Inhaltsverzeichnis

Inhaltsverzeichnis **1**

1 Einleitung **5**
1.1 Einführung und Aufgabenstellung 5
1.2 Übersicht über die Arbeit 5
1.3 Motivation 5

2 Kommunikations- und Präsentationssysteme **7**

3 Anforderungsanalyse **9**
3.1 Informationsbedarfanalyse 9
3.1.1 Universität 10
3.1.2 Personen 10
3.1.3 Veranstaltungen 11
3.1.4 Benutzerkommunikation 12
3.2 Funktionale und dynamische Anforderungen 12
3.2.1 Präsentation 12
3.2.2 Verwaltung der Daten 12
3.2.3 Kommunikation 13
3.2.4 Sicherheit 13
3.3 Software-Ergonomie 15
3.3.1 Dialoggestaltung 15
3.3.2 Ein-/Ausgabe-Gestaltung 16

4 Vorüberlegungen **17**
4.1 Einleitung 17
4.2 Entwurf eines relationalen Datenbankschemas 18
4.1.1 Definitionen des relationalen Datenmodells 18
4.1.2 Konzeptioneller Entwurf des Datenbankschemas 18
4.1.3 Logische Entwurf des Datenbankschemas 20
4.3 Objektorientierter Entwurf 21
4.3.1 Der Architekturentwurf 21
4.3.2 Der Implementierungsentwurf 24

5 Entwurf des Kommunikations- und Präsentationssystem **25**
5.1 Datenhaltung 25
5.1.1 Das Teilschema Universitätsstruktur 26
5.1.2 Das Teilschema Personen 28
5.1.3 Das Teilschema Benutzerkommunikation 32
5.1.4 Das Teilschema Vorlesungen 35
5.1.5 Das Teilschema Seminare 40
5.1.6 Das Teilschema Vorträge 45
5.1.7 Das Teilschema Benutzerrechte 47
5.2 Datenbankschema 48

5.3 Funktionalität und Dynamik 54
5.3.1 Einleitung 54
5.3.2 Entwurf der Fachkonzeptklassen 54
5.3.3 Entwurf der objektorientierten Zwischenschicht 57
5.4 Benutzeroberfläche 60
5.4.1 Einleitung 60
5.4.2 Erscheinungsbild 60
5.4.3 Ereignisbehandlung 63
5.5 Sicherheit 64

6 Realisierung des Kommunikations- und Präsentationssystems 65
6.1 Benutzeroberfläche und Funktionalität 65
6.1.1 Menü 66
6.1.2 Anlegen eines Studenten 67
6.1.3 Anzeigen von Studenten 68
6.1.4 Bearbeiten von Studenten 69
6.1.5 Anlegen einer Abteilung 70
6.1.6 Anzeigen von Vorlesungen 71
6.1.7 Anzeigen von Seminaren 73
6.1.8 Senden von Nachrichten 74
6.1.9 Bearbeiten von Benutzerrechten 75
6.2 Datenhaltung 77

7 Zusammenfassung und Ausblick 81

A Abbildungsverzeichnis 83

B Quellcode-Dateien 85
B1 Fachkonzeptklassen 85
B2 Klassen für die Benutzeroberflächen und deren Ereignisbehandlung 85
B3 Klassen für die objektorientierte Zwischenschicht 86
B4 Sonstige Klassen 86

C Literatur 87

1 Einleitung

1.1 Einführung und Aufgabenstellung

Seit die Möglichkeiten von kooperativen Veranstaltungen über mehrere Universitäten hinzugekommen sind, hat die Komplexität der Teilnehmerverwaltung und -kommunikation durch die zeitliche und örtliche Entzerrung deutlich zugenommen. Hatte man bisher Ansprechpartner und Organisatoren, die örtlich greifbar waren, so haben durch den Gewinn der verteilten Veranstaltungen die Möglichkeiten der Kommunikation abgenommen.

Aufgabe dieser Diplomarbeit ist der Entwurf und die Realisierung eines Systems mit dem universitäre Veranstaltungen und deren Teilnehmer verwaltet werden können. Dabei sollen die Informationen in geeigneter Weise über das WWW (Word Wide Web) präsentiert werden.

Diese Einschränkungen soll durch ein integriertes Kommunikationssystem verbessert bzw. kompensiert werden.

1.2 Übersicht über die Arbeit

In Kapitel 2 werden zunächst Kommunikations- und Präsentationssysteme betrachtet. Im dritten Kapitel werden die Anforderungen eines Kommunikations- und Präsentationsystems für universitäre Umgebungen analysiert. Es werden zuerst in einer Informationsbedarfsanalyse die Daten ermittelt, die in diesem System auftreten. Anschließend werden die funktionalen und dynamischen Anforderungen betrachtet und zum Schluß dieses Kapitels wird auf die Software-Ergonomie eingegangen.
Im anschließenden vierten Kapitel werden Probleme angesprochen, die sich aus der bestehenden Systemumgebung ergeben.
Im fünften Kapitel folgt der Entwurf des Systems. Beginnend wird ein Datenbankschema für ein relationales Datenbanksystem entworfen, danach die Funktionalität und Dynamik des Systems und am des Kapitels werden die Spezifikationen Benutzerschnittstelle gemacht.
In Kapitel 6 wird die Realisierung des Systems zunächst anhand einiger Benutzerschnittstellen vorgestellt und anschließend erfolgt die Auflistung der verwendeten Datenbanktabellen.
Den Abschluß bildet eine Zusammenfassung der vorliegenden Arbeit, sowie ein Ausblick auf mögliche Optimierungen und Ergänzungen des Systems.

1.3 Motivation

Verkaufsprospekt
Mangel an Kommunikation an Universitäten.
Halbanonyme Kommunikation fällt manchen Personen leichter.
Verteilte Veranstaltungen
Schnellerer Zugriff auf Informationen
WWW -> Systemunabhängige Ablaufumgebung mit unbegrenzter Zugriffsmöglichkeit

auf die innovative Technik hinweisen: OO-Anwendung -> RDBS Strukturbruch
Wert auf die modulare Struktur gelegt

Bis vor kurzem konnten sich Studenten über die Inhalte von universitären Veranstaltungen (Vorlesungen, Seminare, ..) nur anhand eines gedruckten Vorlesungsverzeichnisses oder über das Schwarze Brett informieren. Vereinzelt haben einige Abteilung in eigener Regie HTML-Seiten erzeugt, um so den Informationsfluß zwischen Lehrenden und Lernenden zu beschleunigen. Nach meinen Erkenntnissen handelt es dabei um statische HTML-Seiten, die mit Hilfe eines speziellen Editor, wie etwa Frontpage oder Netscape Composer generiert werden.

2 Kommunikations- und Präsentationssysteme

Anforderungen, Aufgaben, mögliche Realisierungen, Probleme, Aktualität, Zugriff, Integration der beiden Systeme in ein System Die immense Zuwachsrate von Anbietern und Nutzern ist besonders durch die Schaffung von benutzerfreundlichen, graphischen Schnittstellen zu den im Internet vorhandenen Ressourcen zu erklären.

Der Zweck dieses Kommunikations- und Präsentationssystems ist nicht nur die Verwaltung und Darstellung von Information, sondern soll auch Information durch Kommunikation bewirken.

Der Informationsbedarf eines solchen Systems „*konkretisiert sich außer in bestimmten inhaltlichen Wünschen insbesondere in den Anforderungen an die Aktualität, Schnelligkeit, Genauigkeit, Vollständigkeit, Redundanzfreiheit und Übersichtlichkeit der erforderlichen Information*“ [MuSnZö75]. Auch die Systemanforderungen und die Höhe der Verfügbarkeit sind von entscheidender Bedeutung.

Bei Verwendung einer Datenbank werden einige dieser Anforderungen bereits erfüllt. So wird die Aktualität der Information durch eine einmalige Eingabe in das System bewirkt. Dies hat den Vorteil, daß mehrere Versionen von Informationsstände vermieden werden, wie es bei der Verwendung von Printmedien vorkommen kann.
Die Vollständigkeit und Redundanzfreiheit muß durch die Informationsbedarfsanalyse und den anschließenden Datenbankentwurf abgedeckt werden.
Die Aufgabe der Präsentation ist die Darstellung der Informationen in einer benutzergerechten und übersichtlichen Form, die durch die Benutzerschnittstelle bewirkt wird.
Bei der Verwaltung von sensitiven Daten, insbesondere personenbezogenen Daten muß das System einen Schutz gewährleisten, der nur berechtigten Personen das Einsehen und Verwalten dieser Daten gestattet. Dieses Problem muß durch eine geeignete Zugriffskontrolle realisiert werden.

Intranet
Noch nicht vollständig

3 Anforderungsanalyse

Ziel der Anforderungsanalyse ist es, die quantitative und qualitative Eigenschaften des Systems zu spezifizieren. Das Ergebnis der Anforderungsanalyse wird im Bereich des Software-Engineering als „Pflichtenheft“ bezeichnet.

Die Anforderungen werden in sogenannte „Sichten“ unterteilt, die die verschiedenen Blickwinkel darstellen, aus denen das System beschrieben werden kann. Zu diesem Zweck werden vier verschiedene Sichten und ihre Zusammenhänge charakterisiert.

Zu den vier Sichten gehören

- die **Daten**,
- die **Funktionen**,
- die **Dynamik** und
- die **Benutzeroberfläche** des Systems.

Die Beschreibung der **Daten**-Sicht wird im Kapitel 3.1 Informationsbedarfsanalyse dargestellt, die Sichten **Funktionen** und **Dynamik** werden zusammengefaßt und im Kapitel 3.2 Funktionale und dynamische Anforderungen beschreiben und die vierte Sicht **Benutzeroberfläche** wird im Kapitel 3.3 Software-Ergonomie spezifiziert.

3.1 Informationsbedarfsanalyse

Die Informationsbedarfsanalyse hat die Aufgabe die Informationen zu ermitteln, die durch das System verwaltet werden [Mei72], [Vet87], [Loc83].
Dabei wurden folgende Quellen zur Informationssammlung verwendet:

- das Personal- und Vorlesungsverzeichnis der Universität Stuttgart,
- der Studienplan der Universität Stuttgart für den Diplomstudiengang Informatik
- die Homepages der Fakultät Informatik der Universitäten Stuttgart und Karlsruhe
- Gespräche mit Studenten und Mitarbeitern

Als Ergebnis der Informationssammlung haben sich die vier Bereiche Universität, Personen, Veranstaltungen und Benutzerkommunikation herauskristallisiert, die nun im Einzelnen in textueller Form vorgestellt werden.

3.1.2 Universität

Eine Universität hat folgende hierarchische Struktur:

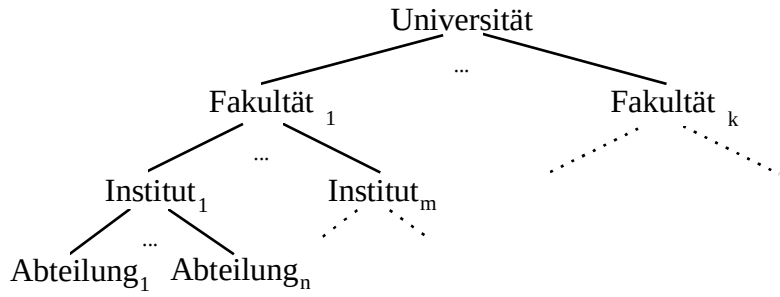


Abbildung 1: Universitätsstruktur

Alle Vertreter dieser Ebenen werden durch einen Namen benannt, (beispielsweise „Informatik“ für eine Fakultät oder „Institut für Parallele und Verteilte Höchstleistungsrechner (IPVR)“ für ein Institut) und zu einer Adresse zugeteilt. Die Angaben der Adresse bestehen aus der Straße, der Postleitzahl und dem Ort.
Jede Ebene dieser hierarchischen Struktur hat spezielle Aufgaben, die durch Personen repräsentiert werden. So besitzt jeder Vertreter dieser Ebene ein Sekretariat, das die jeweiligen Verwaltungsaufgaben wahrnimmt. Dieses Sekretariat wird außer der Person, die diese Tätigkeit ausübt, noch durch die Angaben Sprechstunde, Telefon und Zimmer beschrieben.
Die einzelnen Ebenen besitzen ergänzend noch weitere Positionen, z. B.:

- Fakultät: Dekan, Prodekan und Studiendekan.
- Institut: Direktor, stellvertretende Direktor, Prüfungsausschuß und Studienberatung.
- Abteilung: Leitung.

Diese Positionen werden von Universitätsprofessoren besetzt. Jede dieser Positionen wird entsprechend dem Sekretariat durch eine Person und Angaben über die Sprechstunde, das Telefon und Zimmer beschrieben.

3.1.2 Personen

Eine Universität besteht aus Personen, die an der Universität angestellt sind. Dazu gehören Personen, die Wissen vermitteln und als Lehrkörper bezeichnet werden. Personen, die Vorlesungen halten, werden Dozenten genannt.
Ferner gibt es Personen, die kein Wissen vermitteln, jedoch für verwaltungstechnische Aufgaben zuständig sind. Diese Personen sind in allen Ebenen (Universität, Fakultät, Institut und Abteilung) beschäftigt sein.
Zusätzlich gibt es externe Personen, die von der Universität eingeladen werden, um einen Vortrag zu halten. Bei diesen externen Personen kann es sich um Professoren aus anderen Universitäten, Repräsentanten eines Unternehmens oder Vertreter aus der Politik handeln.
Überdies gibt es noch Studenten, die an einer Universität studieren. Jeder Student studiert ein Studienfach an einer Fakultät. Unter bestimmten Voraussetzungen ist es möglich zwei Studienfächer zu belegen. Bei Beginn des Studiums erhält jeder Student eine eindeutige Matrikelnummer, die z. B. für die Anmeldung von Prüfungen benötigt wird.

Da die Aufgabe dieses Systems auch die Präsentation von Informationen ist, müssen auch Benutzer berücksichtigt werden, die keinen direkten Beziehung zur Universität haben. Es handelt sich hier um anonyme Benutzer, die sich mit Hilfe dieses Systems über eine Universität informieren möchten. Diese Benutzer werden nicht im System verwaltet und haben nur einen beschränkten Zugriff. Beispielsweise könnte es sich um Schulabgänger handeln, die nach der zuständigen Person für die Studienberatung suchen.

Die Beschreibung des Bereiches Personen zeigt, daß es grundsätzlich zwei verschiedene Typen von Benutzern gibt:

- **aktive Benutzer**: Dazu gehören die Studenten und alle Personen, die an der Universität angestellt sind. Diese Personen werden in der Datenbank verwaltet. Für alle diese Personen werden die Anrede, der Name und die Anschrift im System abgespeichert. Unter bestimmten Voraussetzungen können diese Personen zu Rechten gelangen, die das Editieren der Informationen erlaubt.
- **passive Benutzer**: Diese Art der Benutzer erhält die Möglichkeit, sich mit Hilfe des Systems über die Universität zu informieren. Sie haben jedoch keine Berechtigung, aktiv am System teilzunehmen.

3.1.3 Veranstaltungen

An einer Universität gibt es eine Vielzahl von Veranstaltungen. Dazu gehören unter anderem Vorlesungen, Seminare, Hauptseminare, Fachpraktikas, Kompaktkurse, Kolloquien und Vorträge von externen Personen.
Jede Veranstaltung besitzt einen Titel, der den Inhalt der Veranstaltung umschreibt. Durch eine Kurzbeschreibung können Interessierte ergänzende Informationen zum Inhalt einer Veranstaltung erhalten. Alle Veranstaltungen, die beim Prüfungsamt angemeldet werden müssen, besitzen eine eindeutige Prüfungsnummer zur ihrer Identifikation.

Vorlesungen finden normalerweise in einem zweisemestrigen Turnus statt. Sie werden von einem oder mehreren Dozenten gehalten. Bei diesen Dozenten handelt es sich um Professoren oder um deren Assistenten aus ihrer Abteilung. Die Vorlesungen werden von einem Prüfer in einer schriftlichen oder mündlichen Form geprüft. Zu manchen Vorlesungen gibt es Übungen, in denen die Studenten ihre in der Vorlesung erworbene Kenntnisse vertiefen können. Aufgrund der Teilnehmeranzahl werden diese Übungen in Übungsgruppen eingeteilt, die dann von Übungsleitern betreut werden. Die in den Übungen gestellten Aufgaben werden unter Mithilfe der Übungsleiter gelöst. Die Vorlesungen und die dazugehörigen Übungen finden zu festgelegten Zeiten in bestimmten Räumen statt.

In Seminaren haben Studenten die Möglichkeit über ein Thema zu referieren. Die Themen eines Seminars werden in der Regel in Themengruppen unterteilt, die einen zusammenhängenden Aspekt des Seminars behandeln. Die Betreuer des Seminars vergeben die Themen, die dann von den Studenten vorbereitet und an einem festgelegten Termin vorgetragen werden. Vorwiegend besteht für die Teilnehmer eines Seminars Anwesenheitspflicht. Der Unterschied zwischen Seminaren und Hauptseminaren besteht darin, daß die Teilnahme eines Seminars die Voraussetzung für die Teilnahme eines Hauptseminars ist und daß Hauptseminare von einem Prüfer benotet werden.

Eine weitere universitäre Veranstaltung sind Vorträge zu einem speziellen Sachgebiet. Sie werden von einem Referenten gehalten. Zum einen gibt es interne Referenten, bei denen es sich um Studenten oder Mitarbeiter einer Abteilung handeln kann und zum anderen externe Referenten. Diese externe Referenten (z. B. Repräsentanten eines Unternehmens oder Vertreter aus der Politik) haben sonst keine engere Beziehung zum Universitätsbetrieb.

3.1.4 Benutzerkommunikation

Das System hat nicht nur die Aufgabe eines Informationssystems, sondern es wird auch die Möglichkeit geschaffen, daß die aktiven Benutzer, insbesondere die Teilnehmer einer Veranstaltung miteinander kommunizieren können.
Dazu benötigt jeder aktive Benutzer eine eindeutige Adresse, das für das korrekte Senden einer Nachricht notwendig ist. Folglich besteht eine Nachricht aus einem Empfänger, einem Betreff, der den Grund der Nachricht angeben kann und dem Nachrichteninhalt selbst. Um Rückantworten auf eine erhaltene Nachricht zustellen zu können, ist es erforderlich den Absender der Nachricht mit zu verwalten. Dem Empfänger einer Nachricht muß es möglich sein, nicht mehr benötigende Nachrichten zu löschen.

3.2 Funktionale und dynamische Anforderungen

3.2.1 Präsentation

Da eine Vielzahl von Personen dieses System anwenden werden, muß die Anwendung eine hohe Verfügbarkeit besitzen. Dies wird unter anderem dadurch gewährleistet, daß die Informationen über das WWW präsentiert werden. Mit Hilfe eines geeigneten Browsers können die Benutzer die Anwendung auf den lokalen Rechner laden und ausführen. Aus diesem Grund muß das Anwendungssystem nicht auf den Rechnern der Anwender installiert werden. Ein weiterer Vorteil ist, daß das WWW keine Restriktionen bzgl. des verwendeten Betriebssystems oder der Hardware vorgibt und so die Anwender nicht an eine bestimmte Systemumgebung gebunden sind.

3.2.2 Verwaltung der Daten

Die in diesem System anfallenden Daten werden in einer relationalen Datenbank verwaltet. Dazu müssen die in der Informationsbedarfsanalyse erhaltenen Informationen auf eine Datenbankstruktur abgebildet werden. Da es sich bei dieser Anwendung nicht um ein statisches System handelt, müssen Methoden entwickelt werden, die das Erfassen, Ändern und Löschen von Daten ermöglicht.
Durch den Einsatz einer Datenbank zur Speicherung und Verwaltung der Daten stehen infolgedessen die Mechanismen eines relationalen Datenbanksystems zur Verfügung. Dazu gehören Anfragen, die eine beliebige Selektion der Datenbestände ermöglicht. So ist es z. B. für einer Prüfer einer Veranstaltung interessant, wieviele Studenten sich zur Prüfung angemeldet haben. Eine weitere Anfrage eines Studenten zur Bildung von Lerngruppen könnte sein: „welche Studenten in meinem Semester wohnen !!in derselben Gegend bzw.

haben eine ähnliche Postleitzahl!!“. Da die Anwender dieses System nicht direkt mit der Datenbank arbeiten und deshalb keine individuellen Abfragen formulieren und mittels eines Kommandointerpreters ausführen können, sondern über die Anwendung auf die Daten zugreifen, müssen Verfahren entwickelt werden, mit der flexibel Suchanfragen zu generieren sind.

3.2.3 Kommunikation

Der Zweck dieses Kommunikations- und Präsentationssystems ist nicht nur die Verwaltung und Darstellung von Information, sondern soll auch Information durch Kommunikation bewirken.
Alle aktiven Benutzer dieses Informationssystems werden aus diesem Grund über ein integriertes Kommunikationstool Nachrichten an andere aktive Benutzer senden können. Um Nachrichten zu generieren, muß ein Editor entworfen werden, der es ermöglicht die notwendigen Angaben zu edieren. Der Benutzer kann nicht mehr benötigte Nachrichten löschen. Auf Empfangsseite sollte das Tool zur Übersicht dem Benutzer die Möglichkeit bieten, die gesendeten von den erhaltenen und gelöschten Nachrichten zu unterscheiden.

3.2.4 Sicherheit

In der Informatik wird unter Sicherheit

- den Schutz der Daten [LoSch87],
- den Schutz des Anwendungsprogrammes,
- den Schutz des laufenden Betriebsystems [Tan92] und
- falls vorhanden, den Schutz des Netzwerks [Tan96] verstanden.

Der Schutz des Anwendungsprogrammes, des Betriebssystems und des Netzwerks ist die Aufgabe der umgebenden Systeme. Ziel dieser Arbeit ist die Sicherheit der Daten zu gewährleisten.

Den Schutz der Daten kann nach Chamberlin [Cha96] in drei verschiedene !!Konzepte!! unterteilt werden, dazu gehört:

- die ***Datensicherheit***,
- die ***semantische Datenintegrität*** und
- die ***Datenkonsistenz***.

Datensicherheit

Die Datensicherheit hat die Aufgabe die Daten vor unerlaubten Zugriffen und Modifikationen zu schützen.

Bevor die Strategien erarbeitet werden, die zur Wahrung der Datensicherheit in Betracht kommen, folgt eine Begriffsdefinition. In dieser Begriffsdefinition werden drei Fachbegriffe vorgestellt, deren Inhalt aus einem Beitrag von Reuter in [LoSch87] stammen.

Identifikation: „*Darunter versteht man die Anmeldung eines Benutzers beim System unter Angabe einer Benutzerkennung.*“
In den meisten Anmeldesystemen besteht einer Benutzerkennung aus einem Kennwort (engl.: login) und Paßwort (engl.: password), deren Bedeutung durch den nächsten Begriff verdeutlicht wird.

Authentisierung: „*Dies bezeichnet den Prüfvorgang durch das System, mit dem sichergestellt werden soll, daß ein Benutzer, der sich mit einer bestimmten Kennung anmeldet, auch tatsächlich der zur Führung dieser Kennung der Berechtigte ist.*“

Autorisierung: „*Damit bezeichnet man den Akt der Vergabe von Zugriffsrechten (oder allgemeiner: Nutzungsrechten) an einen Benutzer. Der autorisierende Benutzer muß selbstverständlich die Rechte, die er weitergibt, ebenso besitzen wie das Recht ihrer Weitergabe.*“

Um die Datensicherheit zu gewährleisten, muß das System folgende Aufgaben abdecken:
- Ein Anmeldesystem mit dem die Anwender unter Angabe eines Kennwortes und Paßwortes authentifiziert werden können.
- Die Anwender bzw. Anwendergruppen soll nur die Daten zugänglich gemacht werden, die für ihre Arbeit notwendig ist. Personenbezogene Daten, etwa das Kennwort und Paßwort der Anwender unterliegt besonderen Sicherheitsanforderungen.
- Die Nutzungsrechte der Anwender müssen im System von berechtigten Personen verwaltet werden können.

Semantische Datenintegrität

Die semantische Datenintegrität gewährleistet, daß nur sinnvolle und zulässige Änderungen des Datenbankinhalts durchgeführt werden [LoSch87]. Integritätsregel sollen gewährleisten helfen, daß die Datenbank stets ein korrektes Abbild der Wirklichkeit darstelllt. Dieses Charakteristikum kann durch die Mechanismen „Trigger“ und „Constraints“ des Datenbanksystems unterstützt werden.
Trigger sind Konstrukte, die auf Ereignisse Folgereaktionen durchführen, um die Konsistenz zu erhalten. *Constraints* werden verwendet, um Abhängigkeiten zwischen Datenbanktabellen abzubilden.

Datenkonsistenz

Verlorene Datenänderungen, widerspruchsfreie Datenwerte und Vermeidung von Konflikte durch konkurierende Mehrbenutzersysteme, ist die Aufgabe des Transaktionskonzeptes des Datenbanksystems.

Soll ich hier noch auf die ACID-Eigenschaften eingehen?

3.3 Software-Ergonomie

Die **Software-Ergonomie** hat das Ziel, die Software eines Computersystems, mit der die Benutzer arbeiten, an die Eigenschaften und Bedürfnisse dieser Benutzer anzupassen, um ihnen einen hohen Nutzen vieler relevanter Fähigkeiten und Fertigkeiten zu ermöglichen [Bal96]. Dies führt zu der Erkenntnis, daß die Benutzer solcher Systeme nicht isoliert betrachtet werden dürfen, sondern als Elemente dieser Systeme angesehen werden müssen und deshalb beim Entwurf eine nicht untergeordnete Rolle spielen sollten. Die Arbeitswissenschaft hat versucht die Gestaltungs- und Bewertungskriterien für einen software-ergonomischen Arbeitsplatz zu beschreiben [VDI90], [EG89].

3.3.1 Dialoggestaltung

Die Dialoge sollten so gestaltet werden, daß einerseits Anfänger, die zum ersten Mal mit dem System arbeiten und andererseits Benutzer, die täglich mit dem System arbeiten eine angemessene Umgebung vorfinden.
In empirischen Untersuchungen hat der Verband des Deutschen Instituts für Normung e. V. dazu fünf Grundsätze zur Dialoggestaltung herausgegeben [DIN88]:

- **Aufgabenangemessenheit**: „Ein Dialog ist *aufgabenangemessen*, wenn er die Erledigung der Arbeitsaufgabe durch den Benutzer unterstützt, ohne ihn durch Eigenschaften des Dialogsystems unnötig zu belasten."

- **Selbstbeschreibungsfähigkeit**: „Ein Dialog ist *selbstbeschreibungsfähig*, wenn dem Benutzer auf Verlangen Einsatzzweck sowie Leistungsumfang des Dialogsystems erläutert werden können und wenn jeder einzelne Dialogschritt unmittelbar verständlich ist oder dem Benutzer auf Verlangen, dem jeweiligen Dialogschritt entsprechende Erläuterungen, erhalten kann."

- **Steuerbarkeit**: „Ein Dialog ist *steuerbar*, wenn der Benutzer die Geschwindigkeit des Ablaufs sowie die Auswahl und Reihenfolge von Arbeitsmitteln oder Art und Umfang von Ein- und Ausgaben beeinflussen kann."

- **Erwartungskonformität**: „Ein Dialog ist *erwartungskonform*, wenn er den Erwartungen der Benutzer entspricht, die sie aus Erfahrungen mit bisherigen Arbeitsabläufen oder aus der Benutzerschulung mitbringen sowie den Erwartungen, die sich während der Benutzung des Dialogsystems und im Umfang mit dem Benutzerhandbuch bilden."

- **Fehlerrobustheit**: „Ein Dialog ist *fehlerrobust*, wenn trotz erkennbar fehlerhafter Eingaben das beabsichtigte Arbeitsergebnis mit minimalem oder ohne Korrekturaufwand erreicht wird. Dazu müssen dem Benutzer die Fehler zum Zwecke der Behebung verständlich gemacht werden."

3.3.2 Ein-/Ausgabe-Gestaltung

Das Ziel der Ein-/Ausgabe-Gestaltung ist die Informationseingabe und die Informationsausgabe so zu gestalten, daß die Eigenschaften der menschlichen Informationsverarbeitung geeignet berücksichtigt werden [Ebe92], [May92].

Dazu soll
- die Orientierung des Benutzers unterstützt,
- seine Aufmerksamkeitserfordernisse optimiert,
- sein Kurzzeitgedächtnis entlastet werden und
- jede Anwendung sich regelhaft verhalten.

Durch systematische Ablaufstrukturen in der Dialogführung kann dem Benutzer die Orientierung erleichtert werden. So müssen ähnliche Aufgaben (Erfassen eines Studenten bzw. Erfassen eines Mitarbeiters) auch ähnliche Regeln zur Ausführung dieser Tätigkeit haben.

Verschiedene Prozesse der Informationsverarbeitung und Handlungssteuerung erfordern Aufmerksamkeit. Um die Aufmerksamkeitserfordernisse zu optimieren muß die Benutzeroberfläche so gestaltet werden, daß sie die Aufmerksamkeit des Benutzers entsprechend den zu erledigenden Aufgaben und den möglichen Bedienungsfertigkeiten entspricht.

Das Kurzzeitgedächtnis dient der kurzfristigen Speicherung einer begrenzten Menge von Informationen. Das Ziel der Gestaltung lautet daher, die geringe Speicherkapazität und kurze Speicherdauer des Kurzzeitgedächtnisses zu berücksichtigen.

Die Anwendung verhält sich bzgl. der E/A-Gestaltung regelgerecht, wenn sie nach festen, einheitlichen und eindeutigen Regeln abläuft, die für den Benutzer erkennbar sind.

