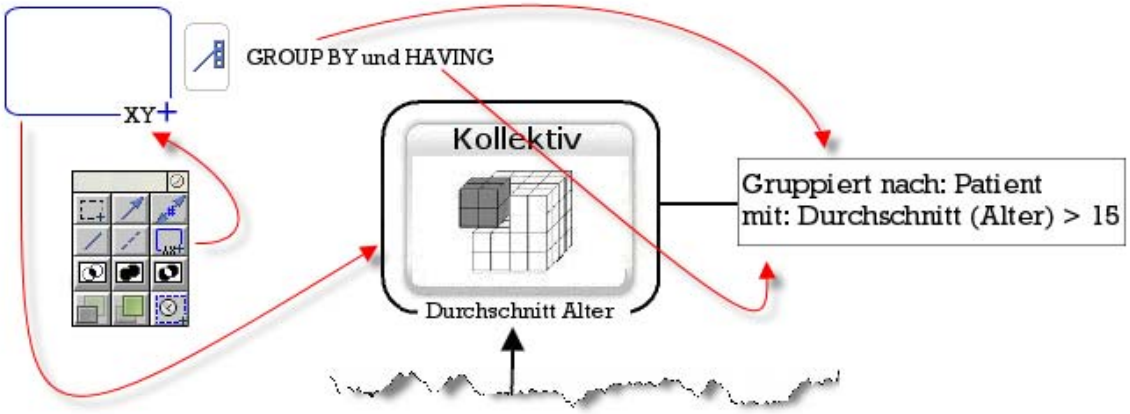


Abbildung 6.24: Widget für Aggregatfunktionen mit der möglichen einschränkenden Funktion.

Kollektiv – Widget

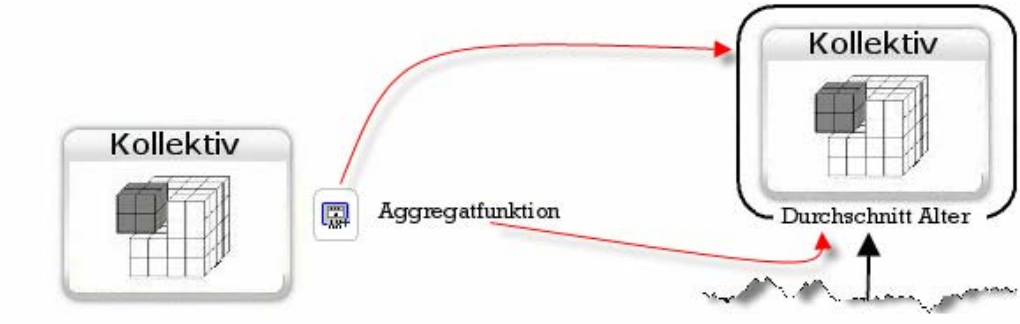


Abbildung 6.25: Widget für das Ergebnis der Anfrage. Es steht eine Funktion zur Verfügung.

Das Kollektiv–Widget symbolisiert das Ergebnis der Anfrage. Auf das Ergebnis können nur noch Aggregatfunktionen definiert werden.

Tool Palette

Tool Palette

Die Tool Paletten sind immer auf der Arbeitsoberfläche vorhanden. Die zusätzlichen Befehle für die Anfragegenerierung werden durch direktes Anklicken der entsprechenden Werkzeuge ausgeführt. Die direkte Manipulation erlaubt so ein natürlicheres und einfacheres Arbeiten

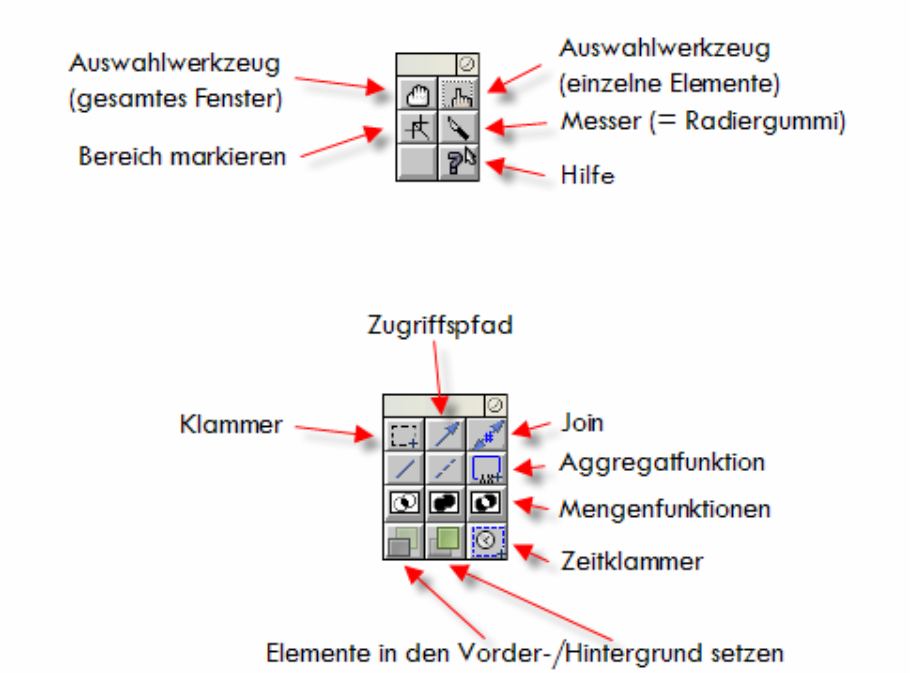


Abbildung 6.26: Tool Palette

Nachdem die Anfrage mit Hilfe der beschriebenen Widgets durchgeführt worden ist, erfolgt die Datenausgabe. Für den Anwender sind Datenüberblick, Anfrage und Anfrageergebnis immer gleichzeitig sichtbar.

6.4. Fehlerquellen bei ungeübten Anwendern

Fehlerklassen

Abgewandelt nach [Aho1986], wo aus der Sicht des Compilerbaus Fehler aufgelistet sind, werden hier einige Fehlerklassen in VISAmed betrachtet.

- Lexikalische Fehler entstehen, wenn ungültige Zeichen verwendet werden.

- Syntaktische Fehler entstehen, wenn syntaktische Elemente falsch angeordnet werden.
- Semantikfehler entstehen, wenn zwar die Syntax der Anfrage richtig ist, die semantische Bedeutung jedoch nicht der beabsichtigten entspricht.
- Logische Fehler entstehen, wenn durch fehlerhafte Verwendung logischer Konstrukte, die Anfrage unerwünschte bzw. falsche Ergebnisse liefert.

Lexikalische Fehler

Lexikalische Fehler sind in VISAmed nicht möglich. Die textuelle Repräsentation der Anfragen wird vom System automatisch generiert. Der Anwender kann nur die vom System angebotenen Bezeichner wählen.

Syntaktische Fehler

Syntaktische Fehler sind in VISAmed nicht möglich. Der Anwender wird mittels direkter Manipulation bei der Anfrageformulierung unterstützt. Die Wörter der Anfragesprache SQL werden vom System selbst generiert und können nicht an falschen Stellen gesetzt oder falsch verwendet werden. Die richtige Reihenfolge von Operatoren und Operanden wird vom VISAmed erzwungen und kann nicht zu Fehlern führen.

Semantikfehler

Bei Semantikfehlern handelt es sich um einen schwierigeren Fehlertyp, da die textuelle Repräsentation der Anfragen, die vom System automatisch generiert wird, nicht gegen die Regeln der textuellen Anfragesprache verstößt. Die graphische Darstellung der Anfrage in VISAmed kann helfen Semantikfehler zu minimieren, verhindern lassen sie sich dadurch nicht.

Logische Fehler

Logische Fehler in der Anfrage können nicht wirkungsvoll eliminiert werden, wie auch eine Studie in [Murphy2003] zeigt. Diese Fehler sind auf die fehlerhafte Verwendung der logischen Operatoren, wie AND, OR, NOT, zurückzuführen. Ihre, von der booleschen Logik abweichende, umgangssprachliche Verwendung führt zu Verwirrungen – das umgangssprachliche UND entspricht dem booleschen ODER.

Einschätzung der Fehlervermeidung.

Mit VISAmed können lexikalische und syntaktische Fehler vollkommen verhindert und semantische Fehler minimiert werden. Logische Fehler bleiben ein Problem, können aber durch geeignete graphische Konstrukte möglichst gut abgefangen werden.

6.5. Zusammenfassung

Dieses Kapitel ging auf die Architektur von VISAmed ein, ohne jedoch alle Details auszuarbeiten.

Verschiedene visuelle Konzepte wurden eingeführt um die strukturellen Anforderungen an den Anwender zu mini–

mieren, d.h. die Forderung nach Kenntnis der Sprachsyntax zu verringern. Der Anwender wird mittels direkter Manipulation durch die Formulierung der Anfragebedingungen geführt. Er erstellt die Anfrage intuitiv durch Interaktionen, wie etwa durch Selektion verschiedener Symbole.

