

Fortgeschrittenenpraktikum

Prof. Dr. Martin Wirsing

**Lehr- und Forschungseinheit für
Programmierung und Softwaretechnik der
Ludwig-Maximilians-Universität München**

Betreuerin: Nora Koch

Bearbeiter: Christian Walter

1 Inhaltsverzeichnis

1	Inhaltsverzeichnis	2
2	Einleitung	4
3	Beschreibung des Ist- und Soll-Zustands	5
3.1	<i>Beschreibung des aktuellen Immobiliendatenbanksystems</i>	5
3.2	<i>Beschreibung der zukünftigen Immobiliendatenbank</i>	5
3.3	<i>Grund für die Neuprogrammierung</i>	5
4	Anforderungsanalyse	5
4.1	<i>Anforderungen an den Inhalt</i>	6
4.2	<i>Anforderung an die Funktionalität</i>	7
4.2.1	Allgemeine Beschreibung zu Use-Case Diagrammen	7
4.2.2	Use-Case Diagramm zur Immobiliendatenbank	8
4.3	<i>Anforderung an die Präsentation</i>	26
5	Design	26
5.1	<i>Allgemeine Beschreibung zum Design</i>	26
5.2	<i>Domänenmodell</i>	26
5.2.1	Allgemeine Beschreibung des Domänenmodells	26
5.2.2	Anmerkung zum Domänenmodell	27
5.2.3	Das Domänenmodelldiagramm	28
5.2.4	Beschreibung der einzelnen Klassen	29
5.3	<i>Navigationsmodell</i>	32
5.3.1	Allgemeine Beschreibung des Navigationsmodells	32
5.3.2	Navigationsklassenmodell	32
5.3.3	Navigationsstrukturmodell	34
5.3.4	Bemerkung zum Navigationsstrukturmodell	34
5.3.5	Erklärung der Notation	34
	Fall: Der nicht registrierte Benutzer	35
	Fall: Der registrierte Benutzer	36
	Fall: Administration	37
	Das gesamte Navigationsstrukturmodell	38
5.4	<i>Präsentationsmodell</i>	39
5.4.1	Statisches Präsentationsmodell	39
	Erklärung der Notation	39
	Das statische Präsentationsmodell der Anwendung	40
6	Implementierung	43
6.1	<i>Allgemeine Aspekte der Implementierung</i>	43
6.2	<i>Probleme bei der Implementierung</i>	43
6.3	<i>Zusammenfassung der Technischen Daten</i>	44
7	Wartung und Sicherheit	44
7.1	<i>Wartung des Programms</i>	44

7.2	<i>Systemrecovery</i>	44
8	Erweiterbarkeit	45
9	Zusammenfassung	45
10	Literaturverzeichnis	46
11	Anhang A: Auszüge aus dem Quelltext	47
12	Anhang B: Beispiel für die fertige Applikation	48

2 Einleitung

Ziel dieses Fortgeschrittenenpraktikums ist das Reengineering einer Anwendung zum Angebot von Immobilienobjekten im World Wide Web.

Die bestehende Immobiliendatenbank erfüllt nicht mehr die gestiegenen Anforderungen der Anwender. Durch die verschiedenen WWW – Angebote empfindet es der Benutzer als wünschenswert, daß er zum Beispiel seine persönlichen Benutzerprofile abspeichern kann oder daß die Navigation ihn möglichst schnell zum gewünschten Ziel führt. Um diese Anforderungen zu erfüllen, wird beim Reengineering ein strukturierter Software-Entwicklungsprozeß verwendet, der in mehrere Phasen unterteilt ist. Dieser Prozeß ist in folgende Phasen unterteilt:

- Erfassen des Ist- und Soll-Zustandes (Kapitel 3)
- Die Anforderungsanalyse (Kapitel 4)
- Das Design (Kapitel 5)
- Die Implementierung (Kapitel 6)
- Die Wartung (Kapitel 7)
- Die Erweiterbarkeit (Kapitel 8)

Der Ist- und Soll-Zustand der Anwendung wird an Hand vom Anwendungsgebiet und der Funktionalität der aktuellen und der neuen Anwendung erklärt. Diese Phase wird dazu benutzt, um sich mit der jetzigen und zukünftigen Anwendung vertraut zu machen. In der Anforderungsanalyse werden die Erkenntnisse aus Kapitel 3 zur Festlegung der Anforderungen an den Inhalt, an die Funktionalität und an die Präsentation verwendet. Der Inhalt und die Präsentation werden textuell festgelegt. Die Anforderungen an die Funktionalität werden mit