4 Vorüberlegungen

4.1 Einleitung

Wie zu Beginn der Anforderungsanalyse erwähnt, kann ein zu entwerfendes System in vier verschiedene Sichten unterteilt werden. Um die einzelnen Sichten zu entwerfen, wurden seit Mitte der siebziger Jahre verschieden Konzepte entwickelt:

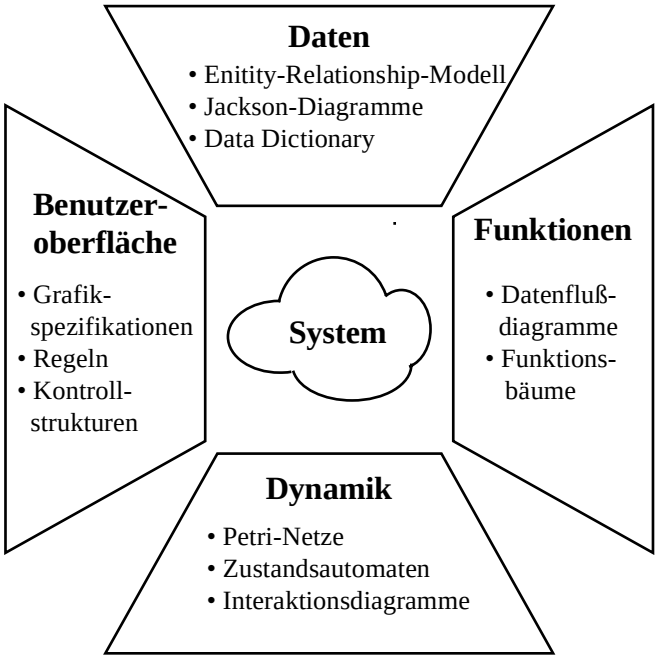


Abbildung 2: Entwurfskonzepte

Durch die zeitlichen Restriktionen können nicht alle Verfahren in dieser Arbeit erläutert werden. Deshalb wird bzgl. der verschiedenen Sichten auf folgende Literatur verwiesen:

Daten-Sicht

- Entity-Relationship-Modell [Chen76], [LaLo95]
- Jackson-Diagramme [Jack75],
- Data Dictionary [Mar79],

aus der Sicht der **Funktionen**

- Datenflußdiagramme [Mar79],
- Funktionsbäume [Jack75], [Mar79],

Dynamik

- Petri-Netze [Rei82],
- Zustandsautomaten [Hop90],
- Interaktionsdiagramme [Jac92], [Boo94]

und der **Benutzeroberflächen**-Sicht

- Grafikspezifikationen [DIN88], [EG89], [VDI90], [May92]
- Regeln und Kontrollstrukturen [Jack75], [Warr78]

Die Aufgabenstellung der vorliegenden Arbeit sieht vor, die Implementierung der Benutzeroberfläche, sowie die Funktionen und deren Dynamik in der objektorientierten Programmiersprache Java zu erstellen und die Daten in einer geeigneten relationalen Datenbank abzulegen. Infolge dieser Konstellation sind Transformationen notwendig, die den *objektorientierten Entwurf* und den *logische Datenbankentwurf* geeignet abbilden. Bevor diese Tatsache näher erläutert wird, folgt eine Übersicht über die Merkmale des Entwurfs eines relationalen Datenschemas und dem eines objektorientierten.

4.2 Entwurf eines relationalen Datenbankschemas

In diesem Kapitel werden die grundlegenden Begriffe des relationalen Datenmodells definiert. Anschließend folgt eine Einführung in die Vorgehensweise eines Datenbankentwurfs.

4.2.1 Definitionen des relationalen Datenmodells

Das relationale Datenmodell wird durch vier Begriffe in Reuter [Reu93] definiert:

- **Attribute** A_i sind beschreibende und identifizierbare Aspekte von Objekten bzw. Beziehungen der abzubildenden Miniwelt.

- Ein **Wertebereich** $W(A_i)$ ist die Menge von Werten, aus der alle Ausprägungen des Attributes A_i stammen. Es sind ausschließlich atomare Wertebereiche zulässig.

- Eine **Relation** $R(A_1, A_2, ..., A_n)$ ist durch Angabe ihres Namens R und der beteiligten Attribute $A_1, ..., A_n$ in ihrer Struktur zeitinvariant festgelegt.

- Ein **Primärschlüssel** der Relation R ist eine minimale Attributkombination, mit der Eigenschaft, daß in allen möglichen Ausprägungen von R jeder Wert nur einmal vorkommt.

4.2.2 Konzeptioneller Entwurf des Datenbankschemas

Für den konzeptionellen Entwurf des Datenbankschemas wird das Entity-Relationship-Modell (ERM) verwendet. Das ERM wurde 1976 erstmalig von Chen [Che76] vorgestellt und stellt ein Mittel zur Informationsmodellierung dar. Dieses semantische Datenmodell findet nicht nur im sogenannten Datenbankdesign seine Anwendung, sondern auch bei der Unternehmensmodellierung [LoSch87], [LaLo95], [Sch90].

Folgende Begriffe werden im ERM verwendet, deren Definitionen von Reuter [Reu93] übernommen werden:

- Unter einer **Entität** versteht man die kleinste informationstragende Einheit, die unabhängig von anderen Objekten aus eigenen Rechten existenzfähig ist, sobald sie identifiziert ist.
- Eine Entität hat Eigenschaften, die durch **Attribute** beschrieben werden.
- Besteht zwischen zwei oder mehreren Entitäten eine Beziehung, so wird dies durch eine **Relationship** zum Ausdruck gebracht.

Zur Darstellung von Entitäten werden die sogenannten **Entity-Relationship-Diagramme** (ERD) eingesetzt, mit deren Hilfe der konzeptionelle Entwurf eines Datenbankschemas übersichtlich dargestellt werden kann. Dabei wird folgende graphische Notation eingesetzt:

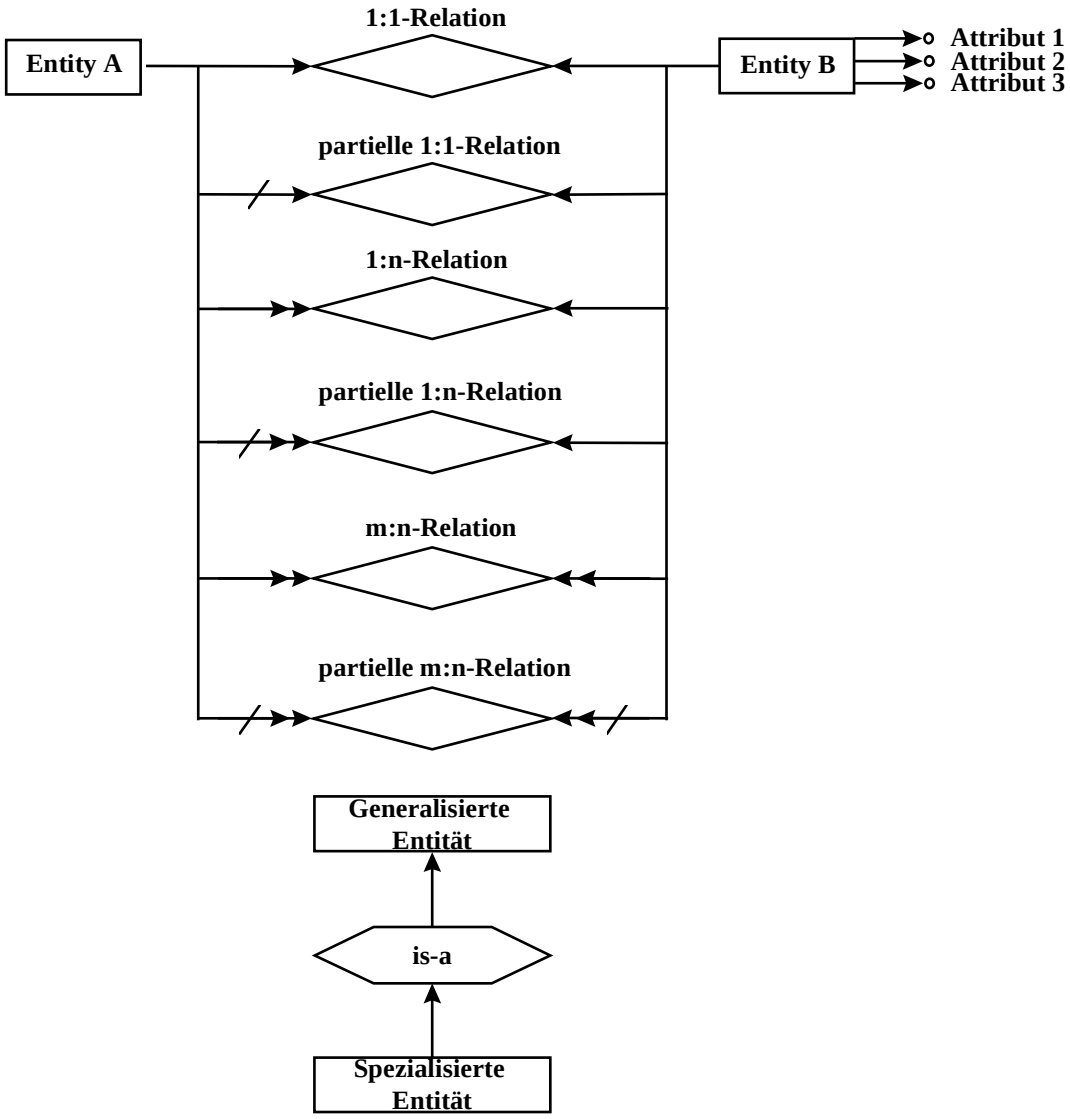


Abbildung 3: Entity-Relationship-Diagramm (ERD)

4.2.3 Logischer Entwurf des Datenbankschemas

Im logischen Entwurf wird das Entity-Relationship-Modell auf ein Relationenmodell abgebildet, normalisiert und das dadurch erhaltene Ergebnis an die Datenbanksystem (DBS)-Umgebung angepaßt.

Bei der Abbildung gelten folgende Regeln:

- Es wird für jedes Entity eine eigene Tabelle spezifiziert.
- Die einzelnen Attribute eines Entities erhalten eine Spalte in der Tabelle, die für das Entity spezifiziert ist.
- Jedem Entity wird ein künstliches, eindeutiges Schlüsselattribut zu seiner Identifikation zugeordnet.
- Bei 1:1-Relationen erhält eine der beteiligten Entitäten einen Verweis auf die andere Entität.
- Im Falle von 1:n-Relationen erhält diejenige Entität den Verweis, bei der dieser eindeutig ist.
- Für m:n-Relationen wird eine eigene Tabelle angelegt, die einen Fremdschlüssel aus den beteiligten Entitäten erhält.
- Generalisierung bzw. Spezialisierung wird durch die Verwendung des Schlüssels der allgemeineren Entität in der Tabelle der spezielleren Entität zum Ausdruck gebracht.

Die beim logischen Entwurf erhaltenen Tabellen befinden sich in der Regel in der ersten Normalform. Die erste Normalform hat die Eigenschaft, daß alle Attribute atomar sind. Diese Eigenschaft reicht jedoch nicht aus, um einen konfliktfreien Datenbankentwurf zu gewährleisten. Deshalb werden in der Praxis die Tabellen bis zur dritten Normalform transformiert.

4.3 Objektorientierter Entwurf

In diesem Kapitel werden nur die für diese Arbeit notwendigen Gesichtspunkte des objektorientierten Entwurfs dargestellt. Für weiterführende Informationen werden die interessierten Leser auf Booch [Boo94] und Coad/Yourdon [CoYo91] verwiesen.

Der objektorientierte Entwurf (**OOD, Object Oriented Design**) läßt sich nach Balzert [Bal96] in zwei Schritte unterteilen:

- den **Architekturentwurf**,
- und den **Implementierungsentwurf**.

Das wesentliche Ziel des Architekturentwurfes ist die Anbindung der Fachkonzeptklassen an die Benutzeroberfläche und an die Datenhaltung, wohingegen im Implementierungsentwurf der Architekturentwurf verfeinert und an die verwendete Programmiersprache angepaßt wird. In den folgenden zwei Kapitel werden die jeweiligen Schritte erläutert.

4.3.1 Architekturentwurf

Grundlagen für den objektorientierten Architekturentwurf sind die Ergebnisse aus der Anforderungsanalyse und die Berücksichtigung von Schnittstellen aus umgebenden Systemen. Bei diesen Schnittstellen kann es sich unter anderem um Systemfunktionen des Betriebssystems oder Kommunikationskomponenten für verteilten Anwendungen handeln.

Die Aufgaben des Architekturentwurfs sind:

- die Anbindung an die Benutzeroberfläche und
- die Anbindung an die Datenhaltung.

Bei der Anbindung von *Fachkonzeptklassen* (entstandene Klassen aus Entities) an die Benutzeroberfläche müssen Beziehungen hergestellt werden, die die Verbindungen zwischen den **GUI (Graphical User Interface) -** Elementen, wie beispielsweise Textfelder und den Attributen aus den *Fachkonzeptklassen* repräsentieren.

Die folgende Abbildung verdeutlicht diesen Sachverhalt:

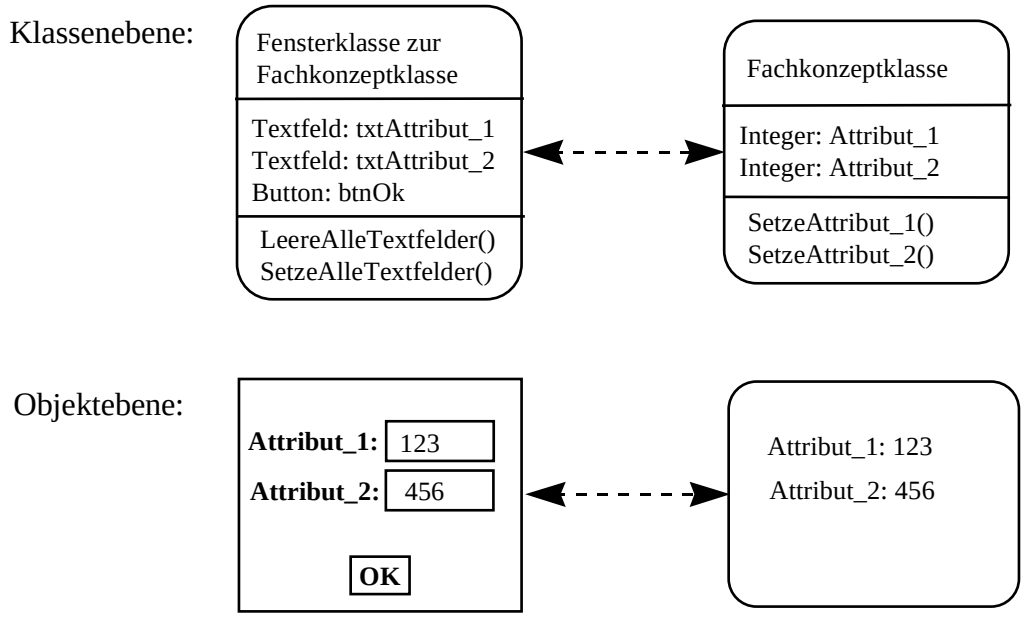


Abbildung 4: Anbindung der Fachkonzeptklassen an die Benutzeroberfläche

Durch Auslösen von Aktionen an GUI-Elementen können Methoden der Fensterklasse die Methoden der Fachkonzeptklasse aufrufen und somit deren Objekte manipulieren.

Die Anbindung an die Datenhaltung wird durch die Art der Datenhaltung bestimmt, wobei es sich um

- ein Dateisystem,
- eine objektorientiertes Datenbanksystem oder
- ein relationales Datenbanksystem handeln kann.

Bei Verwendung eines Dateisystems für die Datenhaltung werden die Verfahren der Dateiorganisation angewendet. Dabei ist die Speicherungsform des Dateisystems zu berücksichtigen. Dabei kann man zwischen sequentiellen, indexsequentiellen oder hash-ähnlichen Verfahren unterscheiden.
Im Falle eines objektorientierten Datenbanksystems ist die Anbindung relativ einfach. Die *Fachkonzeptklassen* erhalten entsprechende Klassenstrukturen, die durch geeignete Schnittstellen ergänzt werden müssen.
Diese beiden erstgenannten Arten der Anbindung werden hier nicht weiter verfolgt, da die Datenhaltung in dieser Arbeit durch ein relationales Datenbanksystem realisiert wird. Weitere Informationen im Falle des objektorientierten Datenbanksystems sind z.B. in Catell [Cat96] zu erhalten.
Bei der Anbindung einer relationalen Datenbank an eine objektorientierte Anwendung müssen einige Aufgaben durchgeführt werden, die im weiteren erläutert werden. Hahn, Toenniessen und Wittkowski [HaToWi95] übergeben diese Aufgaben einer objektorientierten Zwischenschicht, die den Zugriff zwischen objektorientierter Anwendung und relationaler Datenbank regelt.

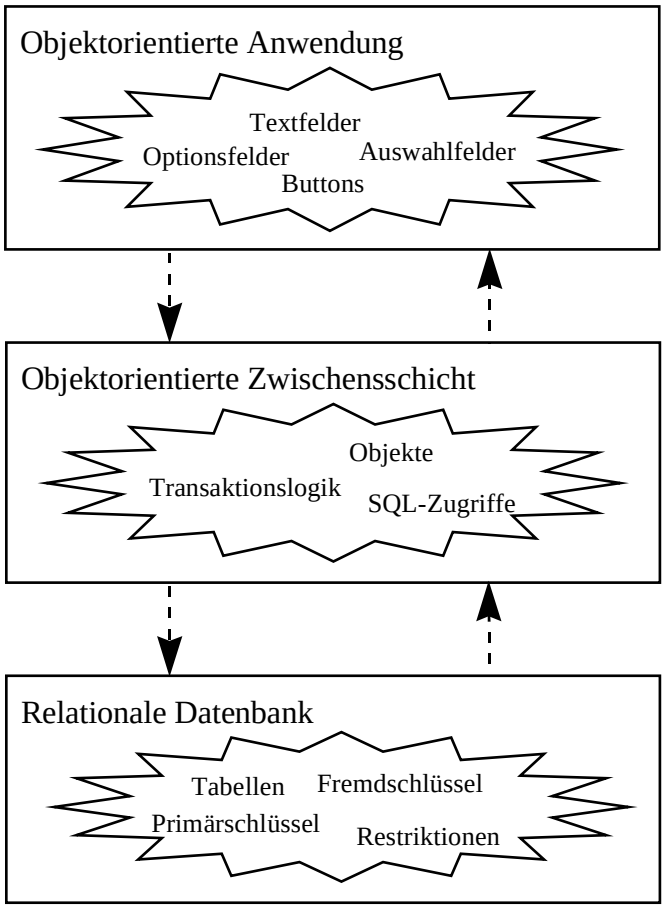


Abbildung 5: Funktionsweise der objektorientierten Zwischenschicht

Diese Zwischenschicht besitzt folgende Aufgaben:

- **Abbildung zwischen Klassen und Tabellen:** Jede Klasse wird mit ihren Attributen auf eine Tabelle mit den entsprechenden Spalten abgebildet.
  Um die Effizienz für den Datenzugriff zu erhöhen kann eine *vertikale* oder *horizontale Unterteilung* der Tabellenattribute nach Rumbaugh [Rum91] hilfreich sein.
  Eine **vertikale Unterteilung** ist dann sinnvoll, wenn eine Klasse viele Attribute besitzt und nur auf wenige dieser Attribute häufig zugegriffen wird. Dabei werden die Attribute, auf die häufig zugegriffen werden in einer eigenen Tabelle abgelegt und durch eine 1:1-Beziehung mit den übrigen Attributen verknüpft.
  Im anderen Fall bietet sich eine **horizontale Verteilung** an, wenn auf bestimmte Datensätze oft zugegriffen werden. Diese Datensätze werden in einer eigenen Tabelle abgelegt.

- **Abbildung von Assoziationen und Aggregationen:** Die Abbildungen unterscheiden sich in den Kardinalitäten:
  Die 1:1-Beziehungen werden durch die jeweiligen Fremdschlüssel in einer der beiden Tabellen realisiert.
  1:m-Beziehung werden durch einen Fremdschlüssel in der m-Klasse modelliert.
  Alle m:n-Beziehung werden auf eine eigene Tabellen abgebildet, die die Fremdschlüssel als Attribute enthalten.

- **Abbildungen von Vererbungen:** Bei der einfachen Vererbung bieten sich drei verschiedene Möglichkeiten an:

  1. Oberklassen und ihre dazugehörige Unterklassen werden jeweils auf eine eigene Tabelle abgebildet, wobei eine Oberklasse und ihre Unterklassen denselben Primärschlüssel besitzen. Ein Vorteil dieses Verfahrens ist der Erhalt des minimalen Datenbankschemas. Nachteilig kann sich einerseits die langsame Navigation zwischen Oberklasse und den Unterklassen und anderseits der mehrmalige Zugriff auf Tabellen desselben Objektes auswirken.

  2. Alle Attribute einer Oberklasse werden in jede Unterklassentabelle übernommen. Dies hat den Nachteil, daß die Eindeutigkeit der Oberklassenattribute nicht eingehalten werden kann.

  3. Alle Unterklassenattribute werden in die Oberklassentabelle übernommen. Diese Möglichkeit der Auflösung hat den Nachteil, daß die dritten Normalform verletzt wird. Die Abbildung von Mehrfachvererbungen wird nicht weiter verfolgt, da die Programmiersprache Java sie nicht unterstützt und somit auch nicht Gegenstand dieser Arbeit ist. Rumbaugh behandelt diesen Mechanismus in ausführlicher Weise und kann in [Rum91] nachgelesen werden.

- **Ausführen von Abfragen mit SQL (Standard Query Language):** Um Abfragen aus der objektorientierten Zwischenschicht auf die relationale Datenbank zu gewährleisten, bietet sich eine eigene Klasse für diese Aufgabe an. In dieser Klasse werden auch die spezifischen Merkmale der Datenbank hinterlegt.

4.3.2 Implementierungsentwurf

Im objektorientierten Implementierungsentwurf wird die Festlegung des Architekturentwurfs verfeinert und die Eigenschaften der verwendeten Programmiersprache berücksichtigt. Es werden die Attribute und Methoden der kreierten Klassen definiert. In der vorliegenden Arbeit wird die Programmiersprache Java als Zielsprache verwendet. Java besitzt die integrierte Datenbankschnittstelle (JDBC, Java Database Connectivity) mit der es möglich ist auf relationale Datenbanksysteme (DBS) zuzugreifen. In diesem !!Zustand!! des Entwurfs muß auch das verwendete Datenbanksystem berücksichtigt werden, d.h. es erfolgt die Abstimmung mit den unterstützenden Datentypen des DBS.

5 Entwurf des Kommunikations- und Präsentationssystems

Im Entwurf werden die in der Anforderungsanalyse erworbenen Kenntnisse in eine strukturierte Form gebracht. Zu Beginn werden die Informationen aus der Informationsbedarfsanalyse auf zusammenhängenden Einheiten und deren Beziehungen untersucht. Mit Hilfe des Enitity-Relationship-Modells werden diese Charakteristika in einer graphischen Notation dargestellt. Anschließend erfolgt der Entwurf des Datenbankschemas in Kapitel 5.2. Im darauffolgenden Kapitel 5.3 wird die Funktionalität und die Dynamik des Systems entworfen. Die dabei erstellten Klassen und ihre Beziehungen werden durch die graphische Notation von Coad/Yourdon [CoYo91] zum Ausdruck gebracht. Das Kapitel schließt mit dem Entwurf der Benutzeroberfläche.

5.1 Die Datenhaltung

In diesem Kapitel werden die in der Informationsbedarfsanalyse erhaltenen Informationen auf ein semantisches Datenmodell abgebildet. Dabei wird das Entity-Relationship-Modell (ERM) verwendet, um die Entities und ihre Beziehungen zueinander zu definieren.

Es ist wurde nicht die vollständige Struktur einer Universität samt ihrer Veranstaltungen abgebildet. Es ist auch nicht das Ziel dieser Arbeit die Studienpläne der einzelnen Fakultäten in dieses System zu integrieren. Diese Thematik auf ein Datenbankschema abzubilden wäre sehr komplex geworden und aufgrund ständig verändernder Studienpläne schwer zu realisieren. Aus diesem Grund wurde unabhängig von geltenden Studienplänen eine für diese Aufgabe ausreichende Universitätsstruktur und einige grundlegende Veranstaltungstypen abgebildet.

Zur Notation:

Zur Übersicht werden die einzelnen Bereiche in Teilschemen unterteilt. Da einige Teilschemen immer noch eine sehr umfangreiche Struktur darstellen, werden sie zum besseren Verständnis in zusammengehörende Bausteine zerlegt und beschrieben.

Um eine übersichtliche Darstellungsweise zu gewähren, werden in der Beschreibung
- Entitäten und Relationen als kursiv und fett,
- und deren Attribute als kursiv

dargestellt.

Am Ende eines jeden Teilschemas erfolgt eine Zusammenfassung in Form einer Grafik, sowie der Entitäten und Relationen zzgl. ihrer Attribute in Tabellenform.

Die Teilschemen werden nun in den folgenden Kapitel vorgestellt :

5.1.1 Das Teilschema Universitätsstruktur

Die Universitätsstruktur beschreibt den inneren hierachischen Aufbau der Institution ***Universität***. Sie besteht aus ***Fakultäten***, die aus ***Instituten*** und diese wiederum aus ***Abteilungen*** bestehen.

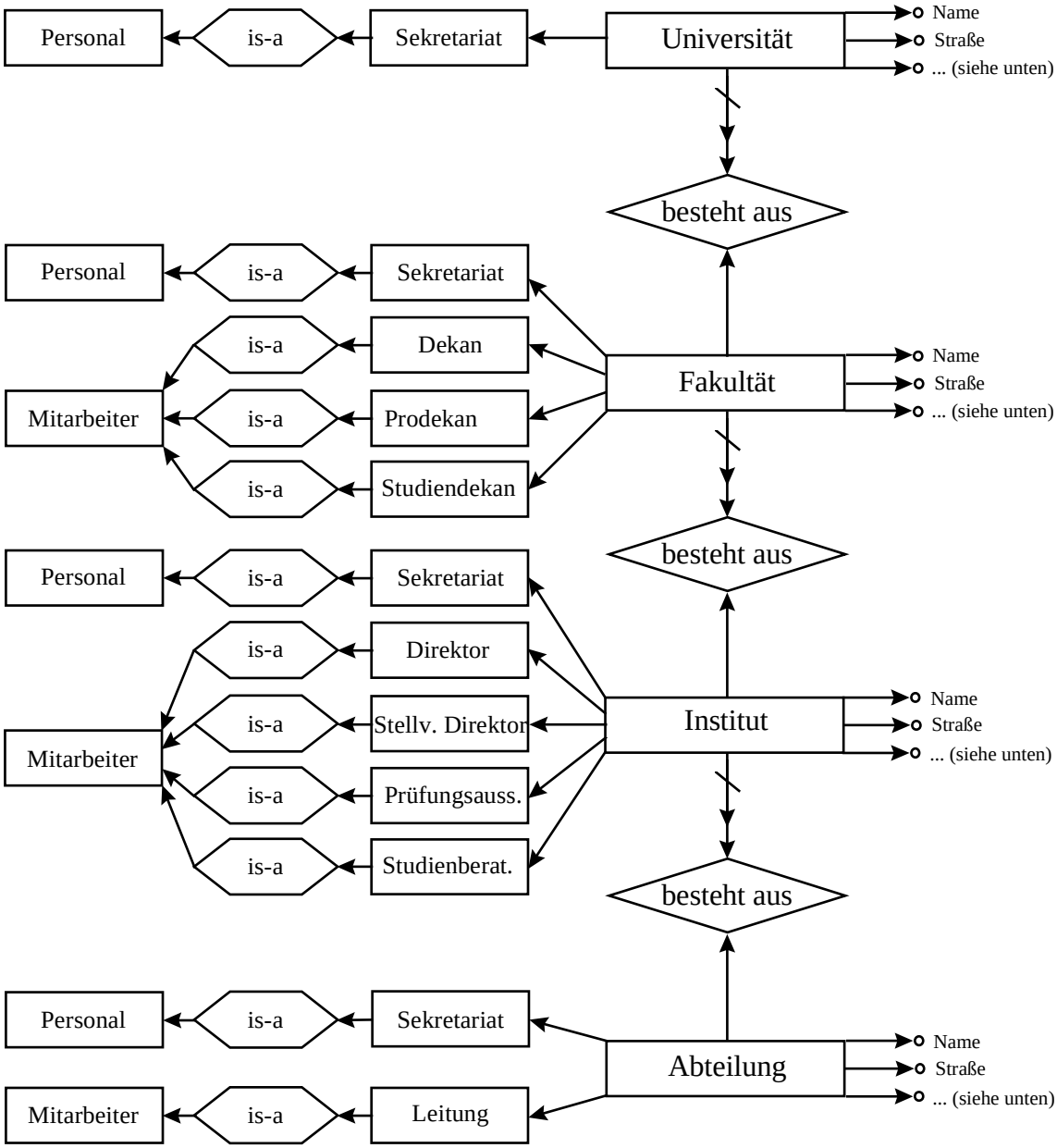


Abbildung 5: ERD Teilschema Universitätsstruktur

Jede der Entitäten ***Universität***, ***Fakultät***, ***Institut*** und ***Abteilung*** besitzt folgende Attribute:

- *Name*: Dieses Attribut dient zur Bezeichnung der jeweiligen Entität.
- *Strasse, Plz* und *Ort*: Diese Angaben dienen zur Bestimmung der Adresse.
- *Sekretariat*: Dieses als Entität fungierende Attribut enthält seinerseits die Attribute *Zimmer*, *Telefon, Sprechstunde* und die *Person*, die diese Stelle ausübt.

Darüber hinaus besitzt die Entität ***Fakultät*** noch die Attribute *Dekan, Prodekan* und *Studiendekan*, die Entität ***Institut*** die Attribute *Direktor, Stellvertrende Direktor, Prüfungsausschuß* und *Studienberatung* und die Entität ***Abteilung*** das Attribut *Leitung*.

Zwischen den Entitäten ***Universität***, ***Fakultät***, ***Institut*** und ***Abteilung*** liegen die Relationen ***besteht aus***, die das hierarchische Verhalten darstellt.