Die Grundidee hinter VISAmed ist die "filter flow" Metapher. Nach diesem Modell werden aus einer Menge von Informationen nach und nach jene Informationen herausgefiltert, die für das Ergebnis der Anfrage relevant sind, indem die Information UND bzw. ODER verknüpft wird.

In medizinischen Datenbanken spielen temporale Bedingungen eine wichtige Rolle und es ist oft wünschenswert die Ausgabe auf einen bestimmten Zeitraum einzuschränken. Zur Konstruktion solcher temporaler Anfragen wurden neue Anfragekonstrukte betrachtet.

Zur Veranschaulichung der theoretisch besprochenen Konstrukte, wurde die Visualisierung der interaktiven Elemente, entsprechend den erarbeiteten Grundlagen aus den vorangegangenen Kapiteln, an Hand der Benutzeroberfläche eines Prototyps betrachtet.

KAPITEL

7

Bewertung und Ausblick

Der wahre Preis einer Sache … ist die Mühe und Plage, ihn zu erarbeiten.

Adam Smith

In dieser Arbeit wurde, mit der Ausarbeitung der Anforderungen und Vorstellung von Realisierungsmöglichkeiten im Rahmen des Anfragegenerators VISAmed, ein Ansatz zur graphischen Analyse multidimensionaler medizinischer Daten vorgestellt. Die schrittweise Spezifikation und Manipulation der Anfrage steht im Vordergrund der Datenanalyse. Die Grundphilosophie des hier verfolgten Ansatzes zielt auf eine Minimierung der linguistischen und strukturellen Anforderungen an den Anwender. Das System soll ohne aufwendige Einarbeitung nutzbar sein und durch seine Einfachheit den Anwender zu seiner Nutzung anregen.

Die wesentlichen Inhalte und Ergebnisse dieser Arbeit werden zunächst kurz zusammengefasst, anschließend werden Anknüpfungspunkte für mögliche Anschlussarbeiten genannt.

7.1. Ergebnisse dieser Arbeit

Die vorliegende Arbeit zeigt, dass der Entwurf und die Realisierung eines neuen Anfragegenerators ein aufwendiges und ambitioniertes Vorhaben ist. Im Rahmen einer Magisterarbeit kann das Vorhaben nicht in vollem Umfang geleistet werden. Dennoch konnten im Verlauf der Arbeit, aus–

gehend von den wesentlichen Anforderungen an einen Anfragegenerator und die durch ihn gegebene graphische Anfragesprache, die visuellen Konzepte mit den graphischen Konstrukten der Anfragesprache definiert werden.

Im Zusammenhang mit den Anforderungen und als Ergebnis eines umfangreichen Literaturstudiums wurden zu Beginn dieser Arbeit wichtige Begriffe festgelegt sowie die Merkmale von Anfragesprachen beschrieben. Besondere Beachtung fanden dabei ausgewählte visuelle Sprachen, deren Eigenschaften später in die Überlegungen zum neuen Anfragegenerator einflossen. In dieser Arbeit wurde zwischen allgemeinen Anforderungen und medizinspezifischen Anforderungen an das System unterschieden. Die allgemeinen Anforderungen sind für jede Art von Anfragesprache gültig. Aus verschiedenen Veröffentlichungen wurde eine umfangreiche Liste von allgemeinen Anforderungen erstellt. Erweiterte Anforderungen, die sich aufgrund der Eigenschaften von medizinischen Daten ergeben, wurden unter den medizinspezifischen Anforderungen zusammengefasst. Spezifische Kriterien von Multimediasystemen erwiesen sich dabei zum Teil kompatibel zu Anforderungen der medizinischen Domäne, wurden aber natürlich noch erweitert. In diesem Zusammenhang wurde auch das bestehende System Archimed, als Basis für das neue System, genau betrachtet.

Aus den diskutierten Anforderungen ging in der vorliegenden Arbeit der graphische Anfragegenerator VISAmed hervor. VISAmed ermöglicht besonders ungeübten Anwendern die selbständige Formulierung von Anfragen an die Datenbank. Die Anforderungen an den Anwender werden durch die durchgehende Benutzung der direkten Manipulation und der dynamischen Anfragegenerierung, sowohl im linguistischen wie auch im strukturellen Bereich minimal gehalten.

Eine Besonderheit von VISAmed ist die Unterstützung von temporalen Anfragen. Medizinische Daten können meist nur unter Berücksichtigung ihrer zeitlichen Zusammenhänge sinnvoll betrachtet werden. Der Schwerpunkt bei dieser Unterstützung liegt in der Kombination von temporalen und nicht temporalen Daten. VISAmed stellt somit eine graphische temporale Anfragesprache dar.

Als eine weitere wichtige Aufgabe von Anfragesprachen für medizinische Datenbanken, stellt sich die Unterstützung der Präsentation der Anfrageergebnisse heraus. Das Vertrauen und Verständnis des Anwenders in die Ergebnisse der Datenselektion kann durch die Visualisierung von Zwischenergebnissen und des Anfrageergebnisses erhöht werden.

Aufbauend auf der Beschreibung der aufgestellten Kriterien wurden visuelle Konzepte für die Benutzeroberfläche und die interaktiven Elemente, sowie die Sprachkonstrukte vorgestellt.

In dieser Magisterarbeit konnte gezeigt werden, dass visuelle Sprachen gegenüber textbasierten Sprachen einige große Vorteile mit sich bringen. Neben der einfacheren, intuitiven Anwendung, basierend auf den guten visuellen Ausdruckselementen, darf aber nicht vernachlässigt werden, dass die verwendeten graphischen Objekte sehr schnell ein komplexes Verhalten zeigen und mitunter mental schwer fassbar werden.

Abschließend ist zu bemerken, dass der Anfragegenerator VISAmed den Versuch zu einer neuen graphischen Anfragesprache darstellt, die sich aus mehreren graphischen Sprachen zusammensetzt, jedoch in dieser Kombination noch keine Verwendung gefunden hatte.

7.2. Ausblick

Die vollständige Implementierung eines einsetzbaren graphischen Anfragegenerators war nicht Gegenstand dieser Arbeit. Eine entsprechende Implementierung und Evaluierung der Konzepte muss als Aufgabe zukünftiger Arbeiten angesehen werden.

Die Kapitel 5 und Kapitel 6 liefern Anknüpfungspunkte für weiterführende Arbeiten. Neben der Konkretisierung betrachteter Komponenten, sind insbesondere Erweiterungen zur Visualisierung der Daten zu nennen. Saake in [Saake2000] sagt, dass Visualisierungstechniken bei der Erforschung der Daten helfen können, indem sie

- eine interaktive Datenerforschung mit Hilfe einer direkten Visualisierung der Daten ermöglichen
- Data Mining Verfahren durch ein direktes Feedback unterstützen, und
- Techniken für eine effektive Interaktion mit den Daten zur Verfügung stellen.

Ansprechende Visualisierungstechniken für Daten sind im [Aduna2006] oder [Ham2002] beschrieben.

Als weitere Erweiterungsmöglichkeiten seien genannt

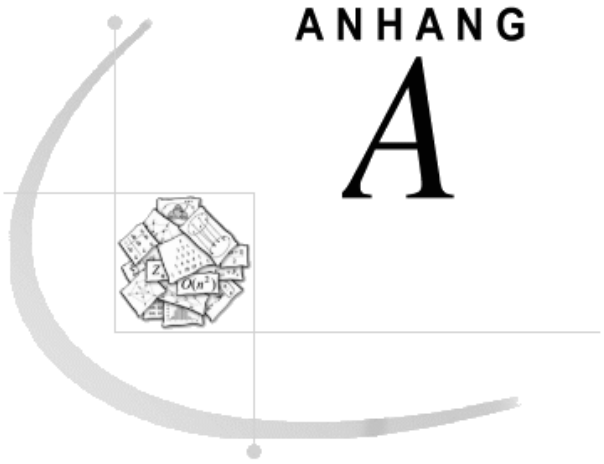
- eine genauere Modellierung von graphischen Komponenten, insbesondere eine bessere Auswahl von geeigneten Präsentations–, Arbeitsmittel– und Architekturmetaphern um das mentale Modell der Nutzer besser zu unterstützen.

- eine verfeinerte Überprüfung der Gültigkeit der Benutzereingaben und differenzierte Unterstützung bei der Anfrageformulierung, um die semantischen und logischen Fehler möglichst gering zu halten.
- Einführung von gewichteten Anfragen, damit zusätzliche Präferenzen des Benutzers abgebildet werden können.
- Einführung von unscharfen Anfragen. Im gleichen Zusammenhang müssen die Mengenoperationen um Konzepte zur Behandlung von Fuzzy–Mengen erweitert werden.

7.3. Fazit

Wie im Vorwort bereits erwähnt war letztendlich das Ziel dieser Arbeit, eine Benutzerschnittstelle für das System Archimed, zur komfortablen, flexiblen und interaktiven Bildung komplexer Kollektive multidimensionaler Daten in klinischen Datenbanken zu diskutieren. Es wird jedoch weiterer Forschung und findiger Ideen bedürfen um sich an eine optimale Lösung für die Bedürfnisse der forschenden Mediziner heranzuarbeiten. Denn – um mit einem Zitat von Hand aus ([Hand1997], S. 78) zu schließen:

> *It is from real problems that the important solutions emerge, and it is by solving real problems that one changes the world.*



ANHANG

A

Literatur-verzeichnis

Manche Bücher darf man nur kosten,
andere muß man verschlingen
und nur wenige kauen und verdauen...