Zusammenfassung der Entitäten des Teilschemas Universitätsstruktur:

Name:	**Attribute:**
Abteilung	Name, Straße, Plz, Ort, Leitung, Sekretariat
Fakultät	Name, Straße, Plz, Ort, Dekan, Prodekan, Studiendekan, Sekretariat
Institut	Name, Straße, Plz, Ort, Direktor, Stellvertetende Direktor, Sekretariat, Prüfungsausschuß, Studienberatung,
Universität	Name, Straße, Plz, Ort, Sekretariat

Abbildung 6: Entitäten des Teilschemas Universitätsstruktur

Zusammenfassung der Relationen des Teilschemas Universitätsstruktur:

Name:	**Attribute:**	**Beziehungstyp:**	**beteiligte Entitäten:**
besteht aus	-	1:n (partiell)	Universität, Fakultät
besteht aus	-	1:n (partiell)	Fakultät, Institut
besteht aus	-	1:n (partiell)	Institut, Abteilung

Abbildung 7: Relationen des Teilschemas Universitätsstruktur

5.1.2 Das Teilschema Personen

Das Teilschema Personen spezifiziert die verschiedenen Typen von Personen, die eine ***Universität*** beinhaltet, sowie dessen Eigenschaften.
Wie in der Anforderungsanalyse festgestellt gibt es drei verschiedene Typen von Personen:

- ***Studenten***,
- ***Mitarbeiter***, die für das Vermitteln von Wissen zuständig sind und
- ***Personal***, das verwaltungstechnische Aufgabe wahrnimmt.

Die verschieden Typen dieser Personen haben gemeinsame Attribute, dazu zählen die *Anrede*, der *Vorname* und *Nachname*, sowie die Anschrift, die sich aus den Attributen *Strasse*, *Plz* und *Ort* zusammensetzt und eventuell die Nummer eines privaten Telefonanschlusses. Weiterhin kann jeder Typ von Personen eine *Mail-Adresse* besitzen, die für die Benutzerkommunikation notwendig ist. Aufgrund dieser gemeinsamen Attribute wird ein Entity mit dem Namen ***Person*** eingeführt, das diese Attribute enthält.
Durch dieses Generalisierung folgt eine *is-a-Beziehung* mit ***Person*** als übergeordnete Entität und den untergeordneten Entitäten ***Student***, ***Mitarbeiter*** und ***Personal***:

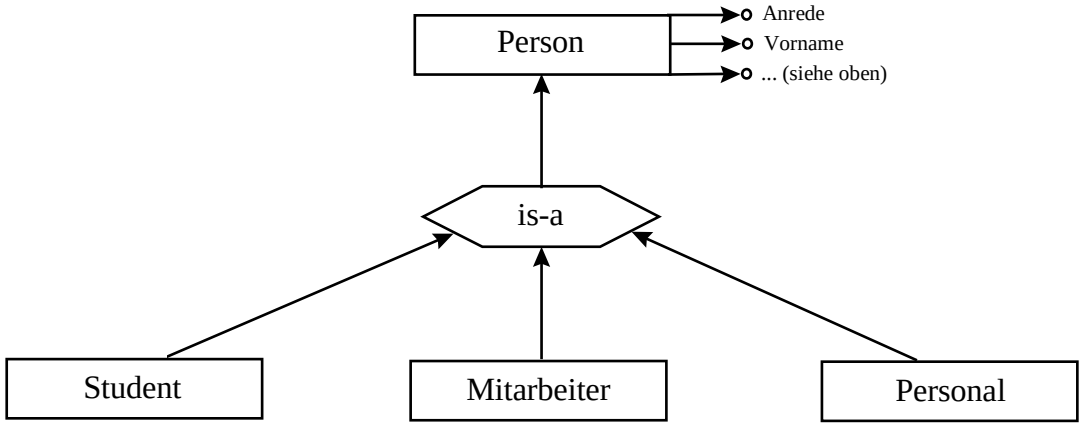


Abbildung 8: ERD Personen01

Im weiteren wird nun auf die einzelnen spezialisierten Enitäten ***Student***, ***Mitarbeiter*** und ***Personal*** eingegangen:

Die Entität *Student*

Jeder ***Student*** ist an einer ***Universität*** immatrikuliert. Dieser Sachverhalt wird durch die Relation ***studiert an*** verdeutlicht, die die Entitäten ***Student*** und ***Universität*** in Beziehung setzt.
Durch die Relation ***begonnen*** kann festgestellt werden, in welchem ***Semester*** ein ***Student*** sein Studium aufgenommen hat.
Jeder immatrikulierte ***Student*** studiert ein ***Studienfach***. In besonderen Fällen kann ein Student zwei Studienfächer belegen.

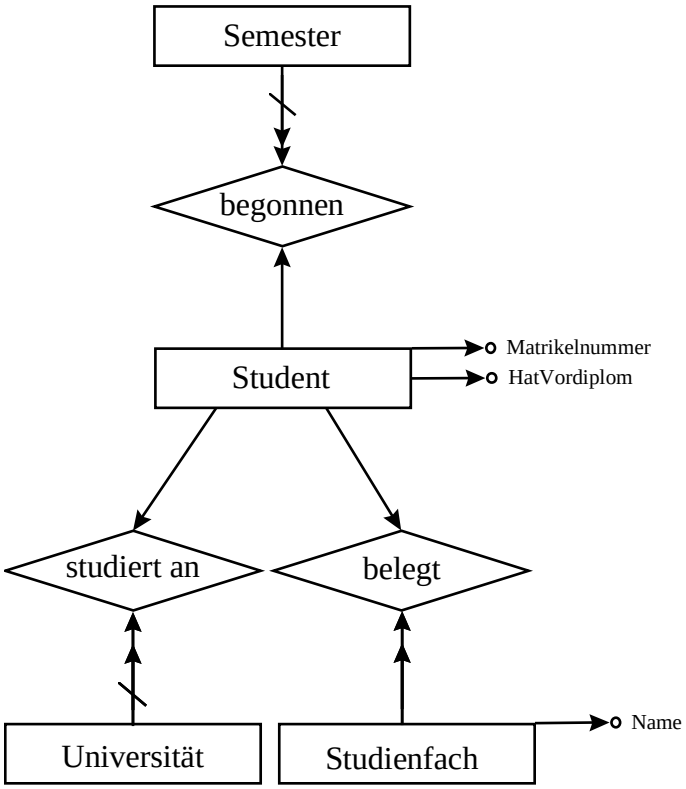


Abbildung 9: ERD Personen02

Ein ***Student*** besitzt folgende zusätzliche Attribute:

- eine eindeutige *Matrikelnummer* und
- eine Eigenschaft mit dem Namen *HatVordiplom* aus dem ersichtlich ist, ob der Student im Besitz des Vordiplomes ist.

Die Entität *Mitarbeiter*

Mitarbeiter haben unter anderem die Aufgabe Wissen zu vermitteln. Aus diesem Grund werden zu der Entität ***Mitarbeiter*** Personen wie z.B. Professoren, Dozenten und sonstige wissenschaftlicher Mitarbeiter zugeordnet, das durch das Attribut *Typ* festgelegt wird. Weitere Attribute eines Mitarbeiters sind:

- der *Titel*, den er führt,
- das *Telefon*,
- das *Zimmer*,
- Angaben über die *Sprechstunde*,
- die Eigenschaft *lehrberechtigt*, die festlegt, ob ein Mitarbeiter außer den Forschungsaufgaben auch eine Lehrtätigkeit wahrnimmt und
- die Eigenschaft *prüfberechtigt*, die bestimmt, ob ein Mitarbeiter das Recht hat Studenten zu prüfen.

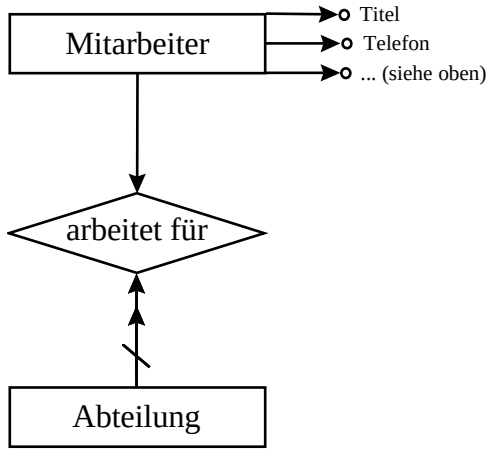


Abbildung 10: ERD Personen03

Die Relation ***arbeitet für*** kennzeichnet, daß jeder *Mitarbeiter* zu einer ***Abteilung*** gehört, in der er seine Arbeit macht.

Die Entität *Personal*

Die Entität ***Personal*** besitzt außer den Attributen der Entität ***Person*** noch das Attribut *Einstellungsdatum*, das den Beginn der Beschäftigung charakterisiert.

Zusammenfassung des Teilschemas Personen

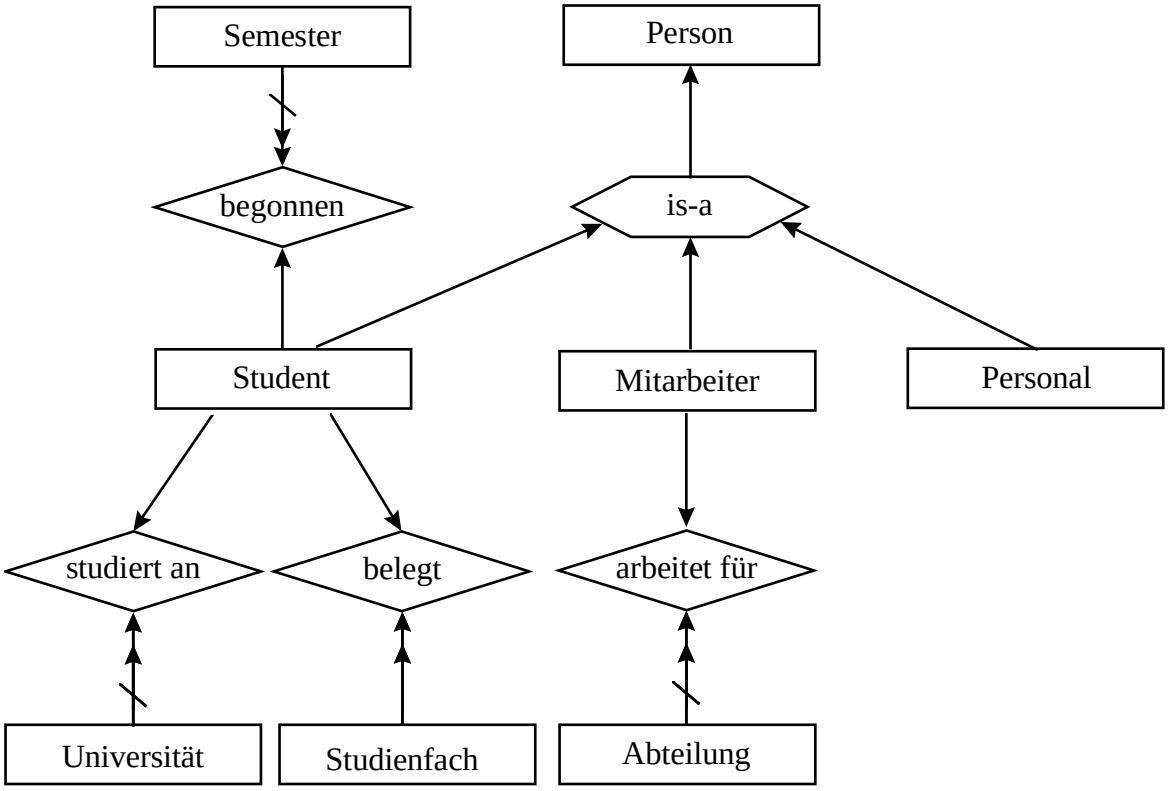


Abbildung 11: ERD Teilschema Personen

Zusammenfassung der Entitäten des Teilschemas Personen:

Name:	**Attribute:**
Abteilung	siehe Teilschema *Universitätsstruktur*
Mitarbeiter	Titel, Telefon, Zimmer, Sprechstunde, Typ, lehrberechtigt, prüfberechtigt
Person	Anrede, Vorname, Nachname, Mail-Adresse, Straße, Plz, Ort, Privat-Telefon, Status, Geburtsdatum, Login, Passwort
Personal	Einstellungsdatum
Seminar	Seminar
Student	Matrikelnummer, HatVordiplom, StudienBeginn
Studienfach	Name
Universität	siehe Teilschema *Universitätsstruktur*

Abbildung 12: Entitäten des Teilschemas Personen

Zusammenfassung der Relationen des Teilschemas Personen:

Name:	**Attribute:**	**Beziehungstyp:**	**beteiligte Entitäten:**
arbeitet für	-	1:n (partiell)	Mitarbeiter, Abteilung
begonnen	-	1:n (partiell)	Student, Semester
belegt	-	1:n	Student, Studienfach
studiert an	-	1:n (partiell)	Student, Universität

Abbildung 13: Relationen des Teilschemas Personen

5.1.3 Das Teilschema Benutzerkommunikation

Das Informationssystem wird durch eine Kommunikationskomponente erweitert, damit beispielsweise die Teilnehmer eines Seminars, das an verschiedenen Orten stattfinden kann, miteinander kommunizieren können.

Mit Hilfe der *Email-Adresse* kann an jede ***Person*** eine ***Nachricht*** gesendet werden. Die Relation ***sendet*** setzt die Entitäten ***Person*** und ***Nachricht*** in Beziehung, damit zu jeder ***Nachricht*** eindeutig ein Sender zugeordnet werden kann.

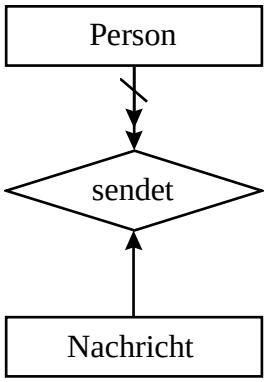


Abbildung 14: ERD Benutzerkommunikation01

Die Entität ***Nachricht*** wird durch die Attribute

- *Betreff*,
- *Datum* und
- *Inhalt* charakterisiert.

Durch das Attribut *Betreff* kann der Sender der ***Nachricht*** den Grund beschreiben, das Attribut *Datum* legt die Zeit fest, zu der die ***Nachricht*** gesendet wird und das Attribut *Inhalt* enthält den eigentlichen Nachrichteninhalt.

Eine ***Nachricht*** kann an einen oder mehrere ***Empfänger*** zugestellt werden, das mittels der Relation ***erhält*** erreicht wird.

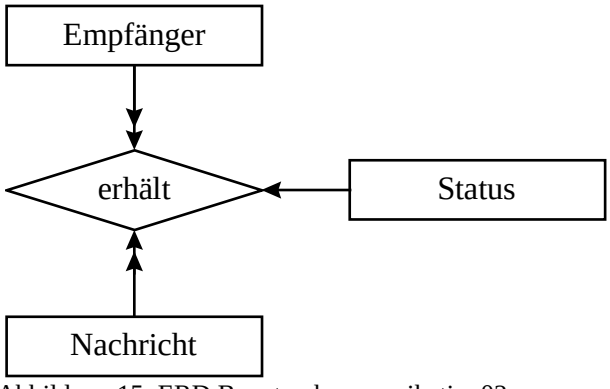


Abbildung 15: ERD Benutzerkommunikation02

Die Relation wird durch ein Attribut mit dem Namen *Status* versehen, dessen Wert vier verschiedene Zustände annehmen kann.

Die vier Zustände werden nun im Einzelnen beschrieben:

Zustand 1
Dieser Zustand beschreibt die Konstellation, daß ein ***Empfänger*** die ***Nachricht*** noch nicht gesehen hat.

Zustand 2
Wenn der Empfänger die Nachricht aus einer Übersicht ausgewählt hat, um sie zu lesen, wird der Wert des Attributes *Status* mit dem zweiten Zustand belegt.

Zustand 3
Der dritte Zustand tritt ein, wenn der Empfänger die Nachricht gelesen und beschlossen hat, sie in den Ordner für gelöschte Nachrichten zu verschieben.

Zustand 4
Wenn sich die Nachricht im Ordner für gelöschte Nachrichten befindet, und der Empfänger die Nachricht endgültig löschen möchte, erhält das Attribut den Zustand vier. Diese Nachricht ist nun für den Empfänger nicht mehr erreichbar.

Wie schon erwähnt, kann es sich bei dem oder den Empfängern um eine einzelne ***Person*** oder eine ***Personengruppe*** handeln.

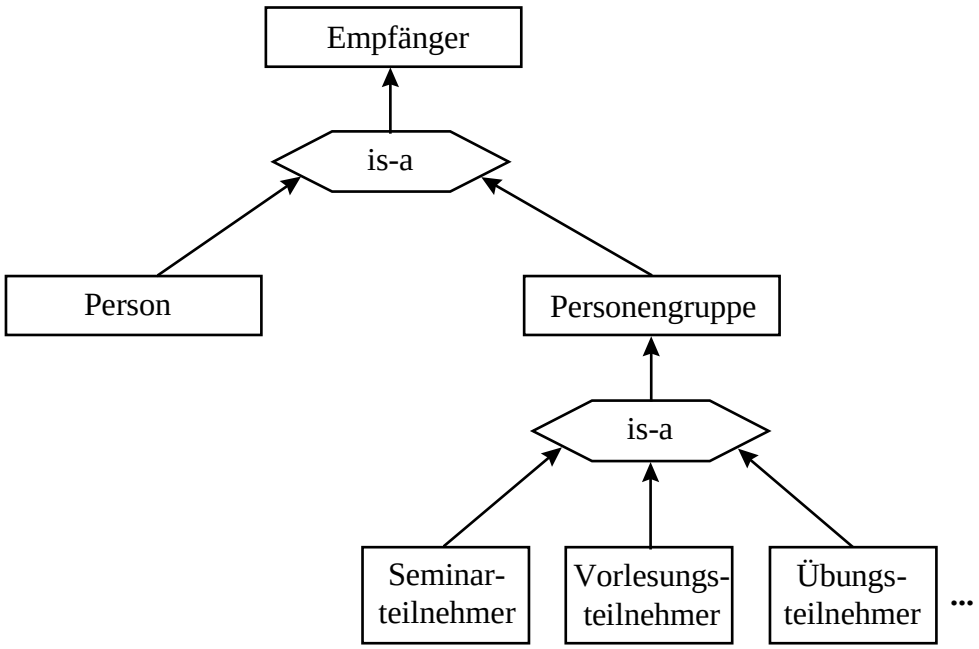


Abbildung 16: ERD Benutzerkommunikation03

Eine ***Personengruppe*** wird repräsentiert durch die Teilnehmer eines Seminars, einer Vorlesung oder einer Übung. Die einzelnen Gruppentypen werden in den nächsten Teilschemen näher erläutert.

Zusammenfassung des Teilschemas Benutzerkommunikation

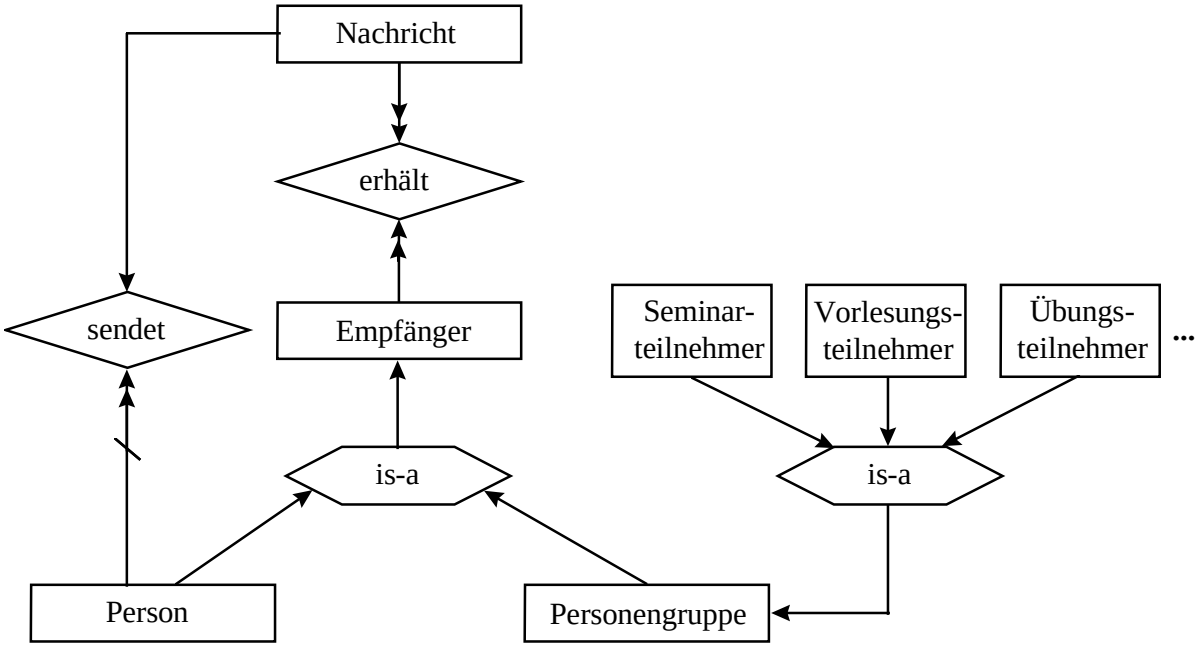


Abbildung 17: ERD Teilschema Benutzerkommunikation

Zusammenfassung der Entitäten des Teilschemas Benutzerkommunikation:

Name:	**Attribute:**
Empfänger	Mail-Adresse
Nachricht	Betreff, Inhalt, Datum
Person	siehe Teilschema *Personen*
Personengruppe	-
Seminarteilnehmer	siehe Teilschema *Seminar*
Übungsteilnehmer	siehe Teilschema *Vorlesung*
Vorlesungsteilnehmer	siehe Teilschema *Vorlesung*

Abbildung 18: Entitäten des Teilschemas Benutzerkommunikation

Zusammenfassung der Relationen des Teilschemas Benutzerkommunikation:

Name:	**Attribute:**	**Beziehungstyp:**	**beteiligte Entitäten:**
erhält	Status	m:n	Nachricht, Empfänger
sendet	-	1:n (partiell)	Nachricht, Person

Abbildung 19: Relationen des Teilschemas Benutzerkommunikation

5.1.4 Das Teilschema Vorlesungen

Der größte Teil der angebotenen Veranstaltungen an einer *Universität* sind die ***Vorlesungen***. Sie sind gekennzeichnet durch

- die *Laufende Nummer*, die zur Anmeldung beim Prüfungsamt benötigt wird,
- die *Prüfungsart*, die den Prüfungsmodus bestimmt, d. h. ob eine Prüfung schriftlich oder mündlich stattfindet,
- den *Umfang*, der beschreibt wieviele Stunden die *Vorlesung* und die evtl. dazugehörige *Übung* umfassen (z. B. 4V+2Ü für vier Vorlesungsstunden und zwei Stunden Übung),
- den *Titel*, der der Vorlesung einen Namen gibt,
- die *Kurzbeschreibung*, die einige Stichworte über den Inhalt der *Vorlesung* enthält und
- die *Voraussetzungen*, die notwendig sind, um Inhalt der *Vorlesung* zu verstehen.

Die ***Vorlesungen***, die während eines ***Semesters*** *angeboten* werden, können von einem oder mehreren ***Dozenten*** gehalten werden. Bei den ***Dozenten*** handelt es sich um ***Mitarbeiter***, die die Eigenschaft *lehrberechtigt* besitzen.

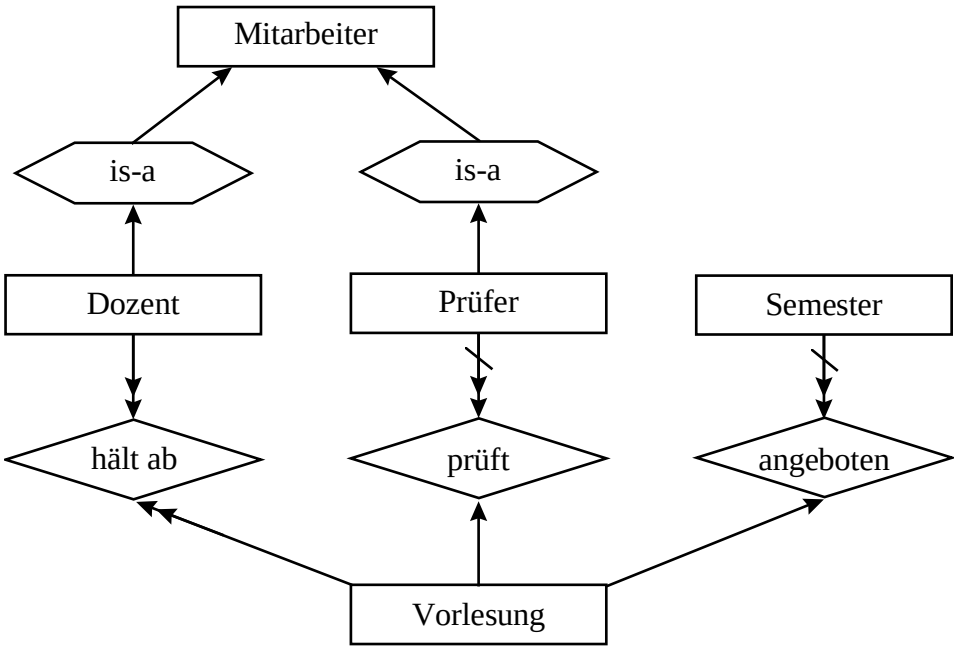


Abbildung 20: ERD Vorlesungen01

Am Ende der ***Vorlesung*** kann ein ***Prüfer*** die Leistungen der ***Vorlesungsteilnehmer*** *prüfen*, sofern sie sich zur Prüfung angemeldet haben. Ein ***Prüfer*** ist ein ***Mitarbeiter*** mit der Eigenschaft *prüfberechtigt*.

Studenten, die eine ***Vorlesung*** besuchen werden ***Vorlesungsteilnehmer*** genannt. Durch die Relation ***nimmt teil*** wird die Beziehung zwischen den Entitäten ***Vorlesung*** und ***Vorlesungsteilnehmer*** modelliert. Die Relation ***nimmt teil*** enthält das Attribut *Prüfung angemeldet*, aus der ersichtlich ist, ob sich der ***Vorlesungsteilnehmer*** zur Prüfung angemeldet hat und sich am Ende der Vorlesung prüfen lassen möchte. Mit Hilfe des Attributes *Note* kann die erbrachte Leistung eines geprüften ***Vorlesungsteilnehmer*** verwaltet werden.

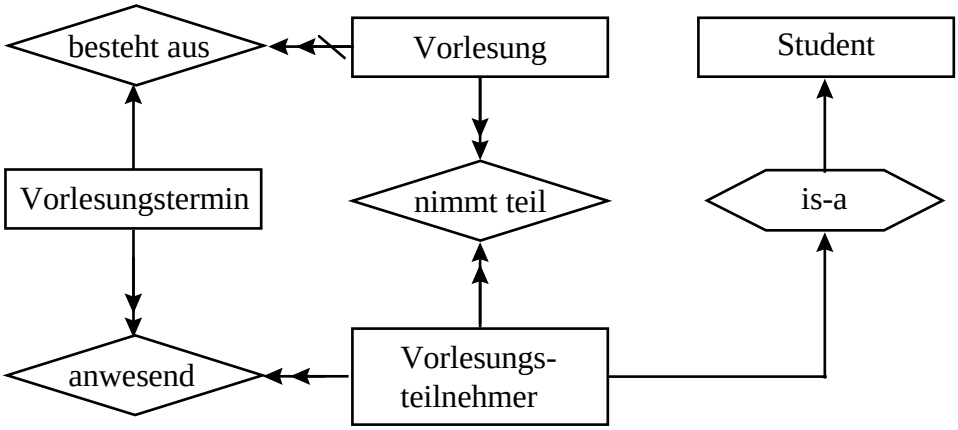


Abbildung 21: ERD Vorlesungen02

Eine ***Vorlesung besteht aus*** einem ***Vorlesungstermin***, an dem die ***Vorlesung*** stattfindet. Die Entität ***Vorlesungstermin*** verfügt über die Attribute *Ort*, *Datum*, *Beginn*, *Ende* und *Themenübersicht*. Der Ort legt fest, wo die ***Vorlesung*** stattfindet und durch die Attribute *Datum*, *Beginn* und *Ende* wird eine zeitliche Beziehung zugeordnet. Mit Hilfe der *Themenübersicht* kann man einen Kurzüberblick über den Inhalt zu einer ***Vorlesung*** an einem bestimmten ***Vorlesungstermin*** erhalten.
In einigen Studiengängen gibt es Vorlesungen, in denen eine Anwesenheitspflicht der ***Vorlesungsteilnehmer*** besteht. Es wird daher im Entwurf durch die Relation ***anwesend*** die Möglichkeit vorgesehen, die Anwesenheit eines ***Vorlesungsteilnehmer*** zu einem bestimmten ***Vorlesungstermin*** zu verwalten.

Besitzt eine ***Vorlesung Übungen***, so können die ***Übungsteilnehmer*** auf verschiedene ***Übungsgruppen*** aufgeteilt werden, wenn die Anzahl der ***Übungsteilnehmer*** dies erfordert.

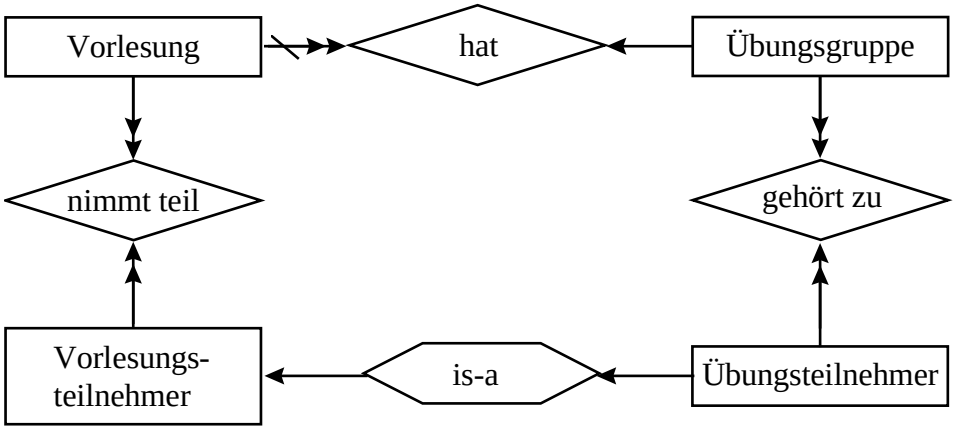


Abbildung 22: ERD Vorlesungen03

Durch das Attribut *Übungsgruppen-Nummer* werden die einzelnen ***Übungsgruppen*** zur Identifikation numeriert.

Außer den ***Dozenten***, die die ***Vorlesungen*** halten, können auch zusätzliche ***Übungsleiter*** die ***Übungen beaufsichtigen***. Auch bei den ***Übungsleiter*** handelt es sich um eine Person der Klassifikation ***Mitarbeiter***.
Die einzelnen ***Übungsgruppen besprechen*** ihre ***Aufgaben*** zu festgelegten ***Übungsterminen***, in denen die ***Lösungen*** zu den einzelnen ***Aufgaben*** erarbeitet werden.

Ein ***Übungstermin*** beinhaltet folgenden Attribute:
- *Ort*:
- *Datum*:
- *Beginn* und *Ende*:
- *Übungsnummer*:

Eine ***Aufgabe*** wird durch die Attribute *Frage* und einer evtl. zugehörigen *Grafik* beschrieben und die dazugehörige ***Lösung*** durch die Attribute *Antwort* und ebenfalls einer evtl. zugehörigen *Grafik*.

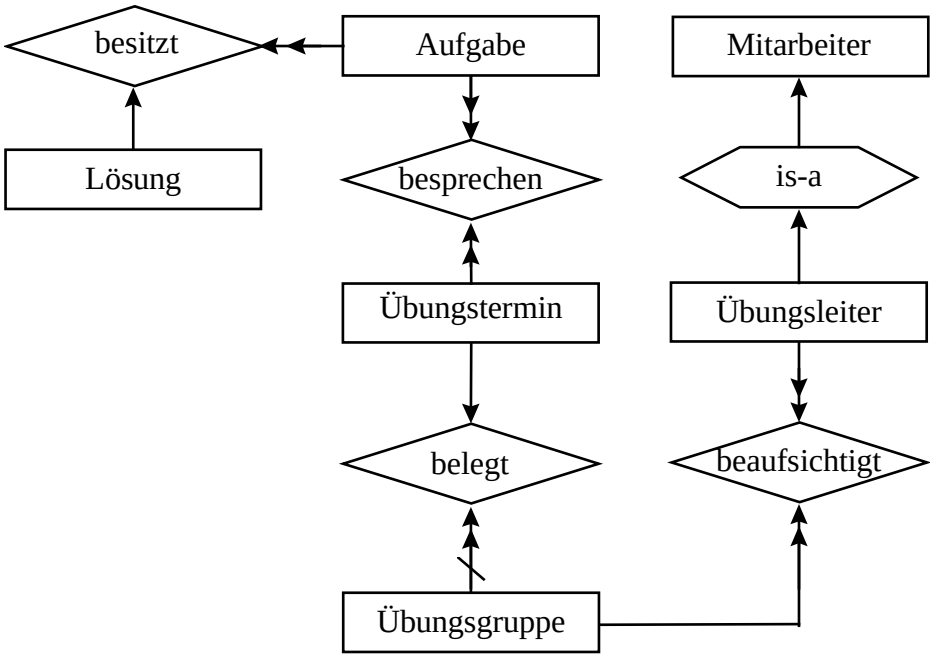


Abbildung 23: ERD Vorlesungen04

Zusammenfassung des Teilschemas Vorlesungen:

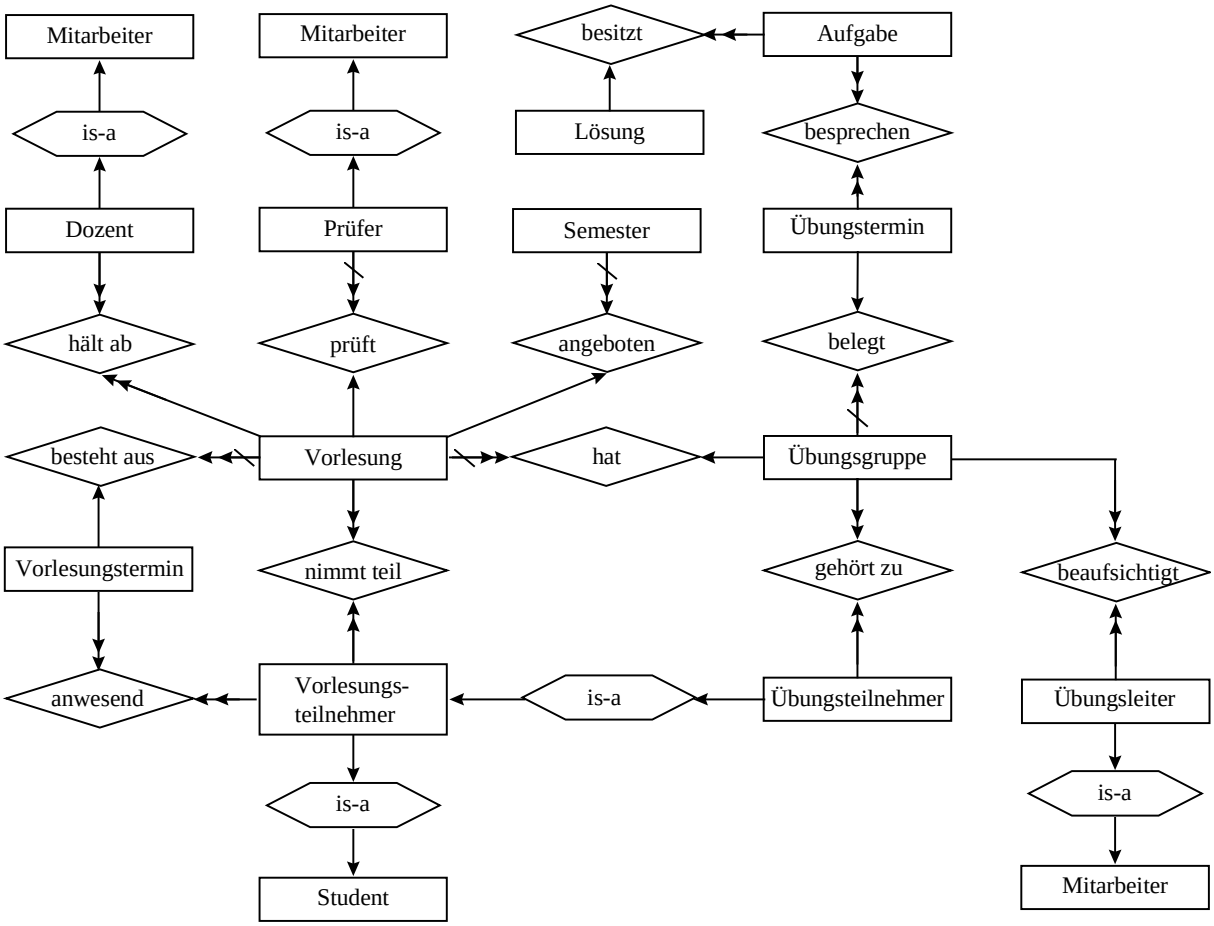


Abbildung 24: ERD Teilschema Vorlesungen

Zusammenfassung der Entitäten des Teilschemas Vorlesungen:

Name:	**Attribute:**
Aufgabe	Frage, Grafik
Dozent	-
Lösung	Antwort, Grafik
Mitarbeiter	siehe Teilschema *Personen*
Prüfer	-
Semester	siehe Teilschema *Personen*
Student	siehe Teilschema *Personen*
Übungsgruppe	Übungsgruppen-Nummer
Übungsleiter	-
Übungstermin	Ort, Datum, Beginn, Ende, Übungsnummer
Übungsteilnehmer	-
Vorlesung	Laufende Nummer, Prüfungsart, Umfang, Titel, Kurzbeschreibung, Voraussetzung,
Vorlesungstermin	Ort, Datum, Beginn, Ende, Themenübersicht
Vorlesungsteilnehmer	-

Abbildung 25: Entitäten des Teilschemas Vorlesungen

Zusammenfassung der Relationen des Teilschemas Vorlesungen:

Name:	**Attribute:**	**Beziehungstyp:**	**beteiligte Entitäten:**
angeboten	-	1:n (partiell)	Vorlesung, Semester
anwesend	-	m:n	Vorlesungstermin, Vorlesungsteilnehmer
beaufsichtigt	-	m:n	Übungsgruppe, Übungsleiter
belegt	-	1:n (partiell)	Mitarbeiter, Abteilung
besitzt	-	1:n	Aufgabe, Lösung
besprechen	-	m:n	Übungstermin, Aufgabe
besteht aus	-	1:n (partiell)	Vorlesung, Vorlesungstermin
gehört zu	-	m:n	Übungsgruppe, Übungsteilnehmer
hält ab	-	m:n	Dozent, Vorlesung
hat	-	1:n (partiell)	Vorlesung, Übungsgruppe
nimmt teil	Note, Prüfung angemeldet	m:n	Vorlesung, Vorlesungsteilnehmer
prüft	-	m:n	Prüfer, Vorlesung

Abbildung 26: Relationen des Teilschemas Vorlesungen

5.1.5 Das Teilschema Seminare

Seminare sind eine weitere Form von Veranstaltungen einer universitären Umgebung.

Ein ***Seminar*** enthält folgende Attribute:

- *Typ*: Mit diesem Attribut wird festgelegt, ob es sich um ein Seminar oder ein Hauptseminar handelt.
- *Laufende Nummer*: Das Attribut *Laufende Nummer* dient bei der Anmeldung beim Prüfungsamt zur eindeutigen Identifikation.
- *Voraussetzungen*: Mit Hilfe dieses Attributes können die Anforderungen beschrieben werden, über die der Teilnehmer verfügen sollte.
- *Titel*: Dieses Attribut enthält den Namen des Seminars.
- *Kurzbeschreibung*: In der *Kurzbeschreibung* werden Informationen über das Themengebiet abgelegt.
- *Umfang*:

Wie in der Anforderungsanalyse festgestellt, wird in einem ***Seminar*** ein bestimmtes Sachgebiet behandelt. Dieses Sachgebiet ***besteht aus*** ein oder mehrere Themengruppen, die wiederum mehrere Themen enthalten können. Die Beziehung zwischen den Entitäten ***Themengruppe*** und ***Thema*** wird durch die Relation ***enthält*** dargestellt und durch die Attribute *Name* und *Kurzbeschreibung* können sie benannt und beschrieben werden.

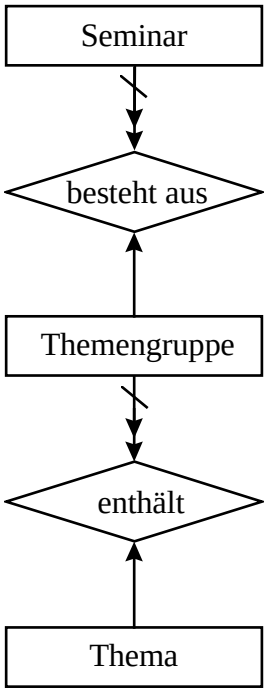


Abbildung 27: ERD Seminare01

Ein ***Seminar*** kann von einem oder mehreren ***Betreuer betreut*** werden. Damit ein ***Seminars*** bewertet werden kann, ist ein ***Prüfer*** notwendig, der die erbrachte Leistung benotet. Dies wird mittels der Relation ***prüft*** modelliert.
Betreuer und *Prüfer* sind spezialisierte Entitäten der Entität *Mitarbeiter*, wobei ein *Prüfer* das Merkmal *prüfberechtigt* besitzen muß
Studenten, die an einem ***Seminar*** teilnehmen werden als ***Seminarteilnehmer*** bezeichnet. Die Zuordnung zwischen den Entitäten ***Seminar*** und ***Seminarteilnehmer*** erfolgt durch die Relation ***nimmt teil***.

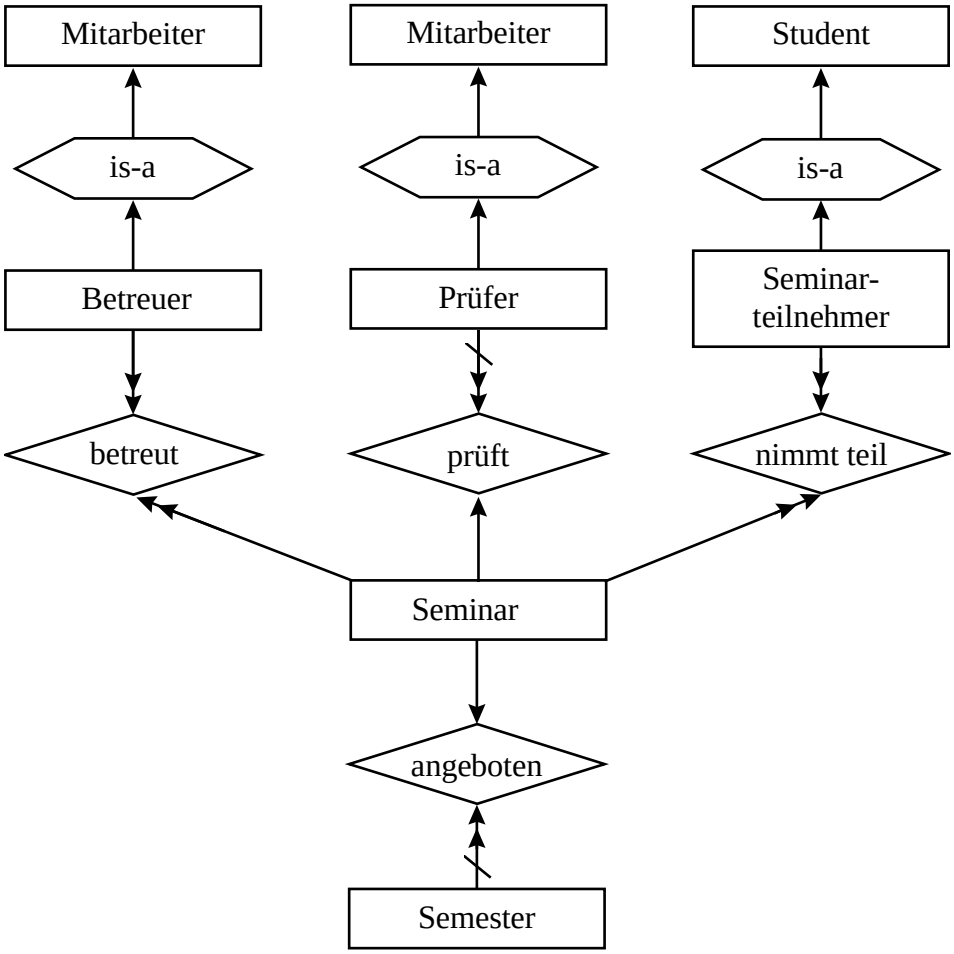


Abbildung 28: ERD Seminare02

Durch die Entität ***Semester*** und die Relation ***angeboten*** wird dem Seminar eine Zeitkomponente hinzugefügt, durch die festgestellt werden kann, in welchem ***Semester*** das ***Seminar*** stattfindet.

Ein ***Seminar*** setzt sich aus Seminarterminen zusammen. Diese Beziehung wird durch die Relation ***hat*** erzeugt. Ein ***Seminartermin*** wird durch die Attribute *Ort*, *Datum*, *Ende* und den *Grund* des Treffens beschrieben. Mögliche Gründe können die „Vergabe der Themen“ oder der „Referat des Themas XYZ“ sein.
Wenn ein ***Thema*** von einem ***Seminarteilnehmer*** gewählt wurde, dann wird das ***Thema*** zu einem ***Referat***. An einem ***Seminartermin*** können mehrere Referate stattfinden. Jeder ***Seminarteilnehmer*** kann einen ***Thema*** bearbeiten. Dazu muß er *Folien* für die Vorstellung des Referats und eine schriftliche *Ausarbeitung* erarbeiten. Diese beiden Ressourcen können nach dem ***Seminartermin*** ins System integriert werden, um sie für andere Personen zugänglich zu machen. Zusätzlich besitzt die Entität ***Referat*** das Attribut *Note* mit dessen Hilfe die Bewertung für einen Referat eines Seminarteilnehmers verwaltet werden kann.

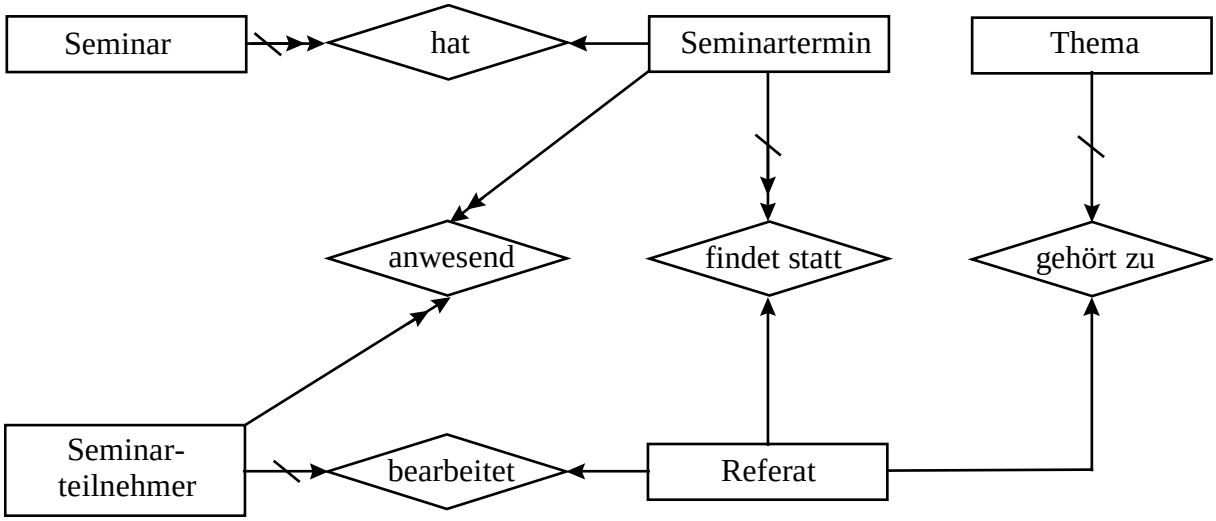


Abbildung 29: ERD Seminare03

In Seminaren besteht meistens Anwesenheitspflicht. Mittels der Relation ***anwesend*** kann der Betreuer eine Anwesenheitsliste führen.

Zusammenfassung des Teilschemas Seminare:

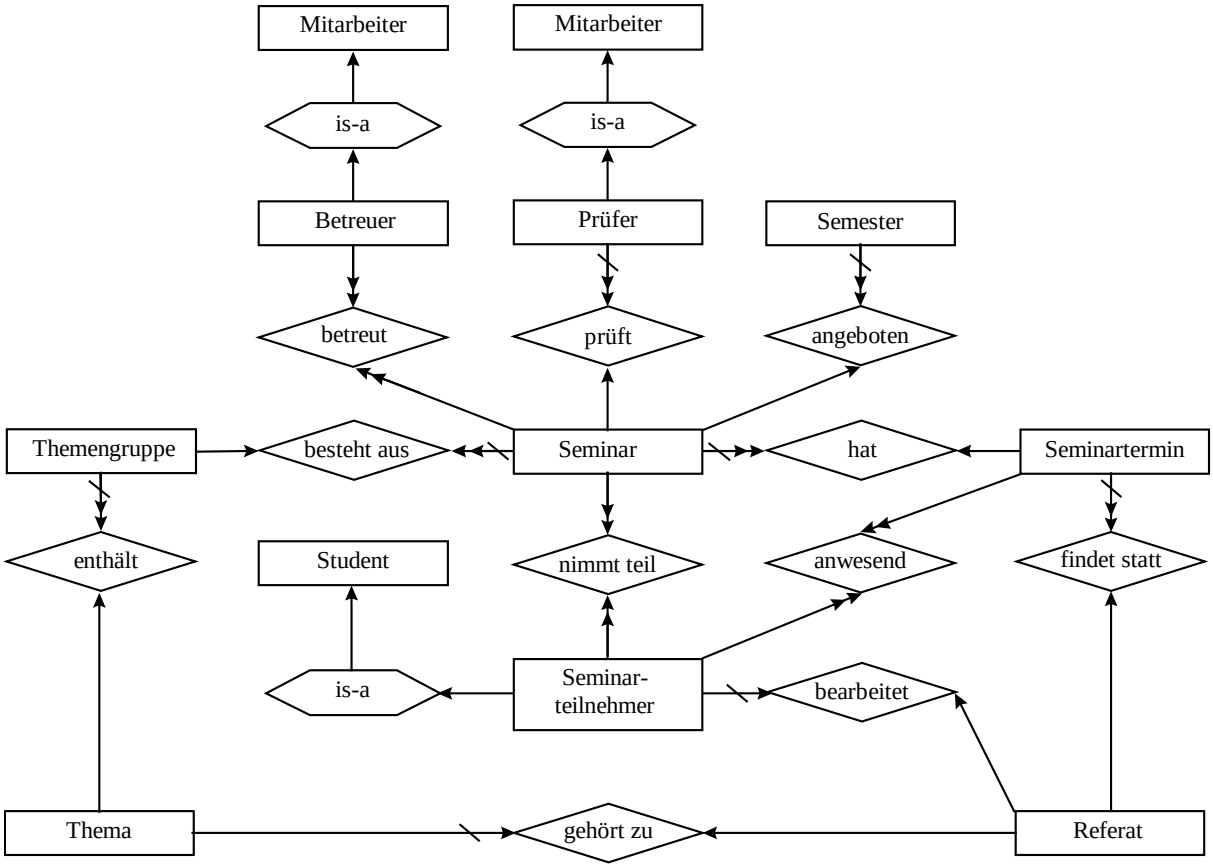


Abbildung 30: ERD des Teilschemas Seminare

Zusammenfassung der Entitäten des Teilschemas Seminare:

Name:	**Attribute:**
Betreuer	-
Mitarbeiter	siehe Teilschema *Personen*
Prüfer	-
Semester	siehe Teilschema *Personen*
Seminar	Typ, Laufende Nummer, Voraussetzung, Titel, Kurzbeschreibung, Umfang
Seminarteilnehmer	-
Seminartermin	Ort, Datum, Beginn, Ende, Grund
Student	siehe Teilschema *Personen*
Thema	Titel, Kurzbeschreibung
Themengruppe	Titel, Kurzbeschreibung
Referat	Note, Folien, Ausarbeitung

Abbildung 31: Entitäten des Teilschemas Seminare

Zusammenfassung der Relationen des Teilschemas Seminare:

Name:	**Attribute:**	**Beziehungstyp:**	**beteiligte Entitäten:**
angeboten	-	1:n (partiell)	Seminar, Semester
anwesend	-	m:n	Seminartermin, Seminarteilnehmer
bearbeitet	-	1:1 (partiell)	Seminarteilnehmer, Referat
besteht aus	-	1:n (partiell)	Seminar, Themengruppe
betreut	-	m:n	Betreuer, Seminar
enthält	-	1:n (partiell)	Themengruppe, Thema
gehört zu	-	1:1 (partiell)	Thema, Referat
findet statt	-	1:n (partiell)	Seminartermin, Referat
hat	-	1:n (partiell)	Seminar, Seminartermin
nimmt teil	-	m:n	Seminar, Seminarteilnehmer
prüft	-	1:n (partiell)	Prüfer, Seminar

Abbildung 32: Relationen des Teilschemas Seminare

5.1.6 Das Teilschema Vorträge

Eine anderer Veranstaltungstyp sind Vorträge zu einem bestimmten Themengebiet. Ein ***Vortrag*** wird beschreiben durch die Attribute *Titel* und *Kurzbeschreibung*. Durch die Relation ***hält*** wird zu einem ***Vortrag*** ein ***Referent*** zugeordnet, der den Vortrag präsentiert.

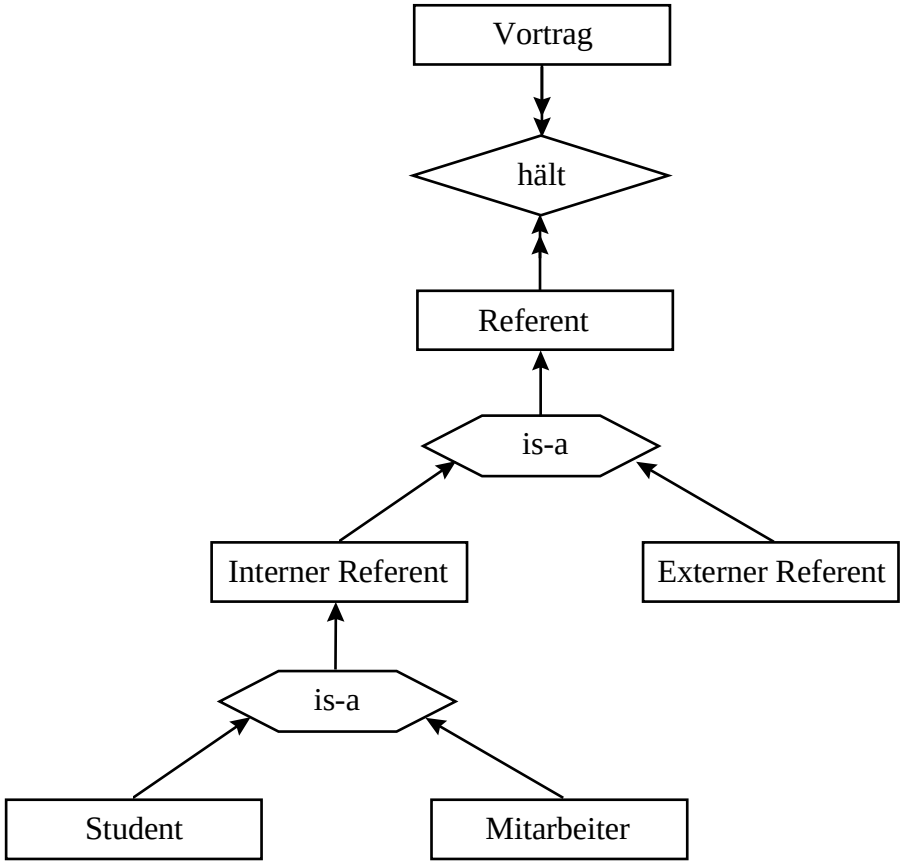


Abbildung 33: ERD Vorträge Ansatz 1

Durch die Attribute *Ort* und *Zeit*, die Bestandteil der Relation ***hält*** sind, wird dem ***Vortrag*** eine zeitliche und örtliche Komponente hinzugefügt.

Wie in der Anforderungsanalyse festgestellt, muß zwischen den Referenten unterschieden werden:

- ***Interner Referent***: Dabei handelt es sich um Personen der Universität, wie beispielsweise Studenten oder Mitarbeiter. Diese Personen werden in der Datenbank verwaltet.
- ***Externer Referent***: Dieser Personentyp hat keine „direkte“ Beziehung zur Universität, d. h. es sind keine Personen, die unmittelbar für den Universitätsbetrieb notwendig sind. Es sind Personen, die einmalig einen Vortrag an der Universität zu einem bestimmten Sachgebiet halten. Ein externer Referent ist demnach keine Person im Sinne von Kapitel 5.2. Der Vortrag eines externen Referenten ist nicht Bestandteil einer Vorlesung oder einer anderen universitären Veranstaltung und somit auch nicht prüfungsrelevant.

Diese Erkenntnisse führen zu einem neuen Ansatz:

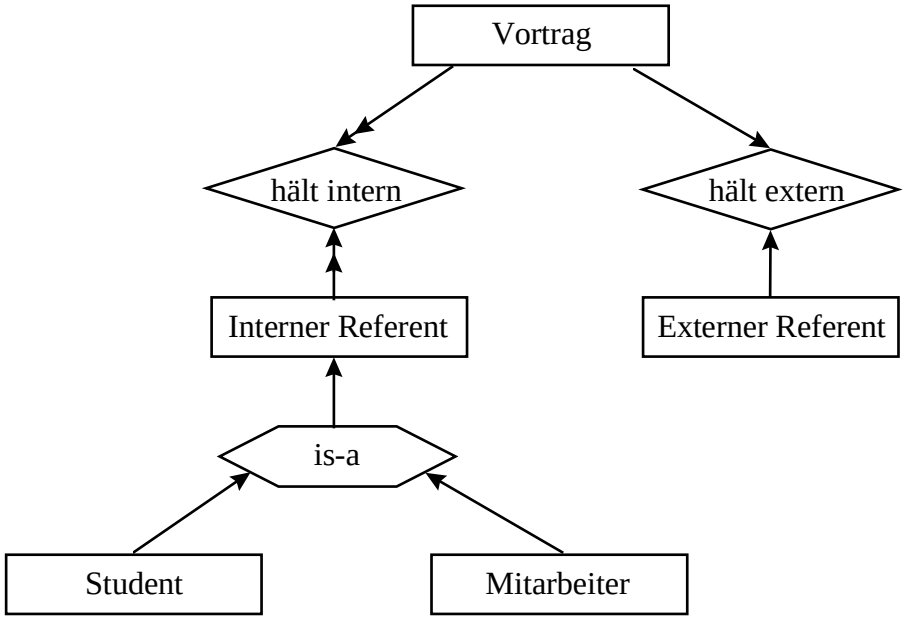


Abbildung 34: ERD Vorträge Ansatz 2

Da die Beziehungen der internen und externen Referenten zu einem Vortrag unterschiedlicher Semantik ist, sind zwei verschiedene Relationen ***hält intern*** und ***hält extern*** notwendig, die diese Beziehung abbilden.

Zusammenfassung der Entitäten des Teilschemas Vorträge:

Name:	**Attribute:**
Vortrag	Titel, Kurzbeschreibung
Interner Referent	-
Externer Referent	Name, Herkunft
Student	siehe Teilschema Personen
Mitarbeiter	siehe Teilschema Personen

Abbildung 35: Entitäten des Teilschemas Vorträge

Zusammenfassung der Relationen des Teilschemas Vorträge:

Name:	**Attribute:**	**Beziehungstyp:**	**beteiligte Entitäten:**
hält intern	Ort, Datum, Zeit	m:n	Vortrag, Interner Referent
hält extern	Ort, Datum, Zeit	1:1	Vortrag, Externer Referent

Abbildung 36: Relationen des Teilschemas Vorträge

5.1.7 Das Teilschema Benutzerrechte

Die Rechte der Personen können in diesem Kommunikations- und Präsentationssystem genauso modifiziert werden, wie die Inhalte einer Vorlesung, d. h. die Vergabe der Rechte werden in dieses System integriert. Ein Recht könnte z. B. „Anlegen einer Vorlesung“, „Löschen eines Studenten“ oder auch die Autorisierung (Vergabe von Benutzerrechten) selbst sein.
Zu diesem Zweck wird eine Entität *Funktion* kreiert, die ein solches Recht abbildet. Diese Entität enthält als Attribute den Namen der Funktion, sowie eine Beschreibung, die zur textuellen Erklärung der Funktion dient.