Sir Francis Bacon

Aduna (2006),'Aduna Cluster Map Library version 2006.1', Aduna, http://www.aduna-software.com, Review: 30.01.2007. *Aduna2006*

AgentSheets (2007),'AgentSheets', http://www.agentsheets.com/index.html, Review: 30.01.2007. *AgentSheets2007*

Aho, A.V.; Sethi, R. & Ullman, J.D. (1986), *Compilers : principles, techniques, and tools*, Addison-Wesley. *Aho1986*

Albertoni, R.; Bertone, A. & De Martino, M. (2005),Semantic analysis of categorical metadata to search for geographic information, *in* 'Database and Expert Systems Applications, 2005. Proceedings. Sixteenth International Workshop on', pp. 453--457. *Albertoni2005*

Alessio, R.A. (2004),'Semantic Web and Information Visualization', citeseer.ist.psu.edu/759633.html, Review: 30.01.2007. *Alessio2004*

Allen, J.F. (1983), 'Maintaining knowledge about temporal intervals', *Commun. ACM* **26**(11), 832--843. *Allen1983*

Anderson, J.R. (2001), *Kognitive Psychologie, 3. Aufl.*, Spektrum Akademischer Verlag. *Anderson2001*

Appelrath1991 Appelrath, H., ed. (1991), Datenbanksysteme in Büro, Technik und Wissenschaft, GI-Fachtagung, Kaiserslautern, 6.-8. März 1991, Proceedings, Vol. 270, Springer.

Badre1996 Badre, A.N.; Catarci, T.; Massari, A. & Santucci, G. (1996),'Comparative Ease of Use of a Diagrammatic Vs. an Iconic Query Language', citeseer.ist.psu.edu/222510.html, Review: 30.01.2007.

Baeza-Yates1999 Baeza-Yates, R. & Ribeiro-Neto, B. (1999), Modern Information Retrieval, Addison Wesley.

Balkir1996 Balkir, N.H.; Sukan, E.; Ozsoyoglu, G. & Ozsoyoglu, Z.M. (1996),VISUAL: A Graphical Icon-Based Query Language, in 'ICDE', pp. 524-533.

Banhart1995 Banhart, F. & Klaeren, H. (1995), 'A graphical query generator for clinical research databases', Methods Inf Med **34**(4), 328--339.

Batini1991 Batini, C.; Catarci, T.; Costabile, M.F. & Levialdi, S. (1991),Visual Query Systems: A Taxonomy, in ' KNUTH, E. (Hrsg.) ; WEGNER, L.M. (Hrsg.): Visual Database Systems (VDB) II. Proceedings of the IFIP TC2/WG 2.6 Second Working Conference on Visual Database Systems Bd. A-7', pp. S. 153-168.

Becker1999 Becker, P. (1999),'Einsatz der Formalen Begriffsanalyse zur Dokumentnavigation', Master's thesis, Philipps Universität Marburg.

Bederson2001 Bederson, B.B. (2001),PhotoMesa: a zoomable image browser using quantum treemaps and bubblemaps, in 'UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology', ACM Press, New York, NY, USA, pp. 71--80.

Beesten1995 Beesten, A. (1995),'Eine visuelle Anfragesprache zur Einbettung von ‘‘ad-hoc’’-Anfragen in CARLOS', Master's thesis, Universität Oldenburg.

Berger2003 Berger, S. (2003),'Conception Of A Graphical Interface For Querying XML', Master's thesis, Ludwig-Maximilians-Universität München.

Berghold2003 Berghold, A. (2005),'Tätigkeitsbericht (2004/2005)Indtitut für medizinische Informatik, Statistik und Dokumentation', http://www.meduni-graz.at/imi/de/praesentation/Jahresbericht_2004_2005.pdf, Review: 30.01.2007.

Blackwell2001 Blackwell, A.F.; Whitley, K.N.; Good, J. & Petre, M. (2001), 'Cognitive Factors in Programming with Diagrams', *Artif. Intell. Rev.* **15**(1-2), 95--114.

Brandt, C.A.; Morse, R.; Matthews, K.; Sun, K.; Deshpande, A.M.; Gadagkar, R.; Cohen, D.B.; Miller, P.L. & Nadkarni, P.M. (2002), 'Metadata-driven creation of data marts from an EAV-modeled clinical research database.', *Int J Med Inform* **65**(3), 225--241. *Brandt2002*

Brundege, J.M. & Dubay, C. (2003), 'BioQuery: an object framework for building queries to biomedical databases.', *Bioinformatics* **19**(7), 901--902. *Brundege2003*

Callahan, J.; Hopkins, D.; Weiser, M. & Shneiderman, B. (1988),An empirical comparison of pie vs. linear menus, *in* 'CHI '88: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 95--100. *Callahan1988*

Card, S.K.; Mackinlay, J.D. & Shneiderman, B. (1999),Using vision to think'Readings in information visualization: using vision to think', Morgan Kaufmann Publishers Inc., , pp. 579--581. *Callahan1988*

de Carvalho, T. & Edelweiss, N. (1997),A visual query system implementing a temporal object-oriented model with roles on a relational database, *in* 'Computer Science Society, 1997. Proceedings., XVII International Conference of the Chilean', pp. 38--47. *Carvalho1997*

Catarci, T. (2000), 'What Happened When Database Researchers Met Usability', *Information Systems* **25**(3), 177-212. *Catarci2000*

Catarci, T.; Costabile, M.F.; Levialdi, S. & Batini, C. (1997), 'Visual Query Systems for Databases: A Survey', *Journal of Visual Languages and Computing* **8**(2), 215-260. *Catarci1997*

Catarci, T.; Mascio, T.D.; Franconi, E.; Santucci, G. & Tessaris, S. (2003),'An ontology based visual tool for query formulation support', citeseer.ist.psu.edu/catarci04ontology.html, Review: 30.01.2007. *Catarci2003*

Catarci, T.; Santucci, G. & Angelaccio, M. (1993), 'Fundamental Graphical Primitives for Visual Query Languages', *Information Systems* **18**(2), 75-98. *Catarci1993*

Chavda, M. & Wood, P.T. (1997),Towards an ODMG-Compliant Visual Object Query Language, *in* 'The VLDB Journal', pp. 456-465. *Chavda1997*

Chavda, M. & Wood, P.T. (1997),'A Visual Query Language For ODMG-Compliant Databases', http://www.dcs.kcl.ac.uk/staff/ptw/tkde.ps.gz., Preliminary version of sections of this paper appeared in Proceedings of the 23rd International Conference on Very Large Data Bases (Athens, Greece, Aug. 25–29), 1997, pp. 456–465,. Review: 30.01.2007. *Chavda1997a*

Checkoway2004 Checkoway, H.; Pearce, N. & Kriebel, D. (2004), *Research Methods in Occupational Epidemiology*, Oxford University Press, New York.

Chen2000 Chen, R.S.; Nadkarni, P.; Marenco, L.; Levin, F.; Erdos, J. & Miller, P.L. (2000), 'Exploring performance issues for a clinical database organized using an entity-attribute-value representation.', *J Am Med Inform Assoc* **7**(5), 475--487.

Chittaro2002 Chittaro, L.; Combi, C. & Trapasso, G. (2002),'Visual Data Mining of Clinical Databases: An Application To the hemodialytic treatment based on 3D Interactive Bar Charts'.

Chung1999 Chung, V.W.H. (1999),'3DComposer - A visual builder for 3D notations', Master's thesis, University of Auckland, New Zealand.

Citrin1995 Citrin, W.; Doherty, M. & Zorn, B. (1995),Design of a Completely Visual Object-Oriented Programming Language, *in* M. Burnett; A. Goldberg & T. Lewis, ed.,'Visual Object-Oriented Programming', Prentice-Hall, New York.

LabView2007 Corporation, N.I. (2007),'LabVIEW', http://www.ni.com/, Review: 30.01.2007.

Corwin2007 Corwin, J.; Silberschatz, A.; Miller, P.L. & Marenco, L. (2007), 'Dynamic tables: an architecture for managing evolving, heterogeneous biomedical data in relational database management systems.', *J Am Med Inform Assoc* **14**(1), 86--93.

Cramer2005 Cramer, B. (2005),'Generierung von graphischen Strukturediteren aus visuellen Spezifikationen', Master's thesis, Universität Paderborn.

Dachselt2004 Dachselt, R. (2004), *Eine deklarative Komponentenarchitektur und Interaktionsbausteine für dreidimensionale multimediale Anwendungen*, Der Andere Verlag.

Davies1997 Davies, R.A. (1997),'A Metatool for Visual Language Development', Master's thesis, National University of Ireland Maynooth Co. Kildare.

Deshpande2003 Deshpande, A.M.; Brandt, C. & Nadkarni, P.M. (2003), 'Temporal query of attribute-value patient data: utilizing the constraints of clinical studies.', *Int J Med Inform* **70**(1), 59--77.

Deshpande2002 Deshpande, A.M.; Brandt, C. & Nadkarni, P.M. (2002), 'Metadata-driven ad hoc query of patient data: meeting the needs of clinical studies.', *J Am Med Inform Assoc* **9**(4), 369--382.

Diener2001 Diener, H. & Schumacher, H. (2001), 'Play the Application', *CG topics* **2**, 26-27.

Doan, D.K.; Paton, N.W.; Kilgour, A.C. & al-Qaimari, G. (1995), 'Multi-Paradigm Query Interface to an Object-Oriented Database', *Interacting with Computers* **7**(1), 25-47. *Doan1995*

Dorda, W.; Gall, W. & Duftschmid, G. (2002), 'Clinical data retrieval 25 years of temporal query management at the University of Vienna Medical School', *Methods Inf Med* **41**(2), 89--97. *Dorda2002*

Dorda, W.; Wrba, T.; Duftschmid, G.; Sachs, P.; Gall, W.; Rehnelt, C.; Boldt, G. & Premauer, W. (1999), 'ArchiMed: a medical information and retrieval system', *Methods Inf Med* **38**(1), 16--24. *Dorda1999*

Duftschmid, G.; Gall, W.; Eigenbauer, E. & Dorda, W. (2002), 'Management of data from clinical trials using the ArchiMed system', *Med Inform Internet Med* **27**(2), 85--98. *Duftschmid2002*

Faehnrich, K.P. (2000), *Methoden und Werkzeuge zur softwareergonomischen Entwicklung von Informationssystemen*, Jost-Jetter Verlag. *Faehnrich2000*

Fails, J.A.; Karlson, A. & Shahamat, L. (2005),'Visual Query of Multi-Dimensional Temporal Data', http://www.cs.umd.edu/class/spring2005/cmsc838s/assignment-projects/visual-query-of-temporal-data/index.htm, Review: 30.01.2007. *Fails2005*

Fayyad, U.M.; Piatetsky-Shapiro, G. & Smyth, P. (1996), 'From Data Mining to Knowledge Discovery in Databases.', *AI Magazine* **17**(3), 37-54. *Fayyad1996*

Fechter, J. (1999),'Verhalten, Kommunikation und Interaktion in Virtuellen 3D-Umgebungen', PhD thesis, Eberhard-Karls-Universität Tübingen. *Fechter1999*

Fegaras, L. (1999),VOODOO: A Visual Object-Oriented Database Language For ODMG OQL, *in* 'ECOOP Workshop on Object-Oriented Databases', pp. 61-72. *Fegaras1999*

Fishkin, K. & Stone, M.C. (1995),Enhanced dynamic queries via movable filters, *in* 'CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, pp. 415--420. *Fishkin1995*

Fluit, C. (2005),AutoFocus: semantic search for the desktop, *in* 'Information Visualisation, 2005. Proceedings. Ninth International Conference on', pp. 480--487. *Fluit2005*

Fluit, C.; Sabou, M. & van Harmelen, F. (2003),'Supporting User Tasks through Visualisation of Light-weight Ontologies', citeseer.ist.psu.edu/fluit03supporting.html, Review: 30.01.2007. *Fluit2003*

Frank2000 Frank, U. (2000), 'Die Unified Modeling Language (UML) - ein bedeutsamer Standard für die konzeptionelle Modellierung', *Das Wirtschaftsstudium (wisu)* Heft 5, 709-718.

Freksa1992 Freksa, C. (1992), 'Temporal reasoning based on semi-intervals', *Artif. Intell.* 54(1-2), 199--227.

Gall2001 Gall, W.; Duftschmid, G. & Dorda, W. (2001), 'Moving time window aggregates over patient histories', *Int J Med Inform* 63(3), 133--145.

Gall2001a Gall, W.; Heinzl, H. & Sachs, P. (2001),'Extracting a statistical data matrix from electronic patient records', *Comput Methods Programs Biomed* **66**(2-3), Elsevier, 153--166.

Gall1999 Gall, W.; Sachs, P.; Duftschmid, G. & Dorda, W. (1999), 'A retrieval system for the selection and statistical analysis of clinical data.', *Med Inform Internet Med* 24(3), 201--212.

Gall2006 Gall, W & Sachs, P. (2006),'Schulung ArchiMed Auswertesystem', Schulungsunterlage, unpublished.

Ghédira2002 Ghédira, C.; Maret, P.; Fayn, J. & Rubel, P. (2002), 'Adaptive user interface customization through browsing knowledge capitalization.', *Int J Med Inform* 68(1-3), 219--228.

Ham2002 van Ham, F. & van Wijk, J.J. (2002),Beamtrees: compact visualization of large hierarchies, *in* 'Information Visualization, 2002. INFOVIS 2002. IEEE Symposium on', pp. 93- 100.

Hand1997 Hand, D.J. (1997),Intelligent Data Analysis: Issues and Opportunities, *in* X. Liu; P. Cohen & M. Berthold, ed.,'Advances in Intelligent Data Analysis: Reasoning about Data, Second International Symposium, IDA-97, London, UK, August 4-6, 1997. Proceedings', Springer Verlag, , pp. 1--14.

Hauglid2002 Hauglid, J.O. & Midtstraum, R. (2002),SESAM: searching supported by analysis of metadata, *in* 'SAC '02: Proceedings of the 2002 ACM symposium on Applied computing', ACM Press, New York, NY, USA, pp. 418--425.

Heinecke2004 Heinecke, A.M. (2004), *Mensch-Computer-Interaktion*, Fachbuchverlag Leipzig im Carl Hanser Verlag, München, Wien.

Heuer1991 Heuer, A. & Scholl, M.H. (1991),Principles of Object-Oriented Query Languages., *in* ' Hans-Jürgen Appelrath (Hrsg.): Datenbanksysteme in Büro, Technik und Wissenschaft (BTW'91), GI-Fachtagung Bd. 270 Springer Verlag,', pp. 178-197.

Hitz2005 Hitz, M.; Kappel, G.; Kapsammer, E. & Retschitzegger, W. (2005), *UML@Work Objektorientierte Modellierung mit UML2*, dpunkt.verlag.

Hoffman1998 Hoffman, F. (1998), *Grafische Benutzeroberflächen*, Spektrum Akademischer Verlag, Heidelberg; Berlin.

Hogl, O.M.J. (2003),'Eine wissensbasierte Benutzerschnittstelle für das Invisible Data Mining', PhD thesis, Universität Erlangen-Nürnberg, Technische Fakultät. *Hogl2003*

INI-GraphicsNet (2005),'Games and Edutainment', http://www.inigraphics.net/press/brochures/games_broch/index.html, Review: 30.01.2007. *INI-GraphicsNet2005*

INI-GraphicsNet (2003),'Human-centered User Interface Design', http://www.inigraphics.net/press/brochures/hci_broch/index.html, Review: 30.01.2007. *INI-GraphicsNet2003*

Inmon, H.R. (1994), *Using the Data Warehouse*, NewYork: JohnWiley & Sons. *Inmon1994*

Jones, S. (1999),'VQuery: a graphical user interface for Boolean query specification and dynamic result preview', citeseer.ist.psu.edu/jones03vquery.html, Review: 30.01.2007. *Jones1999*

Jones, S. (1998),Dynamic Query Result Previews for a Digital Library, *in* ' Third ACM Conference on Digital Libraries', pp. 291-292. *Jones1998*

Keim, D.A.; Lee, J.P.; Thuraisingham, B.M. & Wittenbrink, C. (1995),Database Issues for Data Visualization: Supporting Interactive Database Explorartion, *in* Andreas Wierse; Georges G. Grinstein & Ulrich Lang, ed.,' Proc. IEEE Visualization Work. Database Issues for Data Visualization', Springer-Verlag, , pp. 12--25. *Keim1995*

Kemper, A. & Eickler, A. (2004), *Datenbanksysteme - Eine Einführung, 5. Auflage*, Oldenbourg. *Kemper2004*

Klinische Forschung (1999),'Denkschrift/Deutsche Forschungsgemeinschaft', Karl-Hermann Meyer ... - Weinheim; New York; Chichester; Brisbane; Singapore; Toronto: Wiley-VCH. *Forschung1999*

Koch, C. (2006),'A Visual Query Language for Complex-Value Databases', http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0602006, Review: 30.01.2007. *Koch2006*

Kosara, R. & Miksch, S. (2002), 'Visualization methods for data analysis and planning in medical applications', *Int J Med Inform* **68**(1-3), 141--153. *Kosara2002*

Kosara, R. & Miksch, S. (2001), 'Metaphors of movement: a visualization and user interface for time-oriented, skeletal plans', *Artificial intelligence in medicine.* **22**(2), 111--131. *Kosara2001*

Krause, J. (1996),'Visualisierung und graphische Benutzungsoberflächen', IZ-Arbeitsbericht Nr. 3, (http://www.gesis.org/publikationen/berichte/IZ_Arbeitsberichte/), Review: 30.01.2007. *Krause1996*

Kuhn1997 Kuhn, B.I. (1997),'Entwicklung einer Anfragesprache für temporale Datenbanken: Semantik, Ausdrucksfähigkeit und graphische Benutzerschnittstelle', Master's thesis, Universität Hannover.