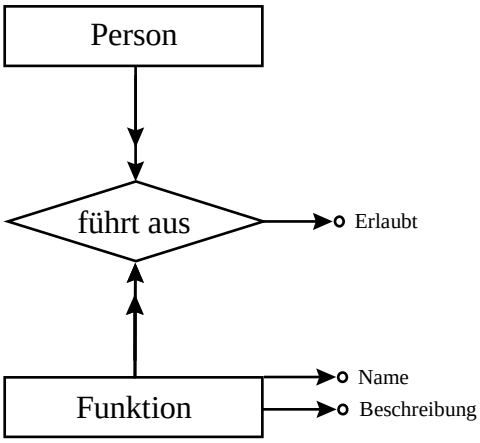


Abbildung 37: ERD des Teilschemas Benutzerrechte

Durch die Relation *führt aus* wird jede *Funktion* auf eine *Person* abgebildet und durch das Attribut *Erlaubt* kann einer *Person* ein Recht zur Ausführung einer *Funktion* erteilt oder entzogen werden.

Zusammenfassung der Entitäten des Teilschemas Benutzerrechte:

Name:	**Attribute:**
Funktion	Name, Beschreibung
Person	siehe Teilschema *Personen*

Abbildung 38: Entitäten des Teilschemas Benutzerrechte

Zusammenfassung der Relationen des Teilschemas Benutzerrechte:

Name:	**Attribute:**	**Beziehungstyp:**	**beteiligte Entitäten:**
führt aus	Erlaubt	m:n	Funktion, Person

Abbildung 39: Relationen des Teilschemas Benutzerrechte

5.2 Datenbankschema

Der im vorhergehenden Kapitel erzeugte konzeptionelle Entwurf wird nun in ein logisches Datenbankschema abgebildet. Dabei werden die Tabellen bis zur dritten Normalform transformiert. !!Schwierigkeiten!!, die sich daraus ergaben, werden bei der Beschreibung der Teilschemen diskutiert.

Für die Tabellenangaben wird folgende Notation verwendet:
- Die Tabellennamen werden um das Präfix „tbl“ ergänzt, damit sie später innerhalb der SQL-Statements von Views unterschieden werden können.
- Schlüsselattribute einer Tabelle sind unterstrichen.

Teilschema Universitätsstruktur

Die im konzeptionellen Entwurf entstandenen ER-Diagramme lassen drei Möglichkeiten für den logischen Entwurf des Datenbankschemas zu:

- Eine eigene Tabelle für die Entität **Sekretariat** mit einem Fremdschlüsselattribut für die Institution der hierarchischen Universitätsstruktur. Da beim Entwurf der Datenbanktabellen jedoch künstliche Schlüsselattribute verwendet werden, ist eine eindeutige Zuordnung zwischen Sekretariat und der jeweiligen Institution nicht mehr rekonstruierbar und deshalb scheidet diese Möglichkeit aus.
- Eine eigene Tabelle für die Entität **Sekretariat** jeder Institution der hierarchischen Universitätsstruktur. Diese Möglichkeit wäre prinzipiell möglich, aber das Datenbankschema würde um vier weitere Entitäten zzgl. der Relationen anwachsen.
- Die Attribute der Entität **Sekretariat** werden in die Entitäten der jeweiligen Institution hinzugenommen. Die Entscheidung fiel für diese Möglichkeit, da sich dadurch die Komplexität des Datenbankschemas im Rahmen dieser Arbeit hält.

Aus den Entitäten:

Name:	**Datenbanktabelle:**
Universität	tblUniversitaet(UniversitaetID, Name, Strasse, Plz, Ort, SekretariatPersonal, SekretariatZimmer, SekretariatTelefon, SekretariatSprechstunde)
Fakultät	tblFakultaet(FakultaetID, Name, Strasse, Plz, Ort, Dekan, ProDekan, StudienDekan, SekretariatPersonal, SekretariatZimmer, SekretariatTelefon, SekretariatSprechstunde, UniversitaetID)
Institut	tblInstitut(InstitutID, Name, Strasse, Plz, Ort, Direktor, StellvDirektor, SekretariatPersonal, SekretariatZimmer, SekretariatTelefon, SekretariatSprechstunde, PruefAusschuss, StudienBeratung, FakultaetID)
Abteilung	tblAbteilung(AbteilungID, Name, Strasse, Plz, Ort, Leitung, SekretariatPersonal, SekretariatZimmer, SekretariatTelefon, SekretariatSprechstunde, InstitutID)

Abbildung 40: Datenbanktabellen aus den Entitäten des Teilschemas Universitätsstruktur

Aus den Relationen:

Name:	beteiligte Entitäten:	Datenbanktabelle:
besteht aus	Universität, Fakultät	in der Tabelle *tblFakultaet* integriert
besteht aus	Fakultät, Institut	in der Tabelle *tblInstitut* integriert
besteht aus	Institut, Abteilung	in der Tabelle *tblAbteilung* integriert

Abbildung 41: Datenbanktabellen aus den Relationen des Teilschemas Universitätsstruktur

Teilschema Personen

Beim logischen Entwurf der Datenbankschemas für das Teilschema *Personen* gab es drei Möglichkeiten:

1. Alle Attribute der Entität ***Person*** werden an die spezialisierten Entitäten ***Student***, ***Mitarbeiter*** und ***Personal*** weitergereicht, d. h. es werden drei Tabellen für die spezialisierten Entitäten entworfen, die alle Attribute *Anrede*, *Vorname*, *Nachname*, *MailAdresse*, *Strasse*, *Plz*, *Ort*, *PrivatTelefon*, *Status*, *GeburtsDatum*, *Login* und *Passwort* enthalten. Dies hat den Nachteil, daß die Eindeutigkeit der Attributnamen verlorengeht.
2. Alle Attribute der spezialisierten Entitäten werden in die allgemeine Entität ***Person*** übernommen. Dabei entsteht nur eine Tabelle für alle Entitäten, die jedoch das Kriterium der dritten Normalform verletzt.
3. Eine Tabelle für die Entität ***Person***, ***Student***, ***Mitarbeiter*** und ***Personal***, wobei zwischen ***Person*** und den spezialisierten Entitäten ***Student***, ***Mitarbeiter*** und ***Personal*** 1:1-Beziehungen bestehen.
   Es wurde dieses Verfahren bevorzugt, weil
   - dieser Entwurf dem objektorientierten Ansatz weitgehendst entspricht,
   - auf diese Weise ein minimales Datenbankschema realisiert werden kann und
   - nach Booch [Boo94] durch diese vertikale Unterteilung eine Effizienzsteigerung möglich ist.

Aus den Entitäten:

Name:	Datenbanktabelle:
Abteilung	siehe Teilschema *Universitätsstruktur*
Mitarbeiter	tblMitarbeiter(MitarbeiterID, Titel, Telefon, Sprechstunde, Typ, Zimmer, LehrBerechtigt, PruefBerechtigt, AbteilungID)
Person	tblPerson(PersonID, Anrede, Vorname, Nachname, MailAdresse, Strasse, Plz, Ort, PrivatTelefon, Status, GeburtsDatum, Login, Passwort)
Personal	tblPersonal(PersonalID)
Semester	tblSemester(SemesterID, Semester)
Student	tblStudent(StudentID, MatrNr, HatVordiplom, StudienBeginn, UniversitaetID)
Studienfach	tblStudienfach(StudentID, StudienfachName)
Universität	siehe Teilschema *Universitätsstruktur*

Abbildung 42: Datenbanktabellen aus den Entitäten des Teilschemas Personen

Aus den Relationen:

Name:	**beteiligte Entitäten:**	**Datenbanktabelle:**
arbeitet für	Mitarbeiter, Abteilung	in der Tabelle *tblMitarbeiter* integriert
begonnen	Student, Semester	in der Tabelle *tblStudent* integriert
belegt	Student, Studienfach	in der Tabelle *tblStudienfach* integriert
studiert an	Student, Universität	in der Tabelle *tblStudent* integriert

Abbildung 43: Datenbanktabellen aus den Relationen des Teilschemas Personen

Teilschema Benutzerkommunikation

Beim Entwurf der Datenbankschemas für das Teilschema Benutzerkommunikation traten keine Schwierigkeiten auf.
Es wird darauf hingewiesen, daß Personengruppen (z. B. Seminarteilnehmer, Übungsteilnehmer oder Vorlesungsteilnehmer) kein eigenes Attribut MailAdresse erhalten. Die Funktionalität, daß eine Nachricht an die Teilnehmer einer Personengruppe gesendet werden kann, wird durch Mechanismen der Anwendung gelöst.

Aus den Entitäten:

Name:	**Datenbanktabelle:**
Empfänger	in der Tabelle *tblPerson* integriert
Nachricht	tblNachricht(NachrichtID, Betreff, Inhalt, Datum, Sender)
Person	siehe Teilschema *Personen*
Personengruppe	in den Tabellen *tblUebungsGruppe*, *tblNimmtAnVTeil*, *tblNimmtAnSTeil* integriert
Seminarteilnehmer	siehe Teilschema *Seminar*
Übungsteilnehmer	siehe Teilschema *Vorlesung*
Vorlesungsteilnehmer	siehe Teilschema *Vorlesung*

Abbildung 44: Datenbanktabellen aus den Entitäten des Teilschemas Benutzerkommunikation

Aus den Relationen:

Name:	**beteiligte Entitäten:**	**Datenbanktabelle:**
erhält	Nachricht, Empfänger	tblErhaelt(PersonID, NachrichtID, Status)
sendet	Nachricht, Person	in der Tabelle *tblNachricht* integriert
zugeordnet zu	Person, Personengruppe	in den Tabellen *tblNimmtAnVTeil*, *tblNimmtAnSTeil*, *tblGehoertZu* integriert

Abbildung 45: Datenbanktabellen aus den Relationen des Teilschemas Benutzerkommunikation

Teilschema Vorlesungen

Aus den Entitäten:

Name:	**Attribute:**
Aufgabe	tblAufgabe(AufgabeID, Frage, Grafik)
Dozent	in der Tabelle *tblHaeltAb* integriert
Lösung	tblLoesung(LoesungID, AufgabeID, Antwort, Grafik)
Mitarbeiter	siehe Teilschema *Personen*
Prüfer	in der Tabelle *tblVorlesung* integriert
Semester	siehe Teilschema *Personen*
Student	siehe Teilschema *Personen*
Übungsgruppe	tblUebungsGruppe(UebungsGruppeID, UebungsGruppenNr, VorlesungID)
Übungsleiter	in der Tabelle *tblBeaufsichtigt* integriert
Übungstermin	tblUebungsTermin(UebungsTerminID, Ort, Datum, Beginn, Ende, UebungsNr, UebungsGruppeID)
Übungsteilnehmer	in der Tabelle *tblGehoertZuUebung* integriert
Vorlesung	tblVorlesung(VorlesungID, SemesterID, LaufendeNr, PruefungsArt, Umfang, Titel, Kurzbeschreibung, Voraussetzung, Pruefer)
Vorlesungstermin	tblVorlesungsTermin(VorlesungsTerminID, Ort, Datum, Beginn, Ende, ThemenUebersicht, VorlesungID)
Vorlesungsteilnehmer	in der Tabelle *tblNimmtTeil* integriert

Abbildung 46: Datenbanktabellen aus den Entitäten des Teilschemas Vorlesungen

Aus den Relationen:

Name:	**beteiligte Entitäten:**	**Datenbanktabelle:**
angeboten	Vorlesung, Semester	in der Tabelle *tblVorlesung* integriert
anwesend	Vorlesungstermin, Vorlesungsteilnehmer	tblVAnwesend(VorlesungsTerminID, VorlesungsTeilnehmer)
beaufsichtigt	Übungsgruppe, Übungsleiter	tblBeaufsichtigt(UebungsGruppeID, UebungsLeiter)
belegt	Übungsgruppe, Übungstermin	in der Tabelle *tblUebungsTermin* integriert
besitzt	Aufgabe, Lösung	in der Tabelle *tblLoesung* integriert
besprechen	Übungstermin, Aufgabe	tblBesprechen(UebungsTerminID, AufgabeID)
besteht aus	Vorlesung, Vorlesungstermin	in der Tabelle *tblVorlesungstermin* integriert
gehört zu	Übungsgruppe, Übungsteilnehmer	tblGehoertZu(UebungsGruppeID, VorlesungsTeilnehmer)
hält ab	Dozent, Vorlesung	tblHaeltAb(VorlesungID, Dozent)
hat	Vorlesung, Übungsgruppe	in der Tabelle *tblUebungsgruppe* integriert
nimmt teil	Vorlesung, Vorlesungsteilnehmer	tblNimmtAnVTeil(VorlesungID, VorlesungsTeilnehmer, Note, PruefungAngemeldet)
prüft	Prüfer, Vorlesung	in der Tabelle *tblVorlesung* integriert

Abbildung 47: Datenbanktabellen aus den Relationen des Teilschemas Vorlesungen

Teilschema Seminare

Aus den Entitäten:

Name:	**Datenbanktabelle:**
Betreuer	in der Tabelle *tblBetreut* integriert
Mitarbeiter	siehe Teilschema *Personen*
Prüfer	in der Tabelle *tblSeminar* integriert
Semester	siehe Teilschema *Personen*
Seminar	tblSeminar(SeminarID, SemesterID, Typ, Voraussetzung, Titel, Kurzbeschreibung, Umfang, Pruefer)
Seminarteilnehmer	in der Tabelle *tblNimmtAnSTeil* integriert
Seminartermin	tblSeminarTermin(SeminarTerminID, Ort, Datum, Beginn, Ende, Grund, SeminarID)
Student	siehe Teilschema *Personen*
Thema	tblThema(ThemaID, Titel, Kurzbeschreibung, ThemenGruppeID)
Themengruppe	tblThemenGruppe(ThemenGruppeID, Titel, Kurzbeschreibung, SeminarID)
Referat	tblReferat(ReferatID, Note, Folien, Ausarbeitung, ThemaID, SeminarTeilnehmer)

Abbildung 48: Datenbanktabellen aus den Entitäten des Teilschemas Seminare

Aus den Relationen:

Name:	**beteiligte Entitäten:**	**Datenbanktabelle:**
angeboten	Seminar, Semester	in der Tabelle *tblSeminar* integriert
anwesend	Seminartermin, Seminarteilnehmer	tblSAnwesend(SeminarTerminID, SeminarTeilnehmer)
besteht aus	Seminar, Themengruppe	in der Tabelle *tblSeminar* integriert
betreut	Betreuer, Seminar	tblBetreut(Betreuer, SeminarID)
enthält	Themengruppe, Thema	in der Tabelle *tblThema* integeriert
findet statt	Seminartermin, Referat	in der Tabelle *tblReferat* integriert
hat	Seminar, Seminartermin	in der Tabelle *tblSeminarTermin* integriert
nimmt teil	Seminar, Seminarteilnehmer	tblNimmtAnSTeil(SeminarID, SeminarTeilnehmer, TeilnahmeArt)
prüft	Prüfer, Seminar	in der Tabelle *tblSeminar* integriert

Abbildung 49: Datenbanktabellen aus den Relationen des Teilschemas Seminare

Teilschema Vorträge

Aus den Entitäten:

Name:	**Datenbanktabelle:**
Vortrag	tblVortrag(VortragID, Titel, Kurzbeschreibung)
Interner Referent	in den Tabellen tblStudent und tblMitarbeiter integriert
Externer Referent	in der Tabelle tblHaeltExtern integriert
Student	siehe Teilschema Personen
Mitarbeiter	siehe Teilschema Personen

Abbildung 50: Datenbanktabellen aus den Entitäten des Teilschemas Vorträge

Aus der Relationen:

Name:	**beteiligte Entitäten:**	**Datenbanktabelle:**
hält intern	Vortrag, Interner Referent	tblHaeltIntern(PersonID, VortragID, Ort, Datum, Zeit)
hält extern	Vortrag, Externer Referent	tblHaeltExtern(VortragID, ExternRefName, ExternRefHerkunft, Ort, Datum, Zeit)

Abbildung 51: Datenbanktabellen aus der Relationen des Teilschemas Vorträge

Teilschema Benutzerrechte

Aus den Entitäten:

Name:	**Datenbanktabelle:**
Funktion	tblFunktion(FunktionID, Funktion, Beschreibung)
Person	siehe Teilschema *Personen*

Abbildung 52: Datenbanktabellen aus den Entitäten des Teilschemas Benutzerrechte

Aus den Relationen:

Name:	**beteiligte Entitäten:**	**Datenbanktabelle:**
führt aus	Funktion, Person	tblFuehrtAus(PersonID, FunktionID, Erlaubt)

Abbildung 53: Datenbanktabellen aus den Relationen des Teilschemas Benutzerrechte

5.3 Funktionalität und Dynamik

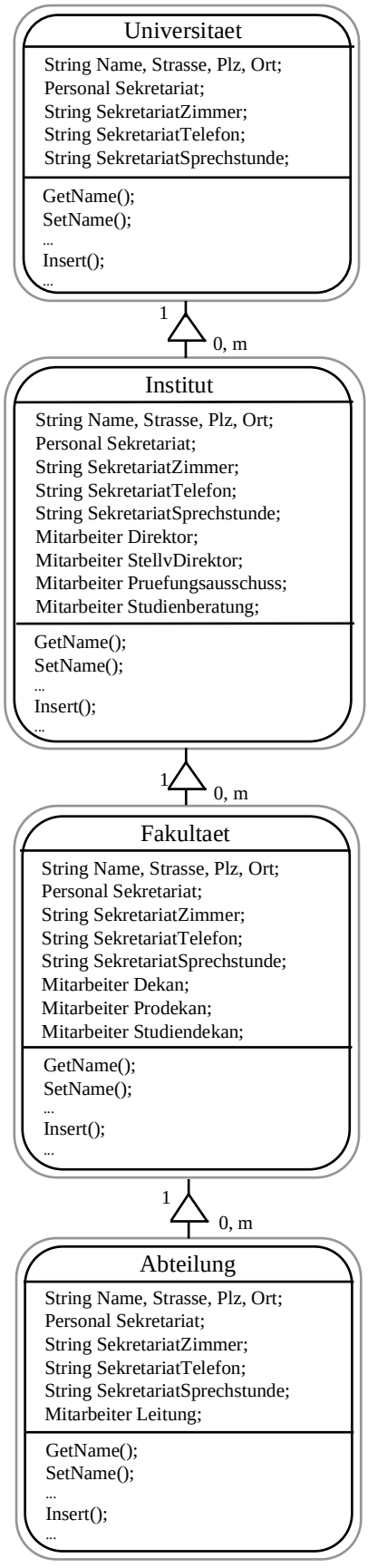
5.3.1 Einleitung

In diesem Kapitel werden die Fachkonzeptklassen mit ihren Attributen und Methoden entworfen und mit Hilfe der Notation von Coad/Yourdon [CoYo91] strukturiert dargestellt.

5.3.2 Entwurf der Fachkonzeptklassen

Basis für den Entwurf der Fachkonzeptklassen sind die Entity-Relationship-Modelle aus Kapitel 5.1 und das daraus resultierende Datenbankschema aus Kapitel 5.2.
Die Attribute der Entitäten bzw. die Felder der Tabellenschemen werden zu Attributen der Klassenstruktur. Jede Klasse erhält die Standard-Methoden, um die Attribute modifizieren und zurückgeben zu können (z.B. GetName(), SetName(String name) für die Klasse *Person*). Weiterhin erhalten die Fachkonzeptklassen drei Methoden, die die Funktionalität auf die Ereignisse Neu, Ändern und Löschen abbilden. Diese Insert-, Update-, und Delete-Methoden müssen zum einen die Oberfläche aktualisieren und zum anderen die Änderungen in der Datenbank vornehmen. Diese Änderungen werden durch die objektorientierte Zwischenschicht realisiert, die zwischen der objektorientierten Anwendung und der relationalen Datenbank steht.
Es werden exemplarisch die Fachkonzeptklassen für das Teilschema *Universitätsstruktur* und *Personen* dargestellt.

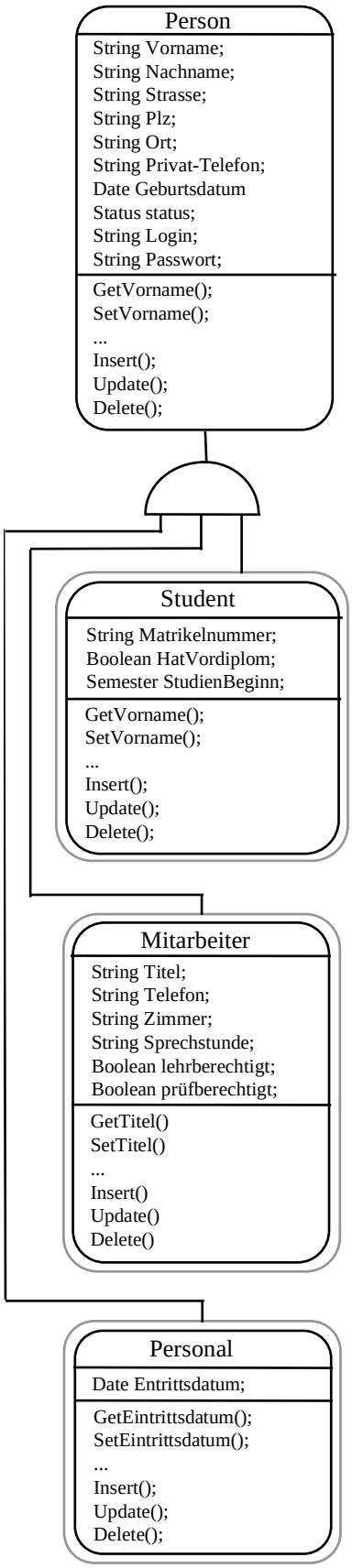
Fachkonzeptklassen für das Teilschema Universitätsstruktur




```
public class Universitaet{
    private String Name;
    private Personal Sekretariat;
    private String SekretariatZimmer;
    private String SekretariatTelefon;
    private String SekretariatSprechstunde;
    private Institut[] arrayInstitut;

    public String GetName();
    public void   SetName(String name);
    ...
    public void Insert();
}

public class Institut{
    private String Name;
    private Personal Sekretariat;
    private String SekretariatZimmer;
    private String SekretariatTelefon;
    private String SekretariatSprechstunde;
    private Mitarbeiter Direktor;
    private Mitarbeiter StellvDirektor;
    private Mitarbeiter Pruefungsausschuss;
    private Mitarbeiter Studienberatung;
    private Fakultaet[] arrayFakultaet;

    public String GetName();
    public void   SetName(String name);
    ...
    public void Insert();
}

public class Fakultaet{
    private String Name;
    private Personal Sekretariat;
    private String SekretariatZimmer;
    private String SekretariatTelefon;
    private String SekretariatSprechstunde;
    private Mitarbeiter Dekan;
    private Mitarbeiter Prodekan;
    private Mitarbeiter Studiendekan;
    private Abteilung[] arrayAbteilung;

    public String GetName();
    public void   SetName(String name);
    ...
    public void Insert();
}

public class Abteilung{
    private String Name;
    private Personal Sekretariat;
    private String SekretariatZimmer;
    private String SekretariatTelefon;
    private String SekretariatSprechstunde;
    private Mitarbeiter Leitung;

    public String GetName();
    public void   SetName(String name);
    ...
    public void Insert();
}
```

Abbildung 54: Fachkonzeptklassen für Teilschema Universitätsstrukur

Fachkonzeptklassen für das Teilschema (TS) Personen




```
abstract class Person {
    private String Vorname;
    private String Nachname;
....private String Strasse;
....private String Plz;
....private String Ort;
....private String Privat-Telefon;
....private Date Geburtsdatum;
....private Status status;
....private String Login;
....private String Passwort;

....public String GetVorname();
    public void SetVorname(String vorname);
    ...
    public void Insert();
    public void Update();
    public void Delete();
}
```

```
public class Student extends Person{
    private String Matrikelnummer;
    private Boolean HatVordiplom;
    private Semester Studienbeginn;

    public String GetMatrikelnummer();
    public void SetMatrikelnummer();
    ...
    public void Insert();
    public void Update();
    public void Delete();
}
```

```
public class Mitarbeiter extends Person{
    private String Titel;
    private String Telefon;
    private String Zimmer;
    private String Sprechstunde;
    private Boolean lehrberechtigt;
    private Boolean prüfberechtigt;

....public String GetTitel();
    public void SetTitel();
    ...
    public void Insert();
    public void Update();
    public void Delete();
}
```

```
public class Personal extends Person{
    private Date Eintrittsdatum;

....public String GetTitel();
    public void SetTitel();
    ...
    public void Insert();
    public void Update();
    public void Delete();
}
```

Abbildung 55: Fachkonzeptklassen für Teilschema Personen

5.3.3 Entwurf der objektorientierten Zwischenschicht

Die Aufgaben dieser Zwischenschicht wurde in Kapitel 4.2.1 ausführlich erläutert. Zusammenfassend handelt es sich dabei um

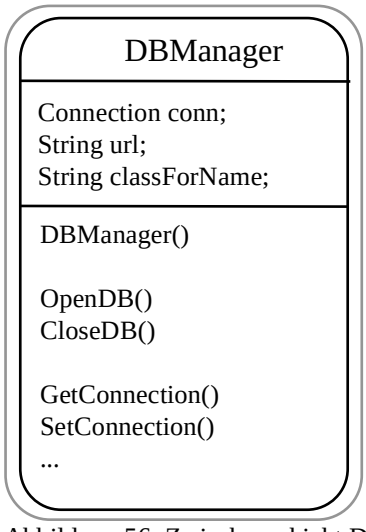
- die Abbildung zwischen Klassen und Tabellen,
- die Abbildung von Assoziationen und Aggregationen,
- die Abbildung von Vererbungen und
- die Ausführung von Abfragen mit SQL.

Eine weitere Aufgabe ist der Verbindungsaufbau und -abbau zur Datenbank. Zu diesem Zweck wird eine Klasse ***DBManager*** kreiert, die diese Aufgaben wahrnimmt. Sie enthält folgende drei Attribute:

1. *conn*: Dieses Objekt ist eine Connection-Schnittstelle von JDBC, die Methoden für die Verwaltung der Transaktionsverarbeitung, zum Erzeugen von Objekten für die Ausführung von SQL-Anweisungen und gespeicherter Prozeduren bereithält. Darüber hinaus bietet sie einige grundlegende Methoden zur Fehlerverarbeitung. Mit Hilfe dieser Schnittstelle kann eine permanente Verbindung zur Datenbank aufgebaut werden, d. h. sie bleibt solange geöffnet bis sie explizit geschlossen wird oder die Variable sich außerhalb des Gültigkeitsbereiches befindet.

2. *url*: Dieser String enthält den Namen des URL (Universal Resource Locator), mit dem die Datenbank identifiziert wird. (Bsp. jdbc:db2:wwwbase)

3. *classForName*: Um mit Hilfe der JDBC-Schnittstellen auf eine Datenbank zugreifen zu können, wird ein spezifischer JDBC-Treiber für die Datenbank benötigt. (Bsp. ibm.sql.DB2Driver)




```
public class DBManager
{
    private Connection conn;
    private String url;
    private String classForName;

    public DBManager();

    public void OpenDB();
    public void CloseDB();

    public Connection GetConnection();
    public SetConnection(Connection con);
    ...
}
```

Abbildung 56: Zwischenschicht DBManager

Für jede der logischen Einheiten *Universitätsstruktur*, *Personen*, *Veranstaltungen*, *Benutzerkommunikation* und *Benutzerrechte* wird eine Klasse entworfen, die die Schnittstelle zur Datenbank bildet.
Es wird am Beispiel für die Einheit Personen die Wirkungsweise der objektorientierten Zwischenschicht verdeutlicht. Gemäß der ermittelten Fachkonzeptklassen aus den Entitäten *Person*, *Student*, *Mitarbeiter* und *Personal* in Kapitel 5.1 Datenhaltung wird dieselbe Struktur für die Zwischenschicht entworfen:

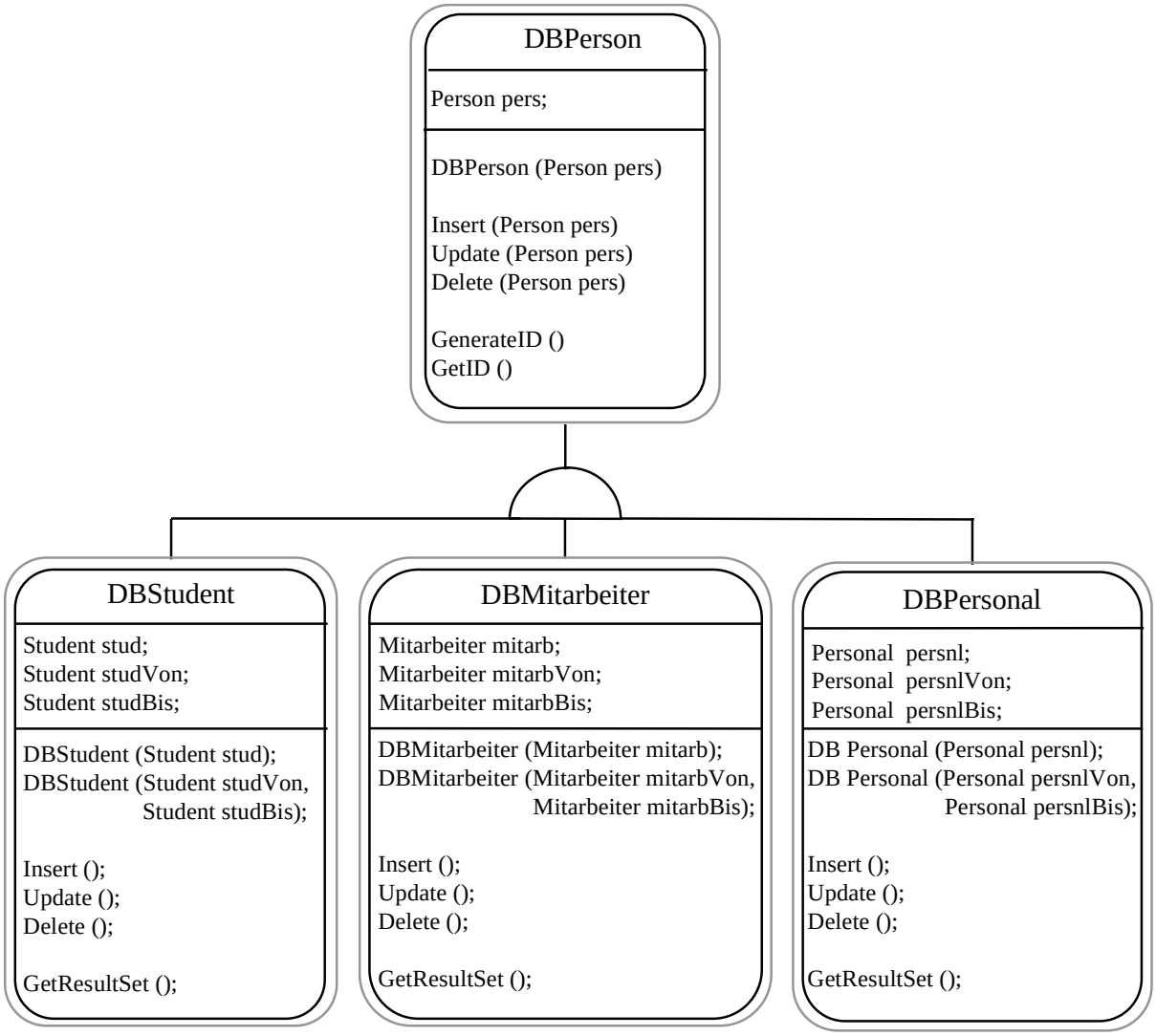


Abbildung 57: Zwischenschicht für Teilschema Personen

Nur in Klassen DBXXX werden SQL-Statements für das relationale Datenbanksystem erzeugt und ausgeführt. Das hat den Vorteil, daß bei einer eventuell späteren Anbindung an eine objektorientierte Datenbank nur diese Klasse entsprechend modifiziert werden müssen.