Lee2006 Lee, B. (2006),'Interactive Visualizations for Trees and Graphs', PhD thesis, University of Maryland, College Park.

Leitner2003 Leitner, F.; Gaus, W.; Haux, R.; Knaup-Gregori, P. & Pfeiffer, K. (2003), *Medizinische Dokumentation*, Schattauer.

Lieberman2003 Lieberman, M.I. (2003),'The Use Of Snomed To Enhance Querying Of A Clinical Data Warehouse', Master's thesis, Oregon Health Sciences University, School of Medicine.

Lochovsky1982 Lochovsky, F.H. & Tsichritzis, D. (1982),An Interactive Query Language for External Data Bases, *in* ' Eigth International Conference on Very Large Data Bases, September 8-10, 1982, Mexico City, Mexico, Proceedings', Morgan Kaufmann, , pp. 11-21.

Lohse1994 Lohse, G.L.; Biolsi, K.; Walker, N. & Rueter, H.H. (1994), 'A classification of visual representations', *Commun. ACM* 37(12), 36--49.

Los2004 Los, R.K.; van Ginneken, A.M.; de Wilde, M. & van der Lei, J. (2004), 'OpenSDE: Row modeling applied to generic structured data entry.', *J Am Med Inform Assoc* 11(2), 162--165.

Mancini1996 Mancini, R. (1996),Interacting with a visual editor, *in* ' AVI '96: Proceedings of the workshop on Advanced visual interfaces', ACM Press, New York, NY, USA, pp. 125--131.

Massari1995 Massari, A. & Chrysanthis, P.K. (1995),Visual Query of Completely Encapsulated Object, *in* ' RIDE-DOM', pp. 18-25.

Murphy2000 Murphy, S.N.; Barnett, G.O. & Chueh, H.C. (2000), 'Visual query tool for finding patient cohorts from a clinical data warehouse of the partners HealthCare system', *Proceedings / AMIA ... Annual Symposium. AMIA Symposium.* -, 1174.

Murphy2002 Murphy, S.N. & Chueh, H.C. (2002), 'A security architecture for query tools used to access large biomedical databases.', *Proc AMIA Symp* -, 552--556.

Murphy2003 Murphy, S.N.; Gainer, V. & Chueh, H.C. (2003), 'A visual interface designed for novice users to find research patient cohorts in a large biomedical database.', *AMIA Annu Symp Proc* -, 489--493.

Murphy1999 Murphy, S.N.; Morgan, M.M.; Barnett, G.O. & Chueh, H.C. (1999), 'Optimizing healthcare research data warehouse design through past COSTAR query analysis', *Proc AMIA Symp* -, 897-900.

Murray, N.; Goble, C.A. & Paton, N.W. (1998), 'A Framework for Describing Visual Interfaces to Databases', *Journal of Visual Languages and Computing* **9**(4), 429-456. *Murray1998*

Murray, N.; Paton, N. & Goble, C. (1998),'Kaleidoquery: A Visual Query Language for Object Databases'. *Murray1998a*

Murray, N.; Paton, N.W.; Goble, C.A. & Bryce, J. (2000), 'Kaleidoquery-A Flow-based Visual Language and its Evaluation', *Journal of Visual Languages and Computing* **11**(2), 151-189. *Murray2000*

Myers, B.A. (1986),Visual programming, programming by example, and program visualization: a taxonomy, *in* ' CHI '86: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 59--66. *Myers1986*

Nadkarni, P.M. (1997), 'QAV: querying entity-attribute-value metadata in a biomedical database.', *Comput Methods Programs Biomed* **53**(2), 93--103. *Nadkarni1997*

Nadkarni, P.M. & Brandt, C. (1998), 'Data extraction and ad hoc query of an entity-attribute-value database.', *J Am Med Inform Assoc* **5**(6), 511--527. *Nadkarni1998*

Nadkarni, P.M.; Brandt, C.M. & Marenco, L. (2000), 'WebEAV: automatic metadata-driven generation of web interfaces to entity-attribute-value databases.', *J Am Med Inform Assoc* **7**(4), 343--356. *Nadkarni2000*

Nadkarni, P.M. & Marenco, L. (2001), 'Easing the transition between attribute-value databases and conventional databases for scientific data.', *Proc AMIA Symp*, 483--487. *Nadkarni2001*

Nadkarni, P.M.; Marenco, L.; Chen, R.; Skoufos, E.; Shepherd, G. & Miller, P. (1999), 'Organization of heterogeneous scientific data using the EAV/CR representation.', *J Am Med Inform Assoc* **6**(6), 478--493. *Nadkarni1999*

Nigrin, D.J. & Kohane, I.S. (2000), 'Temporal expressiveness in querying a time-stamp--based clinical database.', *J Am Med Inform Assoc* **7**(2), 152--163. *Nigrin2000*

Nigrin, D.J. & Kohane, I.S. (1998), 'Data mining by clinicians', *Proc AMIA Symp*, 957--961. *Nigrin1998*

Oellien, F. (2002),'Algorithmen und Applikationen zur interaktiven Visualisierung und Analyse chemiespezifischer Datensätze', PhD thesis, Universität Erlangen-Nürnberg. *Oellien2002*

Oliveira, A.G. & Salgado, N.C. (2006), 'Design aspects of a distributed clinical trials information system.', *Clin Trials* **3**(4), 385--396. *Oliveira2006*

Pane2000 Pane, J. & Myers, B. (2000),Tabular and textual methods for selecting objects from a group, *in* ' Visual Languages, 2000. Proceedings. 2000 IEEE International Symposium on', pp. 157--164.

Pane2002 Pane, J.; Myers, B. & Miller, L. (2002),Using HCI techniques to design a more usable programming system, *in* ' Human Centric Computing Languages and Environments, 2002. Proceedings. IEEE 2002 Symposia on', pp. 198--206.

Pantazi2006 Pantazi, S.V.; Kushniruk, A. & Moehr, J.R. (2006), 'The usability axiom of medical information systems.', *Int J Med Inform* 75(12), 829--839.

Parsia2005 Parsia, B.; Wang, T.D. & Golbeck, J., ed. (2005), *Visualizing web ontologies with cropcircles*, http://www.mindswap.org/papers/cropcircles-iswc-2005.pdf.

Paskamp1999 Paskamp, M. (1999),'Vergleichende Analyse von Anfragesprachen in Multimedia Datenbanken', Master's thesis, Universität Magdeburg.

Plaisant1998 Plaisant, C.; Mushlin, R.; Snyder, A.; Li, J.; Heller, D. & Shneiderman, B. (1998), 'LifeLines: using visualization to enhance navigation and analysis of patient records', *Proceedings / AMIA ... Annual Symposium. AMIA Symposium.* -, 76--80.

Plumlee2006 Plumlee, M.D. & Ware, C. (2006), 'Zooming versus multiple window interfaces: Cognitive costs of visual comparisons', *ACM Trans. Comput.-Hum. Interact.* 13(2), 179--209.

Pokahr2002 Pokahr, A.; Braubach, L.; Bartelt, A.; Moldt, D. & Lamersdorf, W. (2002),Vesuf, eine modellbasierte User Interface Entwicklungsumgebung für das Ubiquitous Computing, *in* M. Herczeg; W. Prinz & H. Oberquelle, ed.,' Mensch & Computer 2002 - Vom interaktiven Werkzeug zu kooperativen Arbeits- und Lernwelten', B. G. Teubner Stuttgart, Leipzig, Wiesbaden, , pp. 185-194.

Prein1999 Prein, B. (1999), *Entwicklung interaktiver Systeme*, Springer-Verlag, Berlin, Heidelberg.

Puigsegur1998 Puigsegur, J. & Agust'i, J. (1998),'Visual Logic Programming by means of Diagram Transformations', citeseer.ist.psu.edu/puigsegur98visual.html, Review: 30.01.2007.

Rainsford2000 Rainsford, C.P. & Roddick, J.F. (2000),Visualisation of Temporal Interval Association Rules, *in* '2nd International Conference on Intelligent Data Engineering and Automated Learning, (IDEAL 2000)', Springer, Shatin, N.T., Hong Kong, pp. 91--96.

Saake, G.; Sattler, K. & Keim, D. (2000),Datenbank- und Visualisierungstechnologie in der Informationsfusion, *in* T. Schulze; P. Lorenz & V. Hinz, ed.,'Simulation und Visualisierung 2000, 11.Märztagung an der Universität Magdeburg', SCS European Publishing House, , pp. 1--13. *Saake2000*

Saake, G.; Schmitt, I. & Türker, C. (1997), *Objektdatenbanken — Konzepte, Sprachen, Architekturen*, Thomson Publishing. *Saake1997*

Schiffer, S. (1996),Visuelle Programmierung - Potential und Grenzen, *in* ' Heinrich C. Mayr (Hrsg.) GI Jahrestagung Beherrschung von Informationssystemen, Tagungsband der Informatik '96, Schriftenreihe der OCG, Band 88,', pp. 267-286. *Schiffer1996*

Schindler, C.S.C. (2000),'Muster-basierte Generierung von Struktur-Editoren für visuelle Sprachen', Master's thesis, Universität Paderborn. *Schindler2000*

Schmidt, C. (2006),'Generierung von Strukturediteren für anspruchvolle visuelle Sprachen', PhD thesis, Universität Padeborn. *Schmidt2006*

Schulz, H. (2004),'Visuelles Data Mining komplexer Strukturen', Master's thesis, Universität Rostock. *Schulz2004*

Schulz, N. (2004),'Formulierung von Nutzerpräferenzen in Multimedia-Retrieval-Systemen', PhD thesis, Otto-von-Guericke-Universität Magdeburg. *Schulz2004a*

Shneiderman, B. (1986), *Designing the user interface: strategies for effective human-computer interaction*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. *Shneiderman1986*

Shneiderman, B. (1983), 'Direct Manipulation: A Step Beyond Programming Languages', *Computer* **16**(8), 57--69. *Shneiderman1983*

Shu, N.C. (1999), 'Visual Programming: Perspectives and Approaches.', *IBM Systems Journal* **38**(2/3), 199-221. *Shu1999*

Sinha, V. & Karger, D.R. (2005),'Magnet: Supporting Navigation in Semistructured Data', citeseer.ist.psu.edu/750639.html, Review: 30.01.2007. *Sinha2005*

Spolsky, J. (2001), *User Interface Design for Programmers*, Springer-Verlag, New York. *Spolsky2001*

Steffen, K. (2006),'Metaphern in der Informatik', http://waste.informatik.hu-berlin.de/Diplom/staatsexamensarbeiten/steffen.pdf, Review: 30.01.2007. *Steffen2006*

Steimann, F.; Thaden, U.; Siberski, W. & Nejdl, W. (2002),Animiertes UML als Medium für die Didaktik der objektorientierten Programmierung, *in* ' Modellierung 2002: *Steimann2002*

Modellierung in der Praxis - Modellierung für die Praxis', GI, , pp. 159--170.

Stein2006 Stein, D. (2006),'Visuelle Repräsentation von OODBMS-Anfragen in OQL (Object Query Language)', Seminararbeit, Universität Duisburg-Essen.

Stephanidis1999 Stephanidis & Akoumianakis (1999),'Multiple Metaphor Environments: Issues for effective interaction design', citeseer.ist.psu.edu/257189.html, Review: 30.01.2007.

Stoffel1998 Stoffel, K.; Davis, J.D.; Rottman, G.; Saltz, J.; Dick, J.; Merz, W. & Miller, R. (1998), 'A graphical tool for ad hoc query generation', *Proc AMIA Symp* **-**, 503--507.

Stuckenschmidt2004 Stuckenschmidt, H.; de Waard, A.; Bhogal, R.; Fluit, C.; Kampman, A.; van Buel, J.; van Mulligen, E.; Broekstra, J.; Crowlesmith, I.; van Harmelen, F. & Scerri, T. (2004),A Topic-Based Browser for Large Online Resources, *in* E. Motta & N. Shadbolt, ed.,' Proceedings of the Proceedings of the 14th International Conference on Knowledge Engineering and Knowledge Management (EKAW'04)', Springer-Verlag, , pp. 433-448.

Svanaes2000 Svanaes, D. & Verplank, W. (2000),'In Search of Metaphors for Tangible User Interfaces', citeseer.ist.psu.edu/svanaes00search.html, Review: 30.01.2007.

Tapken1999 Tapken, H. (1999),'Anfrageverarbeitung und -optimierung im MOA-Kontext', Master's thesis, Universität Oldenburg.

Trzaska2004 Trzaska, M. & Subieta, K. (2004),'Structural Knowledge Graph Navigator for the ICONS Prototype', citeseer.ist.psu.edu/trzaska04structural.html, Review: 30.01.2007.

Trzaska2004a Trzaska, M. & Subieta, K. (2004),The User as Navigator, *in* ' ADBIS 2004'.

Urbán2000 Urbán, M.A.S.; Barriocanal, E.G. & Beardo, J.M.D. (2000),'Towards a Unified Query-By-Example (UQBE): UML as a basis for a generic graphical query language', citeseer.ist.psu.edu/733654.html, Review: 30.01.2007.

VanHarmelen2001 Van Harmelen, F.; Broekstra, J.; Fluit, C.; ter Horst, H.; Kampman, A.; van der Meer, J. & Sabou, M. (2001),Ontology-based information visualisation, *in* ' Information Visualisation, 2001. Proceedings. Fifth International Conference on', pp. 546--554.

Verroust2003 Verroust, A. & Viaud, M. (2003),'Ensuring the drawability of extended Euler diagrams for up to 8 sets', citeseer.ist.psu.edu/verroust03ensuring.html, Review: 30.01.2007.

Vossen, G. (2000), *Datenmodelle, Datenbanksprachen und Datenbankmanagement-Systeme*, R. Oldenbourg Verlag, München, Wien. *Vossen2000*

Wang, T.D. & Parsia, B., ed. (2006), *Cropcircles: topology sensitive visualization of owl class hierarchies*, http://www.mindswap.org/papers/2006/cropcircles-iswc.pdf. *Wang2006*

Ward, N.S. (2004), 'Using computers for intensive care unit research', *Respir Care* **49**(5), 518--524. *Ward2004*

Weidlich, S. (1998),'Visuelle POK und Monitoring für ein OLAP-Interaktionsmodell', Master's thesis, Universität Oldenburg. *Weidlich1998*

Welty, C. & Stemple, D.W. (1981), 'Human factors comparison of a procedural and a nonprocedural query language', *ACM Trans. Database Syst.* **6**(4), 626--649. *Welty1981*

Wessel, M. (1998),'Eine visuelle Sprache zur Definition räumlicher (ebener) Konstellationen', Master's thesis, Universität Hamburg. *Wessel1998*

Wiesman, F. & Hasman, A. (1997), 'Graphical information retrieval by browsing meta-information', *Comput Methods Programs Biomed* **53**(3), 135--152. *Wiesman1997*

Wietek, F. (2000),'Intelligente Analyse multidimensionaler Daten in einer visuellen Programmierumgebung und deren Anwendung in der Krebsepidemiologie', PhD thesis, Universität Oldenburg. *Wietek2000*

Wikipedia (2007),'Entity-Relationship-Modell', http://de.wikipedia.org/wiki/Entity-Relationship-Modell, Review: 30.01.2007. *WikiER07*

Wikipedia (2007),'Unified Modeling Language', http://de.wikipedia.org/wiki/Unified_Modeling_Language, Review: 30.01.2007. *WikiUML07*

Wilde, K.; Hippner, H.; Küsters, U. & Meyer, M. (2001), *Handbuch Data Mining im Marketing - Knowledge Discovery in Marketing Databases*, Vieweg Verlag, Wiesbaden. *Wilde2001*

Zeh, T. (2003), 'Data Warehousing als Organisationskonzept des Datenmanagements', *Informatik - Forschung und Entwicklung* **18**(1), 32-38. *Zeh2003*

Zloof, M.M. (1977), 'Query-by-Example: A Data Base Language.', *IBM Systems Journal* **16**(4), 324-343. *Zloof1977*

ANHANG

B

Glosssar

Auf dem Maskenball der Begriffe hält sich der Slogan als Definition...

Stanislaw Jerzy Lec

In diesem Glossar finden sich wichtige Begriffe, die für die Arbeit Bedeutsam sind.

Folgende Notationen werden verwendet:

~	steht als Kürzel für den erklärten Begriff
Text kursiv	verweist auf einen anderen Begriff im Glossar

A

Ad-hoc-Anfrage Im Rahmen einer explorativen Datenanalyse spontan formulierte und interaktiv gestellte Anfrage an ein Datenbanksystem.

Ätiologie Die ~ ist die Lehre von der Entstehung von Krankheiten, ihren Ursachen und Einflussfaktoren die ihre Entwicklung begünstigen.

Aggregierbarkeit Die ~ einer Kennzahl bezüglich eines kategoriellen Attributs gibt an, ob die Werte der Kennzahl mittels Aggregierung über ein entsprechendes kategorielles Attribut und unter Verwendung einer Aggregatsfunktion zu gröberen Angaben der gleichen Kennzahl zusammengefasst werden können.

Aggregierung Unter ~ wird die Zusammenfassung von Datenwerten eines Datenraums verstanden. Hierunter fallen

- die Berechnung von Makrodaten aus Mikrodaten, indem die betrachteten Objekte über ausgewählte Kategorien gemäß ihrer Eigenschaften klassifiziert und auf den erhaltenen Gruppen Kennzahlen berechnet werden.
- die einfache Vergröberung von Angaben oder die Zusammenfassung zu einem Gesamtwerk bezüglich eines oder mehrerer kategorieller Attribute unter Verwendung der Aggregatsfunktionen der jeweiligen Kennzahl
- die Berechnung neuer Kennzahlen, die ein oder mehrere kategorielle Attribute entsprechend ihrer Berechnungsfunktion eliminieren.