Anhand zweier Beispiele wird die Funktionsweise dieser objektorientierten Zwischenschicht verdeutlicht:

Funktionsweise beim Einfügen eines neuen Datensatzes am Beispiel eines Studenten:

Soll ein neuer Student mittels des Erfassungsfensters in den Datenbestand aufgenommen werden, wird beim Auslösen der Aktion Speichern ein Objekt der Klasse *Student* erzeugt. Mit diesem Objekt wird der Konstruktor *DBStudent(Student stud)* aufgerufen. Beim Aufruf der Methode *Insert()* der Klasse *DBStudent* wird das SQL-Statement entsprechend den Attributen erzeugt und ausgeführt. Zuvor erzeugt das Objekt der Oberklasse *DBPerson* einen eindeutigen Primärschlüssel, der für das Einfügen eines neuen Datensatzes notwendig ist.

Funktionsweise beim Ermitteln der selektierten Datensätze aus der Suchmaske am Beispiel der Studenten:

Werden beispielsweise folgende Angaben in der Suchmaske für die Auswahl der Studenten gemacht,

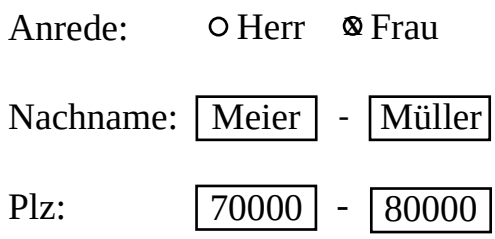


Abbildung 58: Ausschnitt aus einer Suchmaske

dann werden zwei Objekte der Fachkonzeptklasse *Student* erzeugt, wobei ein Objekt für die Attribute Anrede, Nachname und Plz die Werte „Frau“, „Meier“ und „70000“ und das zweite Objekt die Werte „Frau“, „Müller“ und „80000“ erhält. Mit diesen beiden Objekten wird der Konstruktor *DBStudent(Student studVon, Student studBis)* aufgerufen. Die *GetResultSet()*-Methode erzeugt daraufhin ein SELECT-Statement mit folgenden Angaben in der WHERE-Klausel:

```
        ...
WHERE Anrede = studVon.Anrede
AND     Nachname > studVon.Nachname AND Nachname < studBis.Nachname
AND     Plz > studVon.Plz AND Plz < studBis.Plz;
```

mit den Werten:

```
        ...
WHERE Anrede = ‘Frau’
AND     Nachname > ‘Meier’ AND Nachname < ‘Müller’
AND     Plz > ‘70000’ AND Plz < ‘80000’;
```

Mit dem Ergebnis dieser Anfrage wird das anzuzeigende Fenster initialisiert.

5.4 Benutzeroberfläche

5.4.1 Einleitung

Im Fall von Benutzeroberflächen ist die Entkopplung des Entwurf ohne Wissen über die Implementierung nicht immer sinnvoll. Ein Grund dafür ist, daß sich das Design der graphischen Benutzeroberfläche (engl. GUI Graphical User Interface) überwiegend an den Funktionen orientiert, die die verwendeten Werkzeuge anbieten.
In dieser Arbeit stand die Zielsprache schon zu Beginn fest, so daß sie beim Entwurf berücksichtigt werden konnte. Zur Verfügung stand zum einen der Java Workshop 2.0, sowie der JDK (Java Development Kit) 1.1.in der Release 6.
Der Java Workshop 2.0 wurde aus folgenden Gründen nicht für die Implementierung verwendet:

- Es können in dieses Werkzeug keine erweiterten Grafikelemente aus anderen UI-Tools eingebunden werden.
- Ein sinnvolles Arbeiten ist durch die unakzeptablen Antwortzeiten nicht möglich.
- Es werden nicht alle Schnittstellen der Version 1.1 unterstützt. So verwendet der Java Workshop 2.0 zur Ereignisbehandlung der Sprachendefinition von Java 1.0, die einen modularen Aufbau nicht zuläßt

Es kam daher der JDK 1.1 zum Einsatz. Um einen erweiterten „Pool“ von GUI-Elementen verwenden zu können, wurde der Nachfolger des AWT (Abstract Window Toolkit) das *Swing Tool Set* (auch bekannt unter dem Namen JFC Java Foundation Classes 1.1) der Version 1.02 für die Oberflächengestaltung benutzt, das in der nächsten Version 1.2 des JDK Standard sein wird.

5.4.2 Erscheinungsbild

Das Erscheinungsbild hat einen großen Einfluß auf den Erfolg auf die Akzeptanz und Zufriedenheit der Anwender. Es wird daher das Ziel angestrebt ein äußerst funktionale und intuitiv benutzbare Mensch-Maschine-Schnittstelle zu entwerfen.
Die Vorteile ein System für die gesamte Funktionalität zu entwerfen, birgt die Nachteile einer höheren Komplexität für den Benutzer und die Gefahr ihn zu überfordern. Es muß also die Möglichkeit geschaffen werden, alle Funktionen die das System bietet in einer übersichtlichen und effektiv-benutzbaren Weise darzustellen. Aus diesem Grund wird für die Anwendung eine menügesteuerte Dialogführung entworfen. Dies hat den Vorteil, daß das System bei Erweiterungen zu einem späteren Zeitpunkt um weitere Menütitel entsprechend ergänzt werden kann. Bei diesen Erweiterungen kann es sich um eine Ergänzung von weiteren Veranstaltungen oder einer zusätzlich gewünschten Funktionalität handeln.
Die logischen Einheiten *Universitätsstruktur*, *Personen*, *Veranstaltungen* und *Benutzerkommunikation* erhalten einen eigenen Menütitel.

Es wird folgende Menüstruktur entworfen:

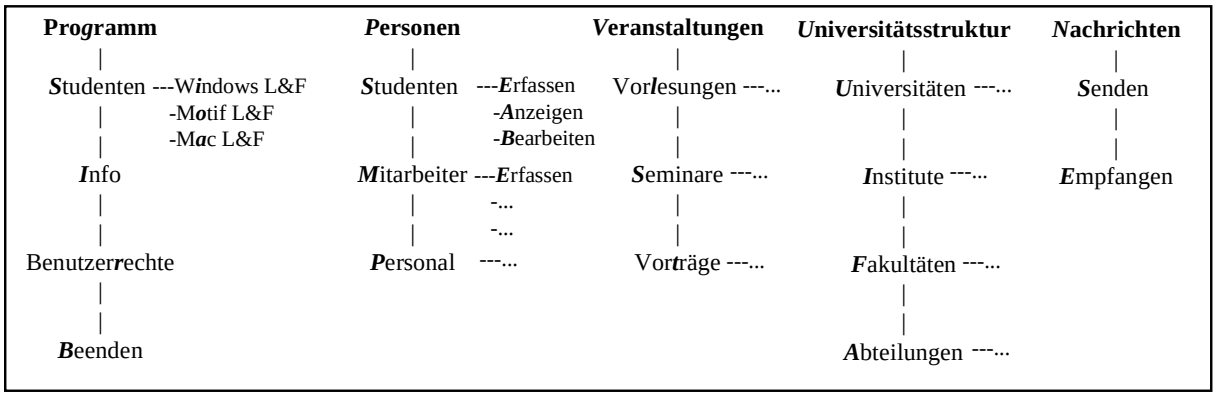


Abbildung 59: Menüstruktur

Die Komponenten der logischen Einheiten werden in einem pull-down-Menü aufgeführt, in der die Aktionen für das Erfassen neuer Daten und das Anzeigen bzw. Bearbeiten vorhandener Daten ausgeführt werden können. Für Benutzer, die häufig mit dem System arbeiten, werden mnemonische Kürzel vorgesehen mit denen die Menüpunkte über die Tastenkombination <Alt+Kennbuchstaben> angewählt werden können.

Es werden ferner folgende Fenstertypen für die Anwendung benötigt:

- Erfassungsmasken
- Übersichtsfenster
- Suchmasken
- Mitteilungsfenster
- Dialogfenster

In den **Erfassungsmasken** können neue Informationen ediert und in der Datenbank abgespeichert werden. Die Eingabefelder werden in logische Gruppen zusammengefaßt und zur Übersichtlichkeit mit einem Rahmen graphisch hervorgehoben. Am Beispiel der Erfassungsmaske für einen Studenten soll der Entwurf verdeutlicht werden:

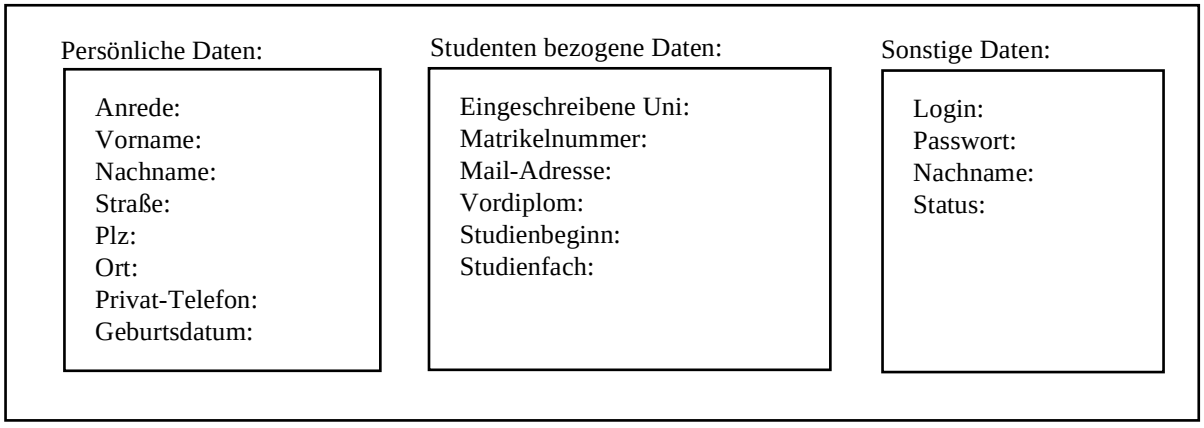


Abbildung 60: Entwurf der Erfassungsmaske für Studenten

Vordefinierte Werte (z. B. Studienbeginn), die in der Datenbank hinterlegt sind, können aus einer Auswahlliste ausgewählt werden. Attribute (z. B. Anrede) zu denen es eine feste Anzahl von Alternativen gibt, werden mittels Optionsfelder dargestellt.

Mit Hilfe von **Suchmasken** kann eine Auswahl aus den vorhandenen Datensätzen eingeschränkt werden. Für textorientierte Attribute können in den Suchmasken Intervalle angegeben werden. Aus diesem Grund werden für diese Attribute zwei Textfelder benötigt, die angeben, ab welchem Wert bzw. bis zu welchem Wert Datensätze selektiert werden sollen. Der prinzipielle Aufbau einer Suchmaske wird am Beispiel der Studenten verdeutlicht:

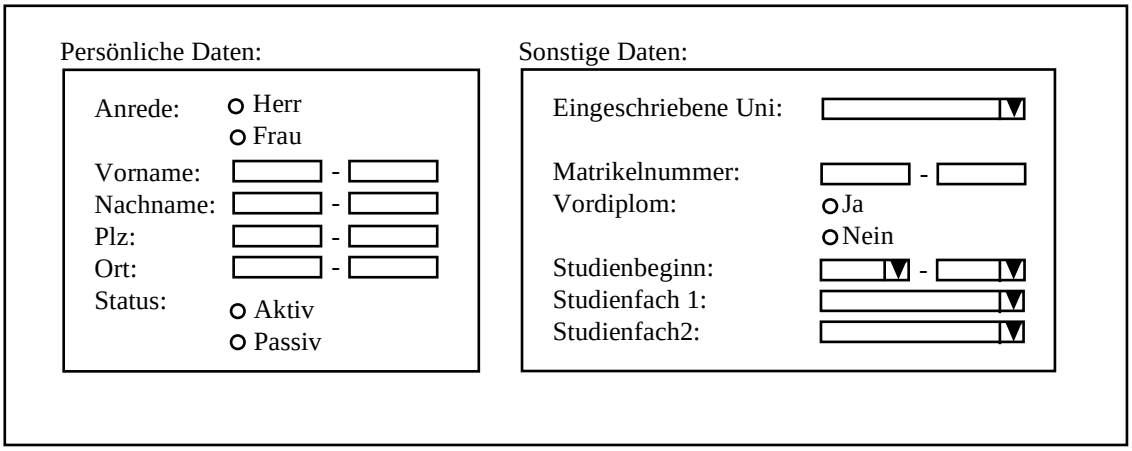


Abbildung 61: Entwurf de Suchmaske für Studenten

Wenn mehr als ein Suchkriterium angegeben wird, werden die Eingaben als UND-Verknüpfung interpretiert. Aus den Eingaben wird daraufhin ein SELECT-Statement für die Anfrage an die Datenbank erzeugt.
Das Ergebnis dieser Suchmaske kann entweder als Einzel- oder Sammelansicht dargestellt werden. Das Erscheinungsbild der **Einzelansicht** entspricht dem einer Erfassungsmaske, wobei zur Navigation zwischen Datensätzen entsprechende Buttons zur Verfügung stehen. Das Ergebnis in der **Sammelansicht** wird in Übersichtsfenster durch eine Tabellenstruktur dargestellt, in der die einzelnen Datensätze horizontal untereinander aufgelistet werden.

Dialogfenster werden für den Sekundärdialog zwischen dem System und dem Benutzer verwendet, z. B. wenn vom Benutzer situationsabhängig Dateneingaben benötigt werden, etwa das Login und Passwort.

Durch **Meldungsfenster** wird der Benutzer über anwendungsspezifische Ereignisse oder über einen bestimmten Verarbeitungszustand hingewiesen. Beispielsweise das falsche Kennwort bei der Anmeldung am System würde ein Meldungsfenster erzeugen, das den Benutzer auf diesen Sachverhalt hinweisen würde.

5.4.3 Ereignisbehandlung

Aktionen, die die Benutzer durch Auswählen von Buttons oder Menüpunkten auslösen, werden durch die Ereignisbehandlung bearbeitet. Die Aktionen finden in einem Fenster statt und infolgedessen erhält jedes Fenster seine eigene Ereignisbehandlung.
Um zwischen der Darstellung der Benutzeroberfläche und der Funktionalität des Systems entsprechend dem Client-Server-Modell eine Unterteilung in logische Komponenten zu garantieren, wird folgende Struktur entworfen:

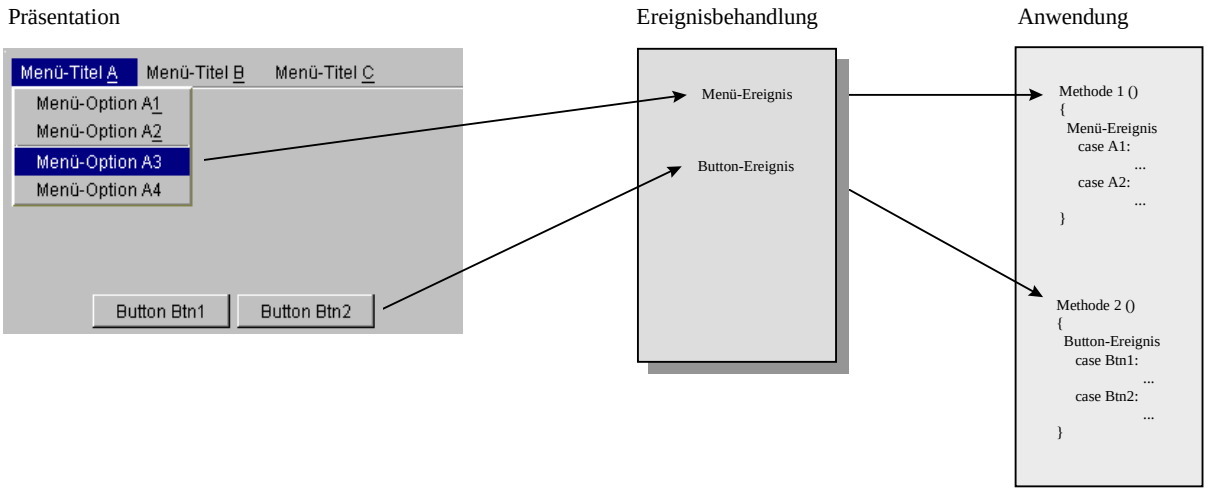


Abbildung 62:Wirkungsweise auf die Ereignisse der Benutzeroberfläche

Wird eine Aktion ausgeführt, dann meldet die Oberfläche synchron an die Ereignisbehandlung, welches Ereignis ausgelöst worden ist. Nach Ermittlung des Ereignisses wird die entsprechende Anwendungsmethode mit einem spezifischen Parameter für das Interaktionselement aktiviert. Innerhalb der Methode wird daraufhin auf das Ereignis reagiert. Dieser callback-Mechanismus, wie er auch bei *Motif* unter *X-Window* verwendet wird, erlaubt eine flexible und übersichtliche Handhabung der Ereignisse.

5.5 Sicherheit

Unberechtigtem Lesen
Da die einzelnen Benutzer bzw. Benutzergruppen in der Regel nur einen bestimmten Teil der Information sehen sollen, werden externe Schemata, sogenannte Sichten (engl. *views)* definiert. Views werden ähnlich einem SQL-Select-Statement aus den Basisrelationen oder anderen Views definiert. Mit Hilfe von Views können somit bestimmte Informationen dem Benutzer verborgen werden. Man spricht daher auch von virtuellen oder abgeleiteten Sichten.

Unberechtigten Ausführen von Funktionen
Mittels des im Datenbankschema entwickelten Teilschemas Benutzerrechte werden die Nutzungsrechte von Aktionen der einzelnen Benutzer verwaltet. Mögliche Aktionen sind beispielsweise:

- Erfassen, Anzeigen und Bearbeiten von Studenten, Mitarbeiter, Vorelsungen, etc.,
- Senden und Empfangen von Nachrichten

Diese Aktionen werden durch Anwählen der Menüpunkte im Menü aktiviert. Jedes Auslösen einer Aktion aus dem Menü bewirkt, daß ein spezifisches Fenster geöffnet wird, in dem die gewünschte Handlung ausgeführt werden kann.
Damit Benutzer nicht Aktionen ausführen können, zu denen sie nicht berechtigt sind, werden beim Authentifizieren durch das Login-Fenster die ausführenden Aktionen aus den Relationen tblFuehrtAus und tblFunktion für den betreffenden Benutzer ermittelt. Beim Laden des Menüs werden daraufhin diejenigen Menüpunkte aktiviert zu denen der Benutzer eine Berechtigung besitzt.

Datenkonsistenz

Datenintegrität
Die semantische Datenintegrität

6 Realisierung des Kommunikations- und Präsentationssystems

6.1 Benutzeroberfläche und Funktionalität

Dieses Kapitel gibt eine Übersicht über die Funktionalität und die Benutzerschnittstelle des Präsentations- und Kommunikationssystems anhand einiger Beispielsitzungen. Aufgrund der umfangreichen Benutzerschnittstelle werden exemplarisch folgende Szenarien aus der Gesamtheit herausgegriffen:

- Anlegen, anzeigen und bearbeiten eines Studenten
- Anlegen einer Abteilung
- Anzeigen von Vorlesungen und Seminaren
- Senden von Nachrichten
- Bearbeiten von Benutzerrechten

Um das Kommunikations- und Präsentationssystem zu starten muß die entsprechende HTML-Seite mit Hilfe eines Appletviewers oder eines Java-fähigen Browsers geladen werden.

Bevor der Benutzer das System anwenden kann, muß er sich mit Hilfe dem „**Login-Fenster**“ anmelden:

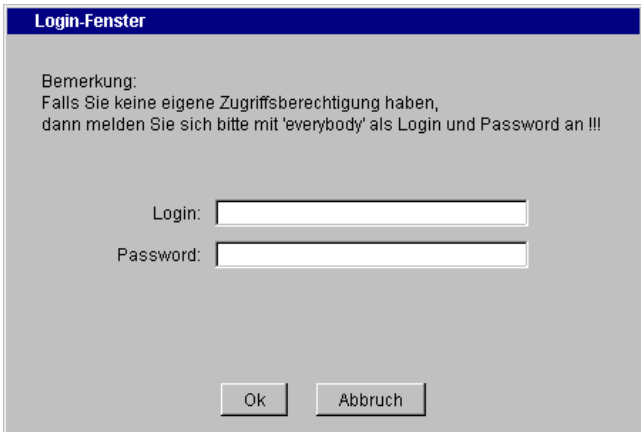


Abbildung 63: Login-Fenster

Nach der Angabe eines Loginnames. des Passwortes und dem Bestätigen der Eingabe wird eine Authentifizierung des Benutzers durchgeführt. Dabei wird eine Anfrage auf die Tabelle *tblPerson* ausgeführt. Wurde der Benutzer erfolgreich identifiziert, dann wird das „Login-Fenster“ geschlossen und das „Hauptmenü“ geöffnet. Gleichzeitig bekommt der Benutzer die Rechte zugeteilt, die zu ihm in der Datenbank hinterlegt sind

Wie in der Anforderungsanalyse in Kapitel *3.1.2 Personen* erwähnt, gibt es auch Benutzer, die nicht in der Datenbank verwaltet werden. Aus diesem Grund wurde ein anonymer Benutzer angelegt mit dem Login und Passwort „***everybody***“. Dieser externe Benutzer hat minimale Rechte, d. h. er kann keine Änderungen in der Datenbank vornehmen, keine Nachrichten versenden und er hat reduzierte Informationsmöglichkeiten.

6.1.1 Das Hauptmenü

Das Hauptmenü ist ein Kopfzeilenmenü mit untergeordneten Pull-Down-Menüs. Die Haupt- und Untermenüpunkte können entweder mit der Maus oder über die Tastenkombination <**Alt**+ entsprechend hervorgehobenen **Kennbuchstaben**> angewählt werden.

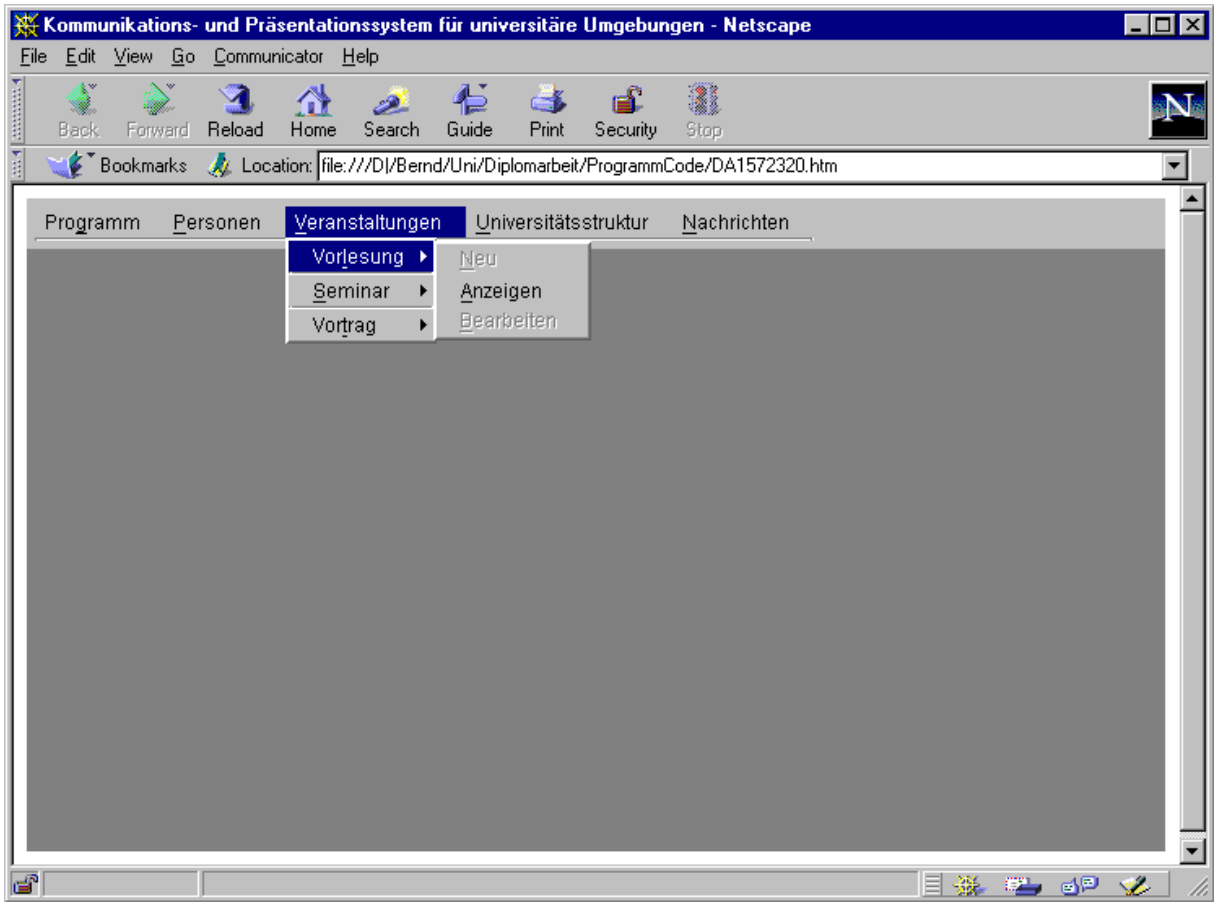


Abbildung 64: Hauptmenü

Am Beispiel des Hauptmenü <***Veranstaltungen***> ist die Struktur der Untermenüs sichtbar, bei der es sich um die typische Ausprägung eines Studenten handelt. Dieser Student hat das Recht sich Vorlesungen anzuzeigen, jedoch nicht die Rechte eine neue Vorlesung anzulegen oder die Inhalte einer Vorlesung zu bearbeiten. Kennzeichnend dafür sind die inaktiven Menüpunkte <***Neu***> und <***Bearbeiten***> des Untermenüs <***Vorlesung***>.

6.1.2 Anlegen eines Studenten

Um einen neuen Studenten in den Datenbestand aufzunehmen, muß der Anwender den Menüpunkt <***Neu***> des Untermenüs <***Studenten***> im Hauptmenü <***Personen***> anwählen.
Diese Aktion kann jedoch nur ausgeführt werden, wenn der Anwender über die entsprechenden Rechte verfügt und damit der Menüpunktes <***Neu***> des Untermenüs <***Student***> aktiv ist.
Jetzt erhält der Anwender die Erfassungsmaske „**Student: Neuanlage**“, in der die Attribute eines Studenten festgelegt werden können. Die Attribute sind in die logischen Gruppen *Persönliche Daten* , *Studenten bezogene Daten* und *sonstige Daten* untergliedert.
Die logische Gruppen *Persönliche Daten* und *Sonstige Daten*, die die Attribute der Tabelle *tblPerson* enthält, sind ebenfalls in den Erfassungsmasken für die Personen **Mitarbeiter** und **Personal** vorhanden.