Aggregierungsebene Zusammenfassung mehrerer Kategorien eine Domäne, die im nicht strengem Sinne als gleich fein angesehen werden.

Aggregierungsfunktion Funktion, die zur Zusammenfassung von Werten einer Kennzahl bei Aggregierung über ein kategorielles Attribut eines Datenwürfels angewendet wird.

Anfragesprache Als Teil der Datenmanipulationssprache (DML), stellt die ~ lediglich Retrieval-Funktionen zur Verfügung und hat keine Update-Funktionen. Die ~ dient zur Bereitstellung der Daten aus einer Datenbank

ANSI/SPARC-Architektur Architektur für Datenbankmanagementsysteme, deren Ziel es ist, Benutzer-Schnittstellen und physikalische Realisierung von Datenbanken klar zu trennen. Sie wird auch als Drei-Schichten-Architektur bezeichnet, da entsprechende Datenbanksysteme ihre Datenbanken über entsprechende Datenschemata auf drei verschiedenen Ebenen, der internen, der konzeptuellen und der externen Ebene modellieren.

Atomare Entität ~ ist eine Datengruppe, deren Elemente durch bestimmte Merkmale charakterisiert sind.

Attribut ~e sind Datenstrukturen zur Spezifikation bestimmter Aspekte von Datenbeständen oder einzelnen Datenobjekten der definierten Kennzahlen bzw. deren Ausprägungen. In VISA sind speziell graphische ~e von Darstellungssymbolen einer Grafik von Interesse.

B

Basisdaten Als ~ werden die in einer Datenbank gespeicherten Ausgangsdaten einer Datenanalyse bezeichnet.

Browser Systeme, mit denen der Anwender den Inhalt einer Datenbank auf unstrukturierte Art und Weise durchschauen kann.

Button Schaltfläche in einer graphischen Benutzeroberfläche.

C

Cache Temporärer Speicher mit begrenzter Kapazität, auf den effizient zugegriffen werden kann.

Clusteranalyse Oberbegriff für statistische Verfahren zur Gruppierung untersuchter Daten anhand ihrer Attribute. In der Epidemiologie werden unter ~ meist Verfahren verstanden, die Häufigkeiten von Fällen in Zeit

und/oder Raum untersuchen bzw. Auffälligkeiten in ihrem Auftreten beurteilen.

Clusterindex Statistische Kennzahl für die räumliche Verteilung (Clusterung) von Fällen innerhalb der betrachteten Studienpopulation und Zeitintervall.

D

Darstellungssymbol Atomares Element einer Graphik (etwa ein Punkt oder Linie), das bestimmte graphische Attribute aufweist, über deren Ausprägungen Datenwerte repräsentiert werden.

Datendefinitionssprache Mit der ~ (DDL) werden Datenbank-Schemata beschrieben.

Data Mart In Hinblick auf Datenumfang, Anwendungsbereich und Funktionalität "kleines" *Data Warehouse*.

Data Mining Bezeichnet die Analyseschritt im Rahmen des Knowledge Discovery in Databases. Im Gegensatz zu der explorativen Datenanalyse spielt die Interaktion mit dem Systemanwender eine untergeordnete Rolle.

Data Warehouse *Datenbanksystem*, das Daten aus verschiedenen operativ genutzten Datenbanksystemen integriert und zum Zweck der Datenanalyse aufbereitet. Manchmal wird unter ~ auch nur die entsprechende *Datenbank* verstanden.

Data Warehousing Oberbegriff für alle Themen, die sich mit Konzeption, Aufbau und Betrieb eines *Data Warehouse* beschäftigen.

Datenbank Der Begriff ~ bezeichnet eine Datensammlung unabhängig von deren physischer Realisierung.

Datenbank-Management-System Alle Softwarekomponenten, die zur Verwaltung der *Datenbank* erforderlich sind, werden unter dem Sammelbegriff ~ (DBMS) zusammengefasst

Datenbankschema Das ~ beschreibt die Struktur des Inhalts einer *Datenbank*.

Datenbanksystem Der Begriff des ~s integriert das DBMS und die verwalteten *Datenbank*.

Datendefinitionssprache (DDL) Bezeichnet bei *Datenbanksystemen* die Sprache, die dem Datenbankverwalter die Definition von Datenstrukturen erlaubt. Zur DDL von SQL gehören zum Beispiel die Befehle CREATE TABLE und CREATE VIEW.

Datenfeld Ein ~ ist ein Bereich auf einem Formular, hat eine Bezeichnung und ist mit einer *Variable* verknüpft.

Datenmanipulationssprache Die ~ (DML) dient zur Formulierung von Abfragen und Operationen auf den Daten einer *Datenbank*. Die ~ (DML) besteht aus zwei Teilen, der *Anfragesprache* und der eigentlichen Datenmanipulationssprache, die wiederum als Teil der ~, zum Einfügen, Löschen und Modifizieren von Daten dient.

Datenmodell Die Modellierung der realen Welt wird in allen *Datenbanksystemen* durch Datenmodelle realisiert. Das Datenmodell legt die Struktur fest, in der ein Abbild der realen Welt im Computer generiert wird.

Datensystem Stellt im Schichtenmodell eine Schicht, die an einer mengenorientierten Schnittstelle das *Datenmodell* und dessen deskriptive Sprache zur Verfügung stellt.

Datenwörterbuch Das ~ ist eine Zusammenfassung von *Metadaten* über den Inhalt einer *Datenbank*.

Darstellungsmodell Das ~ ist ein Konzept zur Darstellung der Komponenten einer *visuellen Sprache*.

DDL (Data Definition Language) siehe *Datendefinitionssprache*.

Deklarativität Die ~ ist die Eigenschaft einer Sprache, die besagt, das die erfragten Informationen vom Anwender nur beschrieben werden müssen und nicht durch eine Berechnungsvorschrift spezifiziert werden müssen.

Direkte Manipulation Bezeichnet die Interaktion zwischen Mensch und Computer über graphische Objekte nach dem *WYSIWYG-Prinzip*.

Dokument Ein teilweise oder ganz ausgefülltes Formular, dass in Verbindung mit einem Patienten abgespeichert wird.

DML (Data Manipulation Language) siehe *Datenmanipulationssprache*

Dünnbesetzte Matrix siehe *Sparse-Matrix*.

E

Eindimensionale Anfragesprache Eine ~ ist eine rein textuelle Sprache die zur Darstellung nur eine Dimension nutzt, im Gegensatz zu mehrdimensionalen Sprachen die zur Darstellung mehrere Dimensionen nutzen.

Epidemiologie Das Studium der Verteilung von Krankheitshäufigkeiten in menschlichen Populationen. Die ~ betrachtet zur Lösung medizinischer Probleme Krankheitshäufigkeiten in unterschiedlichen Bevölkerungsgruppen.

ER-Diagramm Entity-Relationship Diagramm - Diagramm zur Visualisierung von *ER-Modellen*. Gegenstände werden als Rechtecke dargestellt und Beziehungen als Rauten mit Verbindungen zu den entsprechenden Rechtecken.

EER-Diagramm Das ~ ist ein *ER-Diagramm* um Konzepte der Generalisierung, Spezialisierung, Assoziation und Aggregation erweitert.

ER-Modell Entity-Relationship-Modell - Modell um eine gegebene Miniwelt zu beschreiben. Die Miniwelt wird beschrieben als eine Menge von Gegenständen, zwischen denen wohl definierte Beziehungen bestehen.

Externe Ebene Die ~ beschreibt die Sicht der Anwender beziehungsweise der Applikationen auf die *Datenbank*. Die ~ abstrahiert von der physischen Organisation der Daten.

F

Formular Ein ~ ist eine Ansammlung von Datenfeldern. Ein ~ wird zum Sammeln und/oder Anzeigen von Patientendaten verwendet.

Fremdschlüssel Der ~ ist das Attribut einer Relation das auf einen Primärschlüssel einer anderen Relation hinweist.

I

Icon Ein ~ stellt eine bildliche Repräsentation einer Gruppe von Objekten

Informationssystem Ein System zur Speicherung, Wiedergewinnung, Verknüpfung und Auswertung von Informationen. Es besteht aus einer Datenverarbeitungsanlage, einem Datenbanksystem und den Auswertungsprogrammen.

Intentionale Anforderung Unter ~ versteht man die Forderung an den Anwender einer Anfragesprache nach der Fähigkeit, die gesuchten Daten genau zu spezifizieren.

Interne Ebene Im 3 Schichten-Modell der ANSI-SPARC Systemarchitektur bezeichnet die ~ die systemspezifische Implementierung der Datenbank. Auf dieser Ebene befinden sich beispielsweise die Dateiorganisation mit ihren Zugriffspfaden sowie die Pufferverwaltung.

K

KIS Ein Krankenhaus-Informationssystem ist das Teilsystem eines Krankenhauses, welches alle informationsverarbeitenden Prozesse und die an ihnen beteiligten menschlichen und maschinellen Handlungsträger in ihrer informationsverarbeitenden Rolle umfasst.

Konzeptuelle Ebene Auf der ~ werden die Katalogdaten (Metadaten) der Datenbank verwaltet, beispielsweise das Datenbankschema und die Parameter der in der internen Ebene eingesetzten Zugriffsstrukturen. Diese Ebene ist vom Betriebssystem und der verwendeten Hardware unabhängig.

L

LabView ~ steht für ‘‘Laboratory Virtual Instruments Engineering Workbench’’ und ist ein für Anwendungen in der Mess- und Steuerungstechnik geeignetes, visuelles Programmiersystem

Lineare Menüs ~ finden sich auf jedem Desktop, in Form von etwa *Pull–Down–Menüs*.

M

Mehrdimensionalen Sprachen ~ nutzen zur Darstellung mehrere Dimensionen, im Gegensatz zu textuellen Sprachen die zur Darstellung nur eine Dimension nutzen.

Metadaten Daten über Daten. ~ sind also Angaben zur Beschreibung von Daten. Versucht man zwischen Daten und Metadaten zu unterscheiden, so ist es hilfreich den „Zweck“ als Begriff einzuführen. Der Zweck bestimmt das Ergebnis; um in der Lage zu sein, einen bestimmten Zweck zu erfüllen, ein bestimmtes Ergebnis zu erreichen, werden Metadaten benötigt. Das Ergebnis kann aus Daten bestehen, insbesondere können Metadaten in ihrer Rolle als Daten Teil des Ergebnisses sein.

Metainformationen Die ~ sind Informationen, die nicht explizit in der *Datenbank* gespeichert werden, die aber aus gespeicherten Daten berechnet werden können.

O

Ontologien Begriffshierarchien

ORACLE Eine kommerzielle, relationale Datenbank

P

Pie Menü Ein Menü wird in diverse Kuchenstücke zweidimensional aufgeteilt. Dabei können vereinzelte Menüpunkte wieder zu einem eigenen ~ aufgehen. Dieses wird, wie bei den *linearen Menüs*, in mehrere Hierachiestufen aufgeteilt. Der große Vorteil, dieser Menüs bieten ist die Möglichkeit, intuitive Menüführung durch direkte Handbewegungen zu realisieren.

Pull-Down-Menü Ein Menü, das bei Bedarf wie eine Jalousie heruntergelassen und nach der Auswahl durch den Anwender wieder ausgeblendet wird.

Primärschlüssel Ein ~ ist ein oder eine Menge von Attributen, die einen Eintrag einer Relation eindeutig identifizieren.

Problemorientiertes Design ~ ist eine Bezeichnung für Softwaredesign, bei dem die Lösung des Problems im Mittelpunkt steht, die Benutzerschnittstelle jedoch eine untergeordnete Rolle spielt.

Q

QBE Query by Example ist eine visuelle Sprache zur Beschreibung von Datenbankabfragen in relationalen Datenbanken..

R

Raumbezogene Anfragen In epidemiologischen Studien spielen Anfragen bei denen räumliche Trends beobachtet werden eine wichtige Rolle.

RPDR Query Tool ~ ist ein Akronym für Research Patient Data Registry Query Tool. Es ist ein Anfragetool für die klinischen Datenbanken.

S

Schichtenmodell Das ~ ist ein Modell um die Struktur eines Datenbanksystems zu erläutern. Das Datenbanksystem wird dabei in zueinander hierarchisch angeordnete Schichten aufgeteilt, die jeweils unterschiedlich mächtige Abstraktionsniveaus darstellen. Jede Schicht bietet ihrer oberen Schnittstelle eine Reihe von Objekten und Operationen an, die gemäß ihrer Abstraktionsebene bereits gewisse Anforderungen erfüllen bzw. unterstützen und die unter Verwendung der Objekte und Operatoren der darunter liegenden Schicht realisiert sind.

Sparse-Matrix Eine ~ (*Dünnbesetzte Matrix*; von engl. sparse - spärlich) ist eine Matrix, welche nur sehr wenige von Null verschiedenen Einträge besitzt, also dünn besetzt ist. Solche Matrizen spielen eine wichtige Rolle in der numerischen Mathematik, da ihre effiziente Behandlung Bestandteil vieler numerischer Verfahren ist.

Speichersystem Eine Schicht im Schichtenmodell, deren Aufgabe die Realisierung einer seitenorientierten, virtuellen Adressierung in einem unendlichen linearen Adressraum ist. An seiner Schnittstelle zum Zugriffssystem bietet ein ~ seitenorientierte Operationen an.

Sprachbereich Der ~ ist ein Maß über die Anwendbarkeit einer Sprache. Der ~ reicht von genereller und weiter Anwendbarkeit zu spezifischen und eingeschränkten Speziallösungen.

Sprachlevel Ein ~ stellt ist Maß über den Abstraktionsgrad einer Sprache. Der ~ steht im konträren Verhältnis zu der Anzahl an Details, die ein Anwender in den Computer eingeben muss, um das gewünschte Ergebnis zu erreichen.

Sprachprofil Ein ~ ist die dreidimensionale Darstellung zur Visualisierung verschiedener Eigenschaften einer visuellen Sprache.

Standbein einer Sprache Beschreibt die grundlegende Vorraussetzung, damit eine Sprache angewendet werden kann. Es wird unterschieden zwischen tragenden und nicht tragenden Standbeinen. Ein tragendes Standbein beinhaltet eine Voraussetzung, ohne die eine Sprache nicht anwendbar ist. Ein nicht tragendes Standbein beschreibt eine Voraussetzung, ohne die die Möglichkeiten einer Sprache zwar eingeschränkt werden, aber die Sprache anwendbar bleibt.

Strukturelle Anforderung Forderung an den Nutzer einer Datenbank nach Verständnis des Datenmodells und Kenntnis des Datenbankschemas.

U

UML Die Unified Modeling Language ist eine von der Object Management Group (OMG) standardisierte Sprache für die Modellierung von Software und anderen Systemen.

User-zentriertes Design ~ ist eine Bezeichnung für Softwaredesign, bei dem der Benutzer im Mittelpunkt steht.

V

Variable Eine ~ ist ein Wert den ein Datenfeld beinhalten kann.

Visueller Grad Maß inwieweit eine visuelle Sprache visuelle Repräsentationen wie z.B. Icons, Graphen, Diagramme oder Bilder verwendet.

Visuelle Anfragesprache Sprache, die dem Anwender die Spezifikation von Datenbankanfragen mit zwei oder mehrdimensionalen Hilfsmitteln ermöglicht.

Visuelles Anfragesystem Eine Schicht im Schichtenmodell, die mittels eines Darstellungsmodells eine visuelle Sprache zur Verfügung stellt.

Visuelle Sprache Sprachen, deren Sprachkonstrukte zwei- oder mehrdimensional dargestellt werden.

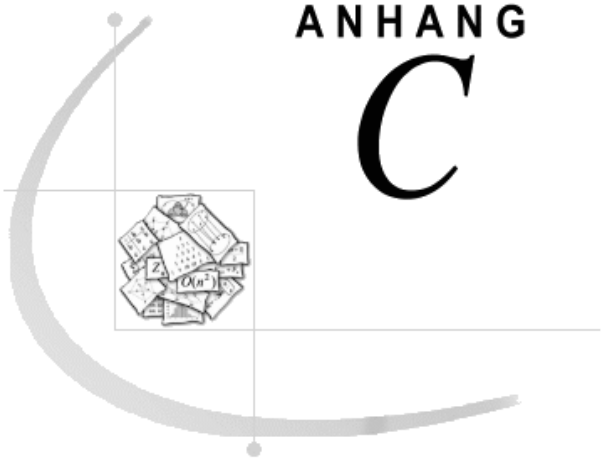
W

WYSIWYG-Prinzip Prinzip der direkten Manipulation. Dies beinhaltet eine permanente, graphische Darstellung der jeweils interessierenden Objekte. Physische Aktionen wie Bewegungen der Maus, Selektionsmechanismen und zum Anklicken gekennzeichneter Schalter ersetzen eine komplexe Kommandosyntax.

Z

Zeitbezogene Anfragen Bei zeitbezogenen Anfragen können sowohl Zeitpunkte, Zeitintervalle wie auch Zeitspannen betrachten werden.

Zugriffssystem Eine Schicht im Schichtenmodell, die eine satzorientierte Schnittstelle realisiert und dazu ein Speichermodell mit zugehöriger prozeduraler Sprache bereitstellt.



ANHANG

C

Abbildungs- & Tabellen-verzeichnis

Auch die Überflüssigen werden ständig gebraucht...

Stanislaw Jerzy Lec

Abbildungsverzeichnis

Kapitel 1

Kapitel 2

Kapitel 3

Kapitel 4

Kapitel 5

Kapitel 6

Tabellenverzeichnis

Kapitel 3

Kapitel 5

Kapitel 6

ANHANG

D

Stichwort-verzeichnis

Wo sind die Fundstellen der Weisheit?
Gewöhnlich dort, wo diese begraben liegt...

Stanislaw Jerzy Lec

A

B

C

D

E

N

O

P

Q

R

S

T

U

V

W

Z