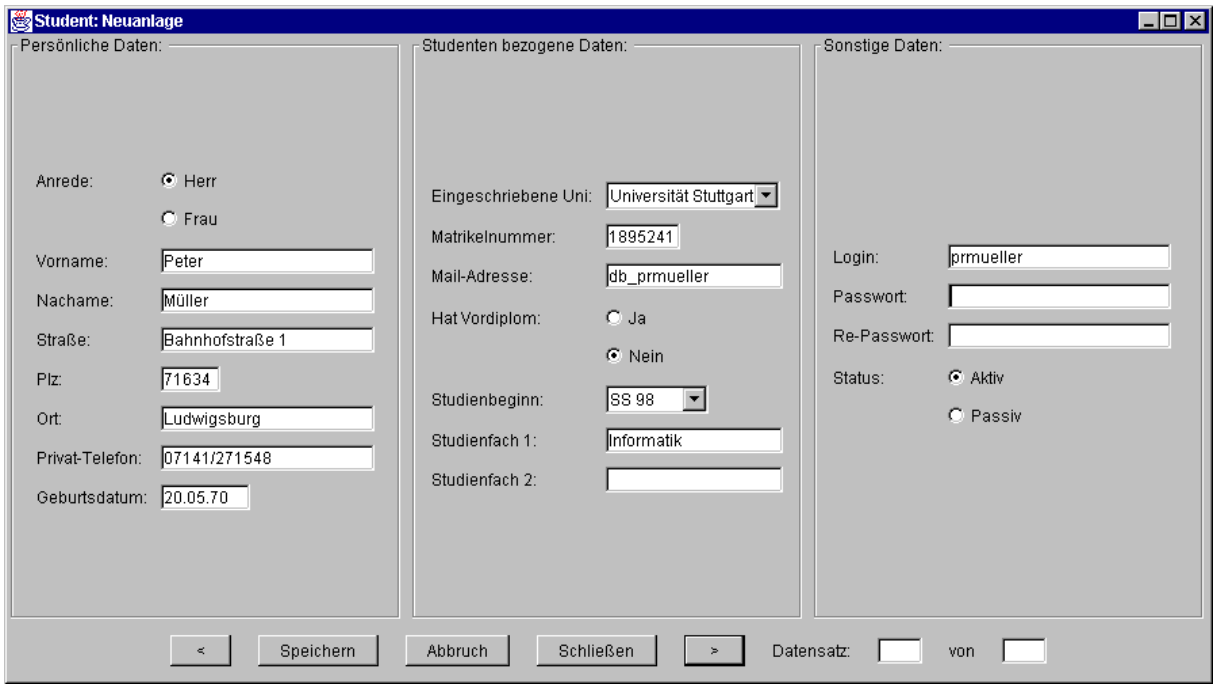


Abbildung 65: Fenster - Student: Neuanlage

Werden die Angaben mit <***Speichern***> bestätigt, erfolgt ein Eintrag in die Datenbanktabellen *tblPerson* und *tblStudent*. Aus den Attributen der Fensterklasse wird ein Objekt der Fachkonzeptklasse Student erzeugt und die Insert-Methode dieses Objektes aktiviert. Diese Insert-Methode erzeugt ein Objekt der Klasse *DBStudent* für die objektorientierte Zwischenschicht, die Eintrag in die Datenbank vornimmt. Wurde der Eintrag erfolgreich durchgeführt, dann werden die Textfelder der Erfassungsmaske geleert und es kann ein neuer Student eingetragen werden.
Bei <***Abbruch***> werden die Eingaben verworfen und mit <***Schließen***> kehrt man zum Hauptmenü zurück.

6.1.3 Anzeigen von Studenten

Um Studenten am Bildschirm anzuzeigen, muß zuerst eine Auswahl über eine Suchmaske getroffen werden. Daher wird beim Anwählen des Menüpunktes <***Anzeigen***> im Untermenü <***Studenten***> die Suchmaske „**Student: Suchen(Anzeigen)**“ geöffnet.
In dieser Suchmaske kann durch die Angabe von Auswahlkriterien die zu suchenden Studenten eingeschränkt werden. Auch hier werden die einzelnen Attribute in die logische Gruppen *Persönliche Daten* und *Sonstige Daten* eingeordnet.

Im unten gezeigten Beispiel wird nach
- Damen und Herren gesucht,
- deren Nachname zwischen dem Intervall ‘Meier’ und ‘Müller’ **und**
- deren Postleitzahl zwischen ‘70000’ und ‘80000’ liegt **und**
- deren Status den Wert ‘Aktiv’ besitzt **und**
- die im Besitz des Vordiplomes sind.

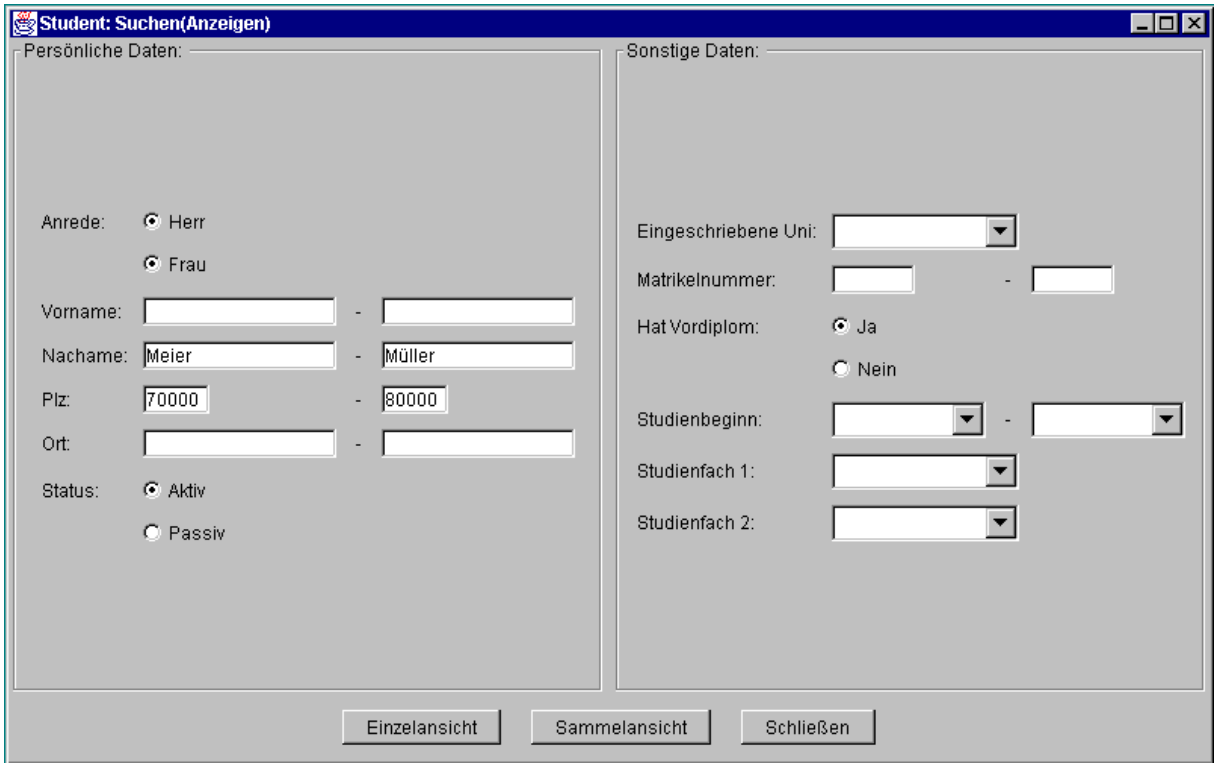


Abbildung 66: Fenster - Student: Suchen(Anzeigen)

Innerhalb der Textfelder können die Wildcards ‘*’ für eine beliebige Zeichenfolge und ’?’ für ein einzelnes Zeichen verwendet werden. Diese Sonderzeichen können in allen Suchmasken verwendet werden.

Die Suche wird gestartet, indem die Eingaben entweder mit <***Einzelansicht***> oder <***Sammelansicht***> bestätigt werden. Es wird ein SQL-SELECT-Statement generiert und auf den Tabellen *tblPerson* und *tblStudent* ausgeführt.

Die Anzeige in der Einzelansicht entspricht der Erfassungsmaske, wobei die Sammelansicht folgendes Erscheinungsbild hat:

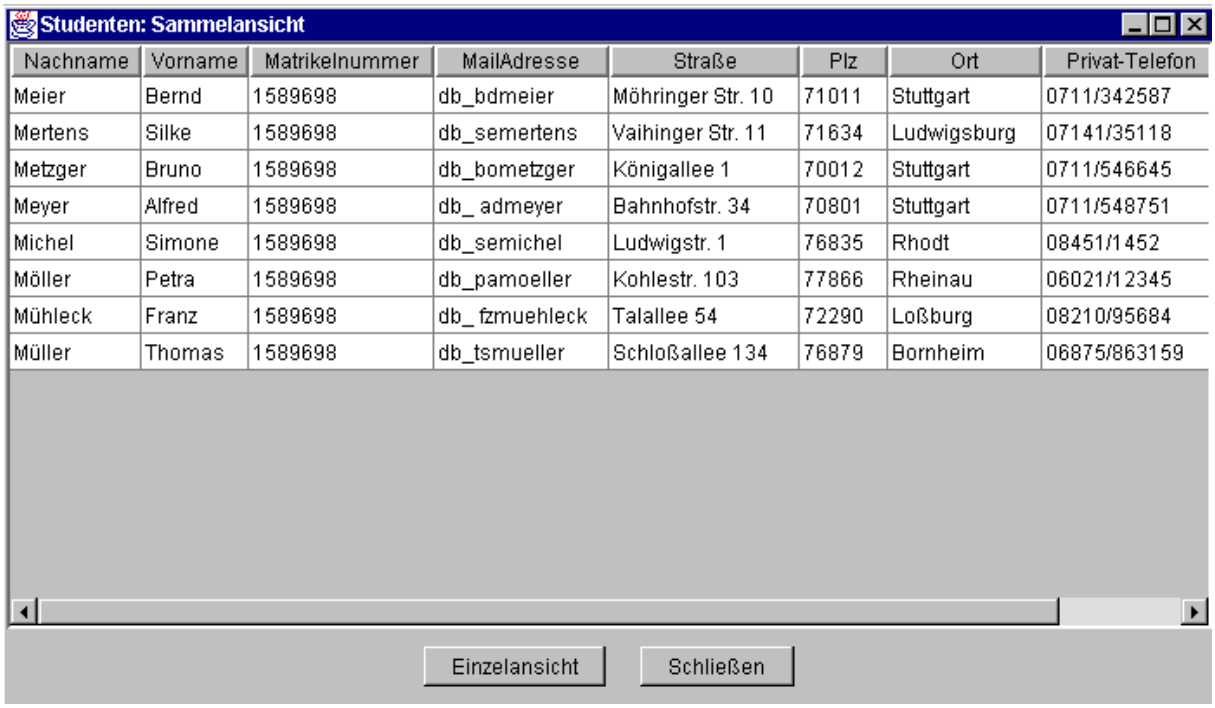


Abbildung 67: Fenster - Student: Sammelansicht

Die Datensätze der Studenten sind in einer Tabellenform in alphabetischer Reihenfolge bzgl. der Nachnamen angeordnet. In der Sammelansicht sind nur die wichtigsten Attribute sichtbar. Um alle Attribute einzusehen, können die gewünschten Studenten markiert und mit <***Einzelansicht***> in die betreffende Erfassungsmaske verschoben werden. In dieser Ansicht werden alle Attribute angezeigt, die dem Nutzungsrecht des Anwenders entsprechen.

6.1.4 Bearbeiten von Studenten

Die Aktion <***Studenten***>-<***Bearbeiten***> erfolgt nach dem gleichen Schema wie <***Studenten***>-<***Anzeigen***>, wobei nur in der Einzelansicht Studenten bearbeitet werden können. Auch hier muß der Anwender das notwendige Recht besitzen, um die Aktion ausführen zu können.
Sind die Studenten ausgewählt, die der Anwender bearbeiten möchte, kann er die Attribute des angezeigten Studenten in der Einzelansicht modifizieren und mittels <***Speichern***> erfolgt die Änderung in der Datenbank.

6.1.5 Anlegen einer Abteilung

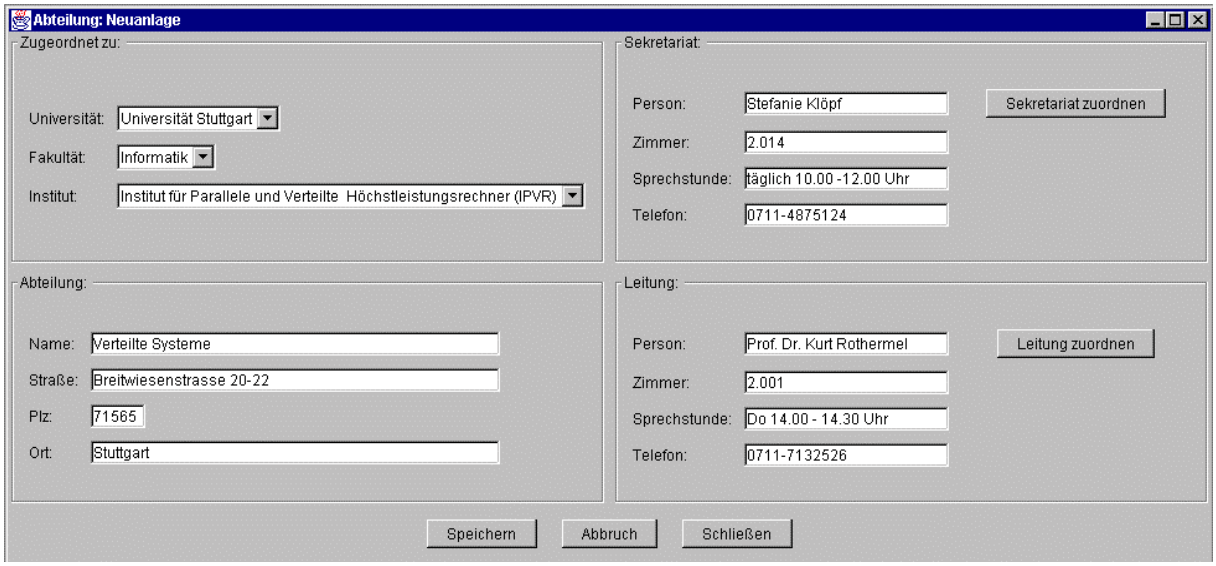


Abbildung 68: Fenster - Abteilung: Neuanlage

6.1.6 Anzeigen von Vorlesungen

Um Informationen über die angebotenen Vorlesungen zu erhalten, können mittels der Suchmaske „**Vorlesung: Suchen(anzeigen)**“ Kriterien für die Einschränkung der gesuchten Vorlesungen eingegeben werden.

Zu diesem Zweck hat der Benutzer folgende Möglichkeiten:
- je ein Auswahlfeld für die Angabe einer Universität, einer Fakultät und des Semesters,
- je ein Textfeld zur Angabe des Titels und der Laufenden Nummer.

Die Angaben für den Titel und die Laufende Nummer in den Textfeldern müssen nicht vollständig sein, es können auch hier die Wildcards ‘*’ für eine Zeichenfolge oder ‘?’ für ein einzelnes Zeichen verwendet werden.

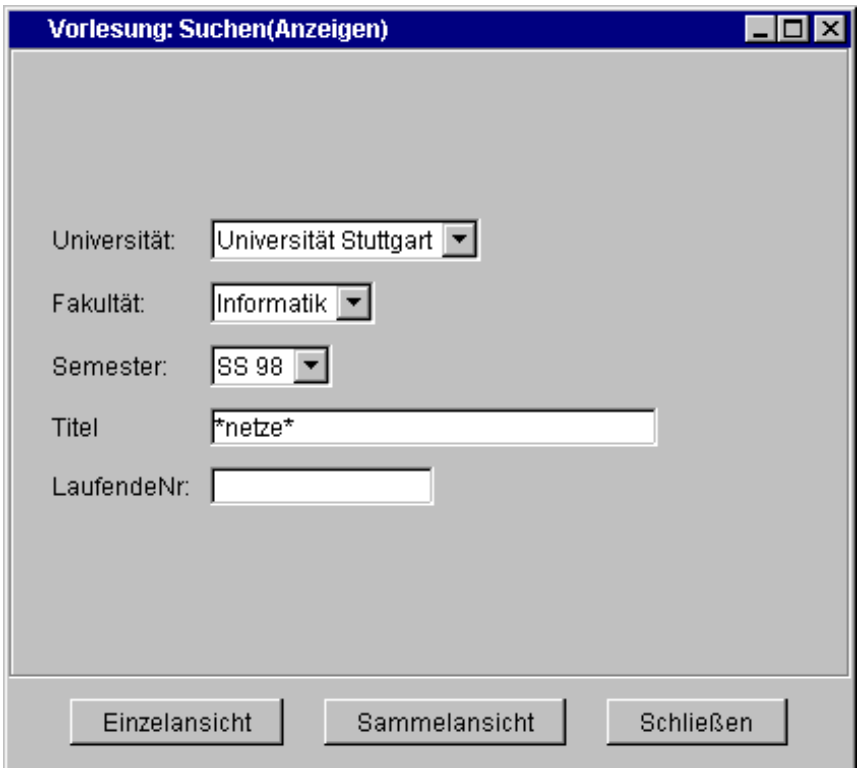


Abbildung 69: Fenster - Vorlesung: Suchen(Anzeigen)

Werden die Eingaben bestätigt, kann das Ergebnis entweder in der Sammelansicht oder in der Einzelansicht angezeigt werden. Die Sammelansicht enthält alle gefundenen Vorlesungen in einer tabellaren Darstellung, jedoch mit eingeschränktem Informationsgehalt, wohingegen die Einzelansicht alle Informationen der Vorlesung enthält.

Das Fenster zur Anzeige der Vorlesungen in der Einzelansicht hat folgendes Erscheinungsbild:

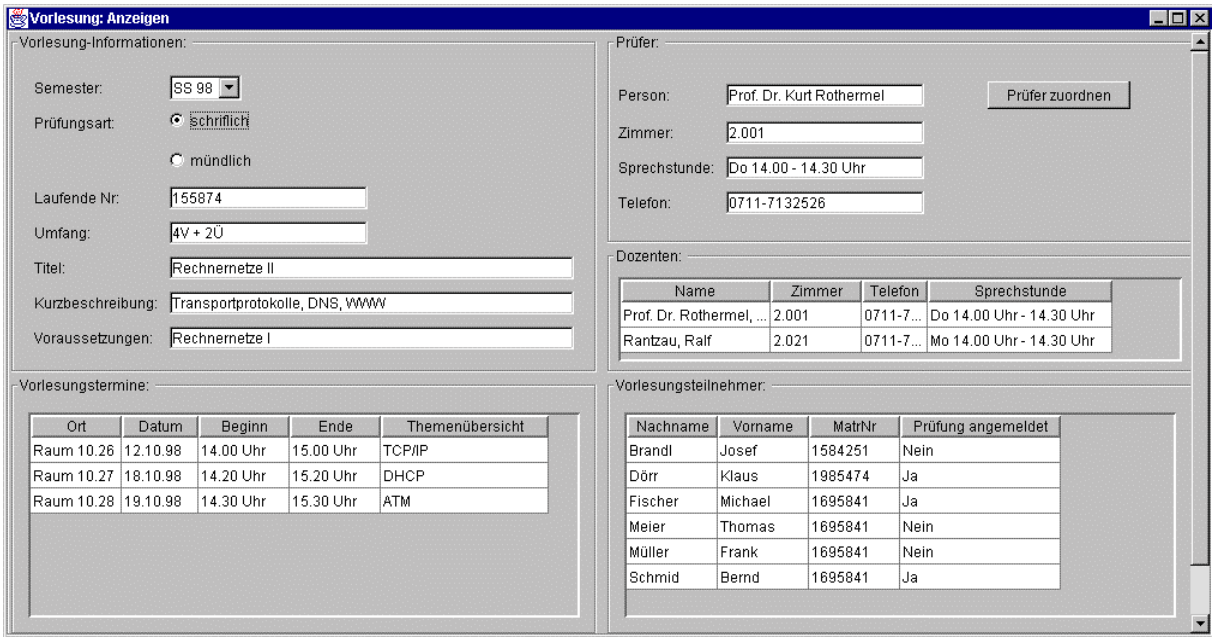


Abbildung 70: Fenster - Vorlesung: Anzeigen

6.1.7 Anzeigen von Seminaren

Das Auswählen zum Anzeigen der Seminare Verläuft nach dem gleichen Prinzip wie bei der Auswahl von Universitäten. Die Suchmaske „**Seminare Suchen(Anzeigen)**“ besitzt dieselben Eingabemöglichkeiten zum Eingrenzen der Auswahl.

Die Einzelansicht für Seminare hat folgendes Aussehen:

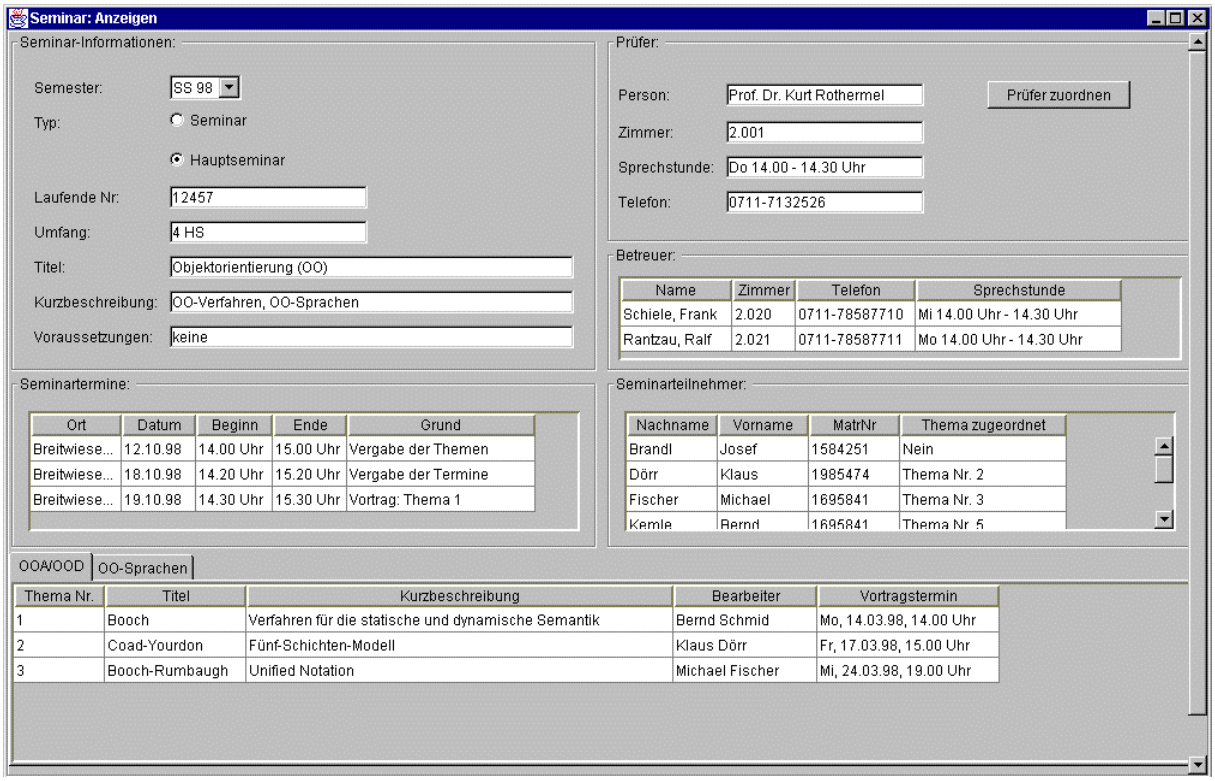


Abbildung 71: Fenster - Seminar: Anzeigen

Die Informationen sind zur Übersichtlichkeit wieder in logischen Gruppen eingeordnet. Dabei handelt es sich um die Gruppen *Seminar-Informationen*, *Seminartermine*, *Prüfer*, *Betreuer*, *Seminarteilnehmer* und *Themengebiete*. Besonderes Merkmal ist das sogenannte *TabbedPane*, in der die einzelnen Themengebiete des Seminars ähnlich einem Karteikasten eingeordnet werden.

6.1.8 Senden von Nachrichten

Die Kommunikation der aktiven Benutzer erfolgt über ein integriertes Tool zum Senden und Empfangen von Nachrichten.

Es gibt zwei Möglichkeiten für das Senden einer Nachricht:

1. Durch Anwählen des Menüpunktes <***Senden***> im Hauptmenü <***Nachrichten***> öffnet sich ein Fenster zum edieren einer Nachricht. Der Benutzer kann nun die Adresse einer oder mehrerer Empfänger, einen Betreff und den Nachrichtentext eintragen. Das Auslösen des Button-Ereignisses <Senden> hat zur Folge, daß ein Objekt der Fachkonzeptklasse *Nachricht* erzeugt und mittels der Klasse *DBNachricht* der objektorientierten Zwischenschicht in die Tabellen *tblNachricht* und *tblErhaelt* eingetragen wird. Zuvor werden die Primärschlüssel des Senders und Empfänger ermittelt, die für die Relationen benötigt werden.
2. Durch markieren einer aktiven Person (siehe unten) und anschließendem Anwählen des Menüpunktes <***Senden***> im Hauptmenü <***Nachrichten***> wird die Adresse der markierten Person automatisch eingefügt. Die nächsten Schritte zum Senden einer Nachricht entsprechen der ersten erwähnten Möglichkeit.

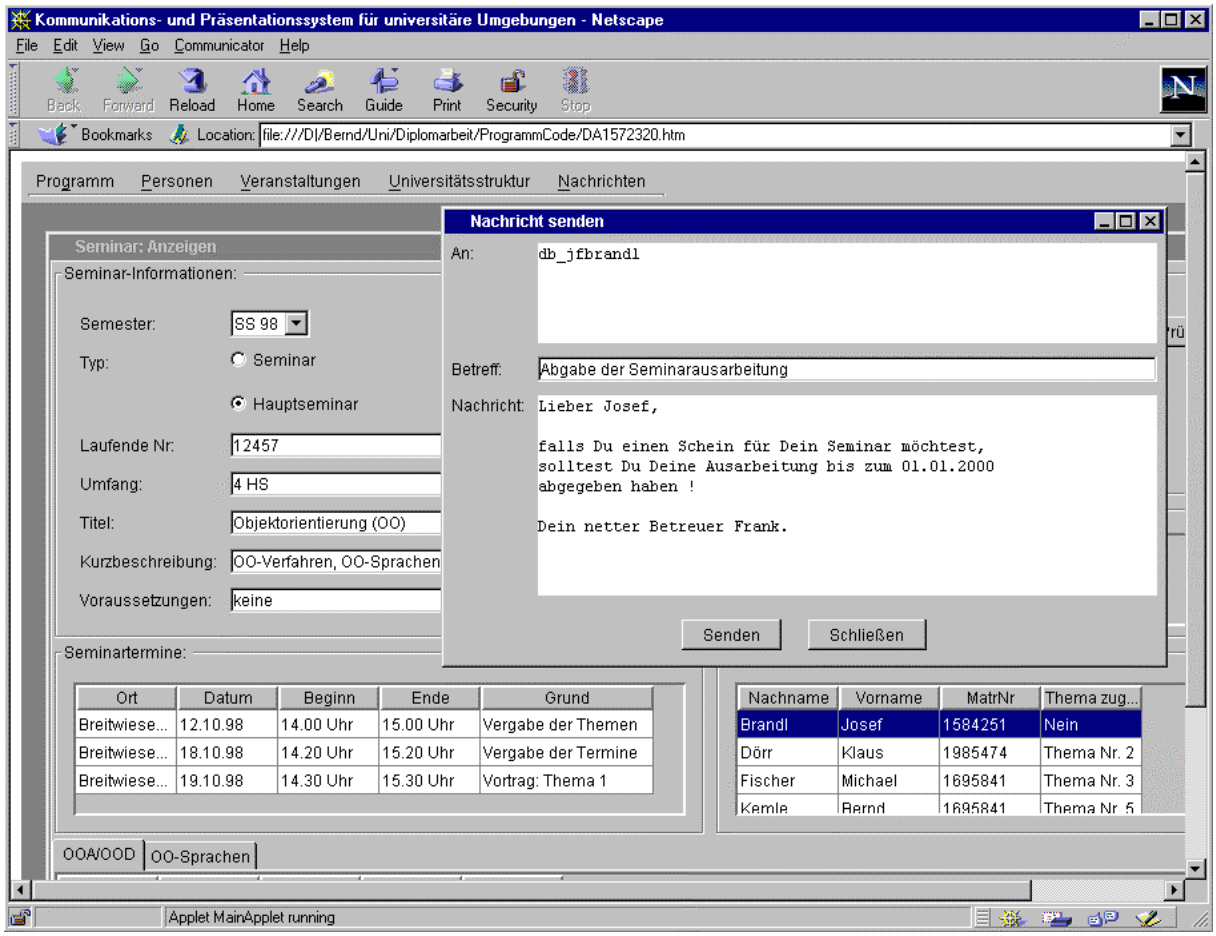


Abbildung 72: Fenster - Nachrichten: Senden

6.1.9 Bearbeiten von Benutzerrechten

Damit autorisierte Anwender die Benutzerrechte bearbeiten können, müssen sie den Untermenüpunkt <***Benutzerrechte***> im Hauptmenü <***Programm***> aktivieren.

Das Fenster „**Benutzerrechte**“ hat folgenden Aufbau:

- Auf der linken Seite sind die drei Personen Personengruppen Student, Mitarbeiter und Personal in je einer Tabelle in alphabetischer Reihenfolge aufgeführt.
- und auf der rechten Seite werden die Nutzungsrechte der markierten Person dargestellt. Die Nutzungsrechte sind entsprechend dem Hauptmenü in logischen Gruppen unterteilt. Innerhalb der logischen Gruppen sind die Menüpunkten aufgelistet, die zu den Hauptmenüs gehören und ausführbare Aktionen sind.

Beim Öffnen des Fensters <***Benutzerrechte***> werden drei Aktionen ausgelöst:
Das erste Aktion initialisiert im Fenster „**Benutzerrechte**“ die Tabelle Studenten. Mit Hilfe der Klasse *DBStudent* aus der objektorientierten Zwischenschicht wird ein Join über den Tabellen *tblPerson* und *tblStudent* gebildet, um die Studenten zu ermitteln. Dabei mit den Werten eines jeden gefundenen Datensatzes ein Objekt der Fachkonzeptklasse Student erzeugt. Für jeden Studenten werden außerdem mittels der Klasse *DBBenutzerechte* ein Join über den Tabellen *tblPerson*, *tblFuehrtAus* und *tblFunktion* gebildet, um die Nutzungsrechte zu ermitteln.
Die zweite und dritte Aktion werden die Tabellen für die Mitarbeiter und das Personal initialisiert. Die Mechanismen entsprechen der ersten erläuterten Aktion.

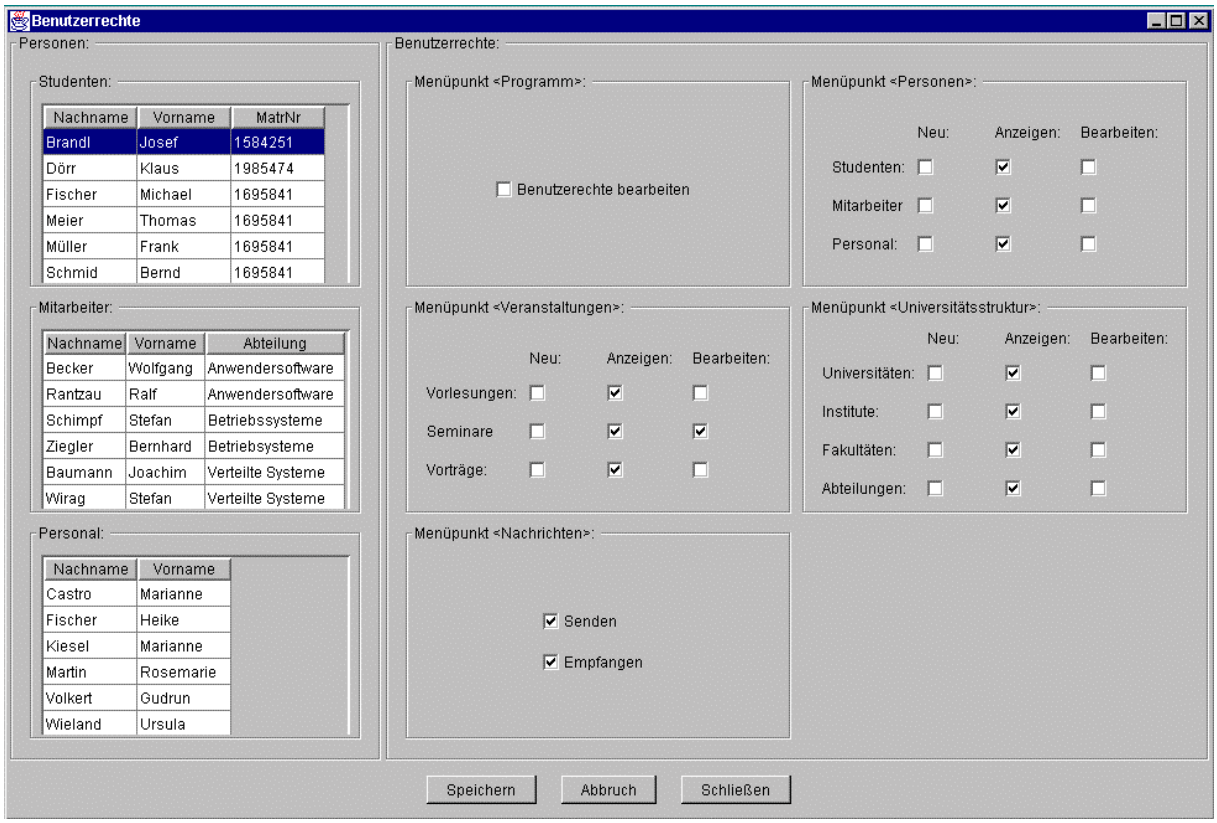


Abbildung 73: Fenster - Benutzerrechte

Sollen beispielsweise die Nutzungsrechte eines Studenten modifiziert werden, dann muß der betreffende Student in der Tabelle für die Studenten markiert werden. Daraufhin werden mittels der Fachkonzeptklasse *Benutzerrechte* die Nutzungsrechte des Studenten in die betreffende Kästchen eingetragen. Ein Häkchen in einem Kästchen bedeutet, daß der Student dieses Recht besitzt. Um dem Studenten ein Recht zu entziehen muß das zuständige Häckchen entfernt und mit <***Speichern***> in der Datenbank aktualisiert werden. Diese Speichern bewirkt ein SQL-Update-Statement auf der Tabelle *tblFuehrtAus*.
Wenn der Student im Anschluß daran das System startet, ist der entsprechende Menüpunkt nicht aktiviert und er kann dadurch die Funktionalität des Menüpunktes nicht ausführen.

6.2 Datenhaltung

Tabellenstruktur für das Teilschema Universitätsstruktur

```
CREATE TABLE tblUniversitaet
   (UniversitaetID Char(12) NOT NULL PRIMARY KEY,
    Name Varchar(100) NOT NULL,
    Strasse Varchar(100),
    Plz Char(5),
    Ort Varchar(100),
    SekretariatPersonal Char(12) NOT NULL,
    SekretariatZimmer Varchar(10),
    SekretariatTelefon Varchar(30),
    SekretariatSprechstunde Varchar(50)
    CONSTRAINT c1 FOREIGN KEY (SekretariatPersonal) REFERENCES tblPersonal
    ON DELETE SET NULL) );

CREATE TABLE tblFakultaet
   (FakultaetID Char(12) NOT NULL PRIMARY KEY,
    Name Varchar(100) NOT NULL,
    Strasse Varchar(100),
    Plz Char(5),
    Ort Varchar(100),
    SekretariatPersonal Char(12) NOT NULL,
    SekretariatZimmer Varchar(10),
    SekretariatTelefon Varchar(30),
    SekretariatSprechstunde Varchar(50)
    Dekan Char(12) NOT NULL,
    ProDekan Char(12) NOT NULL,
    UniversitaetID Char(12) NOT NULL,
    CONSTRAINT c1 FOREIGN KEY (SekretariatPersonal) REFERENCES tblPersonal
    ON DELETE SET NULL,
    CONSTRAINT c2 FOREIGN KEY (Dekan) REFERENCES tblMitarbeiter ON DELETE
    SET NULL,
    CONSTRAINT c3 FOREIGN KEY (ProDekan) REFERENCES tblMitarbeiter ON DELETE
    SET NULL,
    CONSTRAINT c6 FOREIGN KEY (UniversitaetID) REFERENCES tblUniversitaet ON
    DELETE NO ACTION) );

CREATE TABLE tblInstitut
   (InstitutID Char(12) NOT NULL PRIMARY KEY,
    Name Varchar(100) NOT NULL,
    Strasse Varchar(100),
    Plz Char(5),
    Ort Varchar(100),
    SekretariatPersonal Char(12) NOT NULL,
    SekretariatZimmer Varchar(10),
    SekretariatTelefon Varchar(30),
    SekretariatSprechstunde Varchar(50)
    Direktor Char(12) NOT NULL,
    StellvDirektor Char(12) NOT NULL,
    PruefAusschuss Char(12) NOT NULL,
    StudienBeratung Char(12) NOT NULL,
    FakultaetID Char(12) NOT NULL,
    CONSTRAINT c1 FOREIGN KEY (SekretariatPersonal) REFERENCES tblPersonal
    ON DELETE SET NULL,
    CONSTRAINT c2 FOREIGN KEY (Direktor) REFERENCES tblMitarbeiter ON DELETE
    SET NULL,
    CONSTRAINT c3 FOREIGN KEY (StellvDirektor) REFERENCES tblMitarbeiter ON
    DELETE SET NULL,
    CONSTRAINT c4 FOREIGN KEY (PruefAusschuss) REFERENCES tblMitarbeiter ON
    DELETE SET NULL,
    CONSTRAINT c5 FOREIGN KEY (Studienberatung) REFERENCES tblMitarbeiter ON
    DELETE SET NULL,
    CONSTRAINT c6 FOREIGN KEY (FakultaetID) REFERENCES tblFakultaet ON
    DELETE NO ACTION) );
```

```
CREATE TABLE tblAbteilung
   (AbteilungID Char(12) NOT NULL PRIMARY KEY,
    Name Varchar(100) NOT NULL,
    Strasse Varchar(100),
    Plz Char(5),
    Ort Varchar(100),
    SekretariatPersonal Char(12) NOT NULL,
    SekretariatZimmer Varchar(10),
    SekretariatTelefon Varchar(30),
    SekretariatSprechstunde Varchar(50)
    InstitutID Char(12) NOT NULL,
    CONSTRAINT c1 FOREIGN KEY (SekretariatPersonal) REFERENCES tblPersonal
    ON DELETE SET NULL,
    CONSTRAINT c2 FOREIGN KEY (InstitutID) REFERENCES tblInstitut ON DELETE
    SET NULL) );
```

Tabellenstruktur für das Teilschema Personen

```
CREATE TABLE tblPerson
   (PersonID Char(12) NOT NULL PRIMARY KEY,
    Anrede Char(1),
    Vorname Varchar(100),
    Nachname Varchar(100),
    MailAdresse Varchar(200),
    Strasse Varchar(100),
    Plz Char(5),
    Ort Varchar(100),
    PrivatTelefon Varchar(100),
    Status Char(1),
    GeburtsDatum Char(10),
    Login Varchar(25),
    Password Varchar(25) );

CREATE TABLE tblStudent
   (StudentID Char(12) NOT NULL PRIMARY KEY,
    MatrNr Char(7) NOT NULL,
    HatVordiplom Char(1),
    StudienBeginn Char(12) NOT NULL,
    UniversitaetID Char(12) NOT NULL,
    CONSTRAINT c1 FOREIGN KEY (StudienBeginn) REFERENCES tblSemester ON
    DELETE SET NULL,
    CONSTRAINT c2 FOREIGN KEY (UniversitaetID) REFERENCES tblUnivesitaet ON
   DELETE NO ACTION );

CREATE TABLE tblMitarbeiter
   (MitarbeiterID Char(12) NOT NULL PRIMARY KEY,
    Titel Varchar(20),
    Telefon Varchar(100),
    Sprechstunde Varchar(50),
    Typ Varchar(50),
    Zimmer Varchar(10),
    LehrBerechtigt Char(1),
    PruefBerechtigt Char(1),
    AbteilungID Char(12),
    CONSTRAINT c1 FOREIGN KEY (AbteilungID) REFERENCES tblAbteilung ON
    DELETE SET NULL );

CREATE TABLE tblPersonal
   (PersonalID Char(12) NOT NULL PRIMARY KEY);
```

Tabellenstruktur für das Teilschema Benutzerkommunikation

```
CREATE TABLE tblNachricht
   (NachrichtID Char(12) NOT NULL PRIMARY KEY,
    Betreff Varchar(100),
    Datum Char(12),
    Inhalt Varchar(3500),
    Sender Char(12) NOT NULL,
```

```
    CONSTRAINT c1 FOREIGN KEY (Sender) REFERENCES tblPerson ON DELETE SET
    NULL );

CREATE TABLE tblErhaelt
   (PersonID Char(12) NOT NULL PRIMARY KEY;
    NachrichtID Char(12) NOT NULL PRIMARY KEY);
```

Tabellenstruktur für das Teilschema Vorlesungen

```
CREATE TABLE tblAufgabe
   (AufgabeID Char(12) NOT NULL PRIMARY KEY,
    Frage Varchar(1000) NOT NULL,
    Grafik Varchar(200) );

CREATE TABLE tblLoesung
   (LoesungID Char(12) NOT NULL PRIMARY KEY,
    AufgabeID Char(12) NOT NULL PRIMARY KEY,
    Antwort Varchar(1000) NOT NULL,
    Grafik Varchar(200) );

CREATE TABLE tblUebungsGruppe
   (UebungsGruppeID Char(12) NOT NULL PRIMARY KEY
    UebungsGruppenNr Smallint,
    VorlesungID Char(12) NOT NULL
    CONSTRAINT c1 FOREIGN KEY (VorlesungID) REFERENCES tblVorlesung ON
    DELETE SET NULL );
 );

CREATE TABLE tblUebungsTermin
   (UebungsterminID Char(12) NOT NULL PRIMARY KEY,
    Ort Varchar(100),
    Datum Char(12),
    Beginn Char(12),
    Ende Char(12),
    UebungsNr Smallint,
    UebungsGruppeID NOT NULL,
    CONSTRAINT c1 FOREIGN KEY (UebungsGruppeID) REFERENCES tblUebungsGruppe
    ON DELETE SET NULL );

CREATE TABLE tblVorlesung
   (VorlesungID Char(12) NOT NULL PRIMARY KEY,
    SemesterID Char(12) NOT NULL,
    LaufendeNr Varchar(10),
    PruefungsArt Char(1),
    Umfang Varchar(10),
    Titel Varchar(100),
    Kurzbeschreibung Varchar(1000),
    Voraussetzung Varchar(1000),
    Pruefer Char(12),
    CONSTRAINT c1 FOREIGN KEY (SemesterID) REFERENCES tblSemester ON DELETE
    SET NULL );

CREATE TABLE tblVorlesungsTermin
   (VorlesungsTerminID Char(12) NOT NULL PRIMARY KEY;
    Ort Varchar(100),
    Datum Char(12),
    Beginn Char(12),
    Ende Char(12),
    ThemenUebersicht Varchar(1000),
    VorlesungID Char(12) NOT NULL,
    CONSTRAINT c1 FOREIGN KEY (VorlesungID) REFERENCES tblVorlesung ON
    DELETE SET NULL );

CREATE TABLE tblVAnwesend
   (VorlesungsTerminID Char(12) NOT NULL PRIMARY KEY,
    VorlesungsTeilnehmer Char(12) NOT NULL PRIMARY KEY );

CREATE TABLE tblBeaufsichtigt
   (UebungsGruppeID Char(12) NOT NULL PRIMARY KEY,
```

```
    UebungsLeiter Char(12) NOT NULL PRIMARY KEY );

CREATE TABLE tblBesprechen
   (UebungsTerminID Char(12) NOT NULL PRIMARY KEY;
    AufgabeID Char(12) NOT NULL PRIMARY KEY );

CREATE TABLE tblGehoertZu
   (UebungsGruppeID Char(12) NOT NULL PRIMARY KEY;
    VorlesungsTeilnehmer Char(12) NOT NULL PRIMARY KEY );

CREATE TABLE tblHaeltAb
    (VorlesungID Char(12) NOT NULL PRIMARY KEY,
    Dozent Char(12) NOT NULL PRIMARY KEY );

CREATE TABLE tblNimmtAnVTeil
   (VorlesungID Char(12) NOT NULL PRIMARY KEY,
    VorlesungsTeilnehmer Char(12) NOT NULL PRIMARY KEY,
    Note Char(3),
    PruefungAngemeldet Char(1) );
```

Tabellenstruktur für das Teilschema Seminare

```
CREATE TABLE tblSeminar
   (SeminarID Char(12) NOT NULL PRIMARY KEY;
    SemesterID Char(12) NOT NULL,
    Typ Char(1),
    Voraussetzung Varchar(1000),
    Titel Varchar(200),
    Kurzbeschreibung Varchar(1000),
    Umfang Varchar(10),
    Pruefer Varchar(12) );

CREATE TABLE tblSeminatTermin
   (SeminarTerminID Char(12) NOT NULL PRIMARY KEY,
    Ort Varchar(100),
    Datum Char(12),
    Beginn Char(12),
    Ende Char(12),
    Grund Varchar(1000),
    SeminarID Char(12) NOT NULL );

CREATE TABLE tblThema
   (ThemaID Char(12) NOT NULL PRIMARY KEY,
    Titel Varchar(200),
    Kurzbeschreibung Varchar(1000),
    ThemenGruppeID Char(12) NOT NULL );

CREATE TABLE tblThemenGruppe
   (ThemenGruppeID Char(12) NOT NULL PRIMARY KEY,
    Titel Varchar(200),
    Kurzbeschreibung Varchar(1000),
    SeminarID Char(12) NOT NULL );

CREATE TABLE tblVortrag
```

7 Zusammenfassung und Ausblick

Die Auswirkungen des Internet für die Gesellschaft läßt sich nicht in wenigen Worten beschreiben. Es bieten sich ungeahnte Möglichkeiten zur Sättigung des Informationsbedarfes. Es gehört heute zum guten Ton einer Firma im Internet präsent zu sein, damit die Kunden rund um die Uhr mit Informationen befriedigt werden können.

Die immense Zuwachsrate von Anbietern und Nutzern ist besonders durch die Schaffung von benutzerfreundlichen, graphischen Schnittstellen zu den im Internet vorhandenen Ressourcen zu erklären.

einfache Anbindung einer objektorientierten DB
einfache Ergänzung um weitere Veranstaltungen

A Abbildungsverzeichnis

Abbildung 1: Universitätsstruktur 10
Abbildung 2: Entwurfskonzepte 17
Abbildung 3: Enitity-Relationship-Diagramm (ERD) 19
Abbildung 4: Anbindung der Fachkonzeptklassen an die Benutzeroberfläche 22
Abbildung 5: Funktionsweise der objektorientierten Zwischenschicht 23
Abbildung 5: ERD des Teilschemas Universitätsstruktur 26
Abbildung 6: Entitäten des Teilschemas Universitätsstruktur 27
Abbildung 7: Relationen des Teilschemas Universitätsstruktur 27
Abbildung 8: ERD Personen01 28
Abbildung 9: ERD Personen02 29
Abbildung 10: ERD Personen03 30
Abbildung 11: ERD des Teilschemas Personen 30
Abbildung 12: Entitäten des Teilschemas Personen 31
Abbildung 13: Relationen des Teilschemas Personen 31
Abbildung 14: ERD Benutzerkommunikation01 32
Abbildung 15: ERD Benutzerkommunikation02 32
Abbildung 16: ERD Benutzerkommunikation03 33
Abbildung 17: ERD des Teilschemas Benutzerkommunikation 34
Abbildung 18: Entitäten des Teilschemas Benutzerkommunikation 34
Abbildung 19: Relationen des Teilschemas Benutzerkommunikation 34
Abbildung 20: ERD Vorlesung01 35
Abbildung 21: ERD Vorlesung02 36
Abbildung 22: ERD Vorlesung03 36
Abbildung 23: ERD Vorlesung04 37
Abbildung 24: ERD des Teilschemas Vorlesungen 38
Abbildung 25: Entitäten des Teilschemas Vorlesungen 38
Abbildung 26: Relationen des Teilschemas Vorlesungen 39
Abbildung 27: ERD Seminare01 40
Abbildung 28: ERD Seminare02 41
Abbildung 29: ERD Seminare03 42
Abbildung 30: ERD des Teilschemas Seminare 43
Abbildung 31: Entitäten des Teilschemas Seminare 43
Abbildung 32: Relationen des Teilschemas Seminare 44
Abbildung 33: ERD Vorträge Ansatz 1 45
Abbildung 34: ERD Vorträge Ansatz 2 46
Abbildung 35: Relationen des Teilschemas Vorträge 46
Abbildung 36: Entitäten des Teilschemas Vorträge 46
Abbildung 37: ERD des Teilschemas Benutzerrechte 47
Abbildung 38: Relationen des Teilschemas Benutzerrechte 47
Abbildung 39: Entitäten des Teilschemas Benutzerrechte 47
Abbildung 40: Datenbanktabellen aus den Entitäten des Teilschemas Universitätsstruktur 48
Abbildung 41: Datenbanktabellen aus den Relationen des Teilschemas Universitätsstruktur 49
Abbildung 42: Datenbanktabellen aus den Entitäten des Teilschemas Personen. 49
Abbildung 43: Datenbanktabellen aus den Relationen des Teilschemas Personen. 50
Abbildung 44: Datenbanktabellen aus den Entitäten des Teilschemas Benutzerkomm 50
Abbildung 45: Datenbanktabellen aus den Relationen des Teilschemas Benutzerkomm 50

Abbildung 46: Datenbanktabellen aus den Entitäten des Teilschemas Vorlesungen 51
Abbildung 47: Datenbanktabellen aus den Relationen des Teilschemas Vorlesungen 51
Abbildung 48: Datenbanktabellen aus den Entitäten des Teilschemas Seminare 52
Abbildung 49: Datenbanktabellen aus den Relationen des Teilschemas Seminare 52
Abbildung 50: Datenbanktabellen aus den Entitäten des Teilschemas Vorträge 53
Abbildung 51: Datenbanktabellen aus den Relationen des Teilschemas Vorträge 53
Abbildung 52: Datenbanktabellen aus den Entitäten des Teilschemas Benutzerrechte 53
Abbildung 53: Datenbanktabellen aus den Relationen des Teilschemas Benutzerrechte 53
Abbildung 54: Fachkonzeptklassen für das Teilschema Universitätsstruktur 55
Abbildung 55: Fachkonzeptklassen für das Teilschema Personen 56
Abbildung 56: Zwischenschicht DBManager 57
Abbildung 57: Zwischenschicht für das Teilschema Personen 58
Abbildung 58: Ausschnitt aus einer Suchmaske 59
Abbildung 59: Menüstruktur 61
Abbildung 60: Entwurf der Erfassungsmakse für Studenten 61
Abbildung 61: Entwurf der Suchmaske für Studenten 62
Abbildung 62: Wirkungsweise auf die Ereignisse der Benutzeroberfläche 63
Abbildung 63: Login-Fenster 65
Abbildung 64: Hauptmenü 66
Abbildung 65: Fenster - Studenten: Neuanlage 67
Abbildung 66: Fenster - Studenten: Suchen(Anzeigen) 68
Abbildung 67: Fenster - Studenten: Sammelansicht 69
Abbildung 68: Fenster - Abteilung: Neuanlage 70
Abbildung 69: Fenster - Vorlesung: Suchen(Anzeigen) 71
Abbildung 70: Fenster - Vorlesung: Anzeigen 72
Abbildung 71: Fenster - Seminar: Anzeigen 73
Abbildung 72: Fenster - Nachrichten: Senden 74
Abbildung 73: Fenster - Benutzerrechte 75

B Quellcode-Dateien

B1 Fachkonzeptklassen

Person.java
Student.java
Mitarbeiter.java
Personal.java
Universitaet.java
Institut.java
Fakultaet.java
Abteilung.java
Vorlesung.java
Seminar.java
Vortrag.java
Benutzerrechte.java

B2 Klassen für die Benutzeroberfläche und deren Ereignisbehandlung:

AbteilungEinzelClass.java
AbteilungEinzelClassAction.java

BenutzerrechteClass.java
BenutzerrechteClassAction.java

FakultaetEinzelClass.java
FakultaetEinzelClassAction.java

InstitutEinzelClass.java
InstitutEinzelClassAction.java

LoginClass.java
LoginClassAction.java

MenuClass.java
MenuClassAction.java

MitarbeiterEinzelClass.java
MitarbeiterEinzelClassAction.java
MitarbeiterSammelClass.java
MitarbeiterSammelClassAction.java
MitarbeiterSuchenClass.java
MitarbeiterSuchenClassAction.java

PersonalEinzelClass.java
PersonalEinzelClassAction.java
PersonalSammelClass.java

PersonalSammelClassAction.java
PersonalSuchenClass.java
PersonalSuchenClassAction.java

SeminarEinzelClass.java
SeminarEinzelClassAction.java

StudentEinzelClass.java
StudentEinzelClassAction.java
StudentSammelClass.java
StudentSammelClassAction.java
StudentSuchenClass.java
StudentSuchenClassAction.java

UebungenClass.java
UebungenClassAction.java

UniversitaetEinzelClass.java
UniversitaetEinzelClassAction.java
UniversitaetSammelClass.java
UniversitaetSammellClassAction.java

VorlesungEinzelClass.java
VorlesungEinzelClassAction.java

B3 Klassen für die objektorientierte Zwischenschicht

DBManager.java

DBPerson.java
DBStudent.java
DBMitarbeiter.java
DBPersonal.java
DBUniversitaet.java
DBInstitut.java
DBFakultaet.java
DBAbteilung.java
DBVorlesung.java
DBSeminar.java
DBVortrag.java
DBBenutzeerrechte.java

B4 Sonstige Klassen

MainApplet.java

C Literatur

[Bal96] Balzert H.,
„***Lehrbuch der Software-Technik***“,
Spektrum Akademischer Verlag, 1996

[Bau97] Baumann J.,
„***Vorlesung Rechnernetze II***“,
Fakultät Informatik der Universität Stuttgart, SS97

[BeHaGo87] Bernstein P. A., Hadzilacos V., Goodman N.,
„***Concurrency control and Recovery in Database Systems***“,
Addision-Wesley, 1987

[Boo94] Booch G.,
„***Object-Oriented Analysis and Design with Applications***“,
Benjamin/Cummings Publishing Company, 1994

[Cat96] Catell R. R. G.,
„***The Object Database Standard - OMG 93***“,
Morgan Kaufmann Publishers, 1996

[CaWa98] Campione M. ,Walrath K.,
„***The Java Tutorial***“,
Javasoft http://java.sun.com/docs/books/tutorial/, 1998

[Cha96] Chamberlin D.,
„***Using the new DB 2***“,
Morgan Kaufmann Publishers, 1996

[Chen76] Chen P. P.-S.,
„***The Entity-Relationship-Modell - Towards a Unified View of Data***“,
ACM Transaction of Database Systems, 1976

[CoDi88] Coulouris G. F., Dillimore J.
„***Distributed Systems: Concepts and Design***“,
Addision-Wesley, 1988

[CoYo91] Coad P, Yourdon E.,
„***Object-Oriented Design***“,
Yourdon Press, 1991

[Dat90] Date C. J.,
„***An introduction to Database Systems, Volume I***“,
Addision-Wesley, 1990

[DIN88] Deutsches Institut für Normung e. V.,
„***Bildschirmarbeitsplätze - Grundsätze ergonomischer Dialoggestaltung***“,
DIN 66234, 1988

[Ebe92] Eberleh E.,
„***Industrielle Gestaltungsrichtlinien für Benutzerrichtlinien***“,
UNIX-Magazin Ausgabe10, 1992

[EG89] Europäische Gemeinschaft,
„***Mindestvorschriften bzgl. der Sicherheit und des Gesundheitsschutzes bei der Arbeit an Bildschirmgeräten***“,
Amtsblatt der Europäischen Gemeinschaften, 1989

[Jac92] Jacobson I.,
„***Object-Oriented Software Engineering - A Use Case Driven Approach***“
Addision Wesley, 1992

[Jack75] Jackson M. A.,
„***Principles of Program Design***“,
Academic Press, 1975

[Gos91] Goscinski A.,
„***Distributed Operating Systems: The Logical Design***“,
Addision-Wesley, 1991

[HaToWi95] Hahn W., Toenniessen F., Wittkowski A.,
„***Eine objektorientierte Zugriffsschicht für relationale Datenbanken***“,
Informatik Spektrum 18, 1995

[HeHo89] Hertwig R. G., Hommel G.
„***Kooperation und Konkurrenz: Nebenläufige, verteilte und echtzeitabhängige Programmsysteme***“,
Springer Verlag, 1989

[Hop90] Hopcroft J. E., Ullamn J.,
„***Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie***“,
Addision-Wesley, 1990

[IEEE87] IEEE: Institute of Electrical and Electronics Engineers.,
„***Software Engineering Standards***“,
IEEE, 1987

[JaSo98] JavaSoft,
„***JDBC Drivers***“
http://java.sun.com/products/jdbc/jdbc.drivers.html

[JDBC98] JavaSoft,
„***JDBC (Java Database Connectivity)***“

http://java.sun.com/products/jdbc/

[LaLo95] Lang S. M., Lockemann P. C.,
„***Datenbankeinsatz***“,
Springer Verlag, 1995

[Loc83] Lockemann P. C.
„***Systemanalyse***“
Springer Verlag, 1983

[LoSch87] Lockemann P. C., Schmidt J. W.,
„***Datenbank-Handbuch***“.
Springer Verlag, 1993

[LoShRu87] Loomis M. E. S., Shah A. V., Rumbaugh J. E.,
„***An object modelling technique for conceptual design***“
Springer Verlag, 1987

[Mar79] DeMarco T.,
„***Structured analysis and System Specifikation***“
Prentice Hall, 1979

[May92] Mayhew D.,
„***Principles and ‘Guidelines in Software User Interface Design***“
Prentice Hall, 1992

[Mey88] Meyer B.,
„***Object-Oriented Software Construction***“
Prentice Hall, 1988

[Mei72] Meindl U.,
„***Überlegungen zur Informationsbedarfsanalyse bei der Entwicklung von Informationssystemen***“,
Studiengruppe für Systemforschung e. V. Heidelberg, Bericht Nr. 95, 1972

[MS95] Microsoft,
„***The Windows Interface Guideline for Software Design***“
Microsoft Press, 1995

[Mul89] Mulender S.,
„***Distributed Systems***“,
Addision-Wesley, 1989

[MuSnZö75] Mundhenke E., Sneed H., Zöllner U.,
„***Informationssystem für Hochschulverwaltung und -politik***“,
De Gruyter, 1975

[OL90] Sun Microsystems Inc.,
„***OPEN LOOK Graphical User Interface Functional Specification***“
Addision-Wesley, 1990

[OSF94] Open Software Foundation,
„***OSF/Motif Style Guide***“
Prentice Hall, 1994

[Rei82] Reisig W.,
„***Petri-Netze - eine Einführung***“,
Springer-Verlag, 1985

[Reu93] Reuter A.,
„***Vorlesung Informationssysteme und Datenbanken***“,
Fakultät Informatik der Universität Stuttgart,WS93/94

[Roth96] Rothermel K.,
„***Vorlesung Rechnernetze I***“,
Fakultät Informatik der Universität Stuttgart, WS96/97

[Roth97] Rothermel K.,
„***Vorlesung Grundlagen der Verteilten Systeme***“,
Fakultät Informatik der Universität Stuttgart, SS97

[Rum91] Rumbaugh A., Blaha M., Premberlani W., Eddy F., Lorensen W.,
„***Object-Oriented Modelling and Design***“,
Prentice Hall, 1991

[Sch90] Scheer A. W.,
„***Wirtschaftsinformatik - Informationssysteme im Industriebetrieb***“
Springer Verlag, 1990

[ShMe88] Shlaer S., Mellor S. J.,
„***Object-Oriented systems analysis***“
Yourdon Press, 1988

[Shn87] Shneiderman B.,
„***Designing the User Interface: Strategies for Effective Human-Computer Interaction***“,
Addision-Wesley, 1987

[Tan92] Tanenbaum A. S.,
„***Modern Operating Systems***“,
Prentice Hall, 1992

[Tan95] Tanenbaum A. S.,
„***Distributed Operating Systems***“,
Prentice Hall, 1995

[Tan96] Tanenbaum A. S.,
„***Computer Networks 3rd edition***“,
Prentice Hall, 1996

[VDI90] Verein Deutscher Ingenieure,
„***Software-Ergonomie in der Bürokommunikation***“,
VDI-Richtlinie 5005, 1990

[Vet87] Vetter M.,
„***Aufbau betrieblicher Informationssysteme mittels konzeptioneller Datenmodellierung***“,
Teubner Verlag, 1987

[Wan92] Wandmacher J.,
„***Software-Ergonomie***“,
De Gruyter, 1992

[Warr78] Warrier J.-D.,
„***Locial Construction of Programs***“,
Van Nostrand, 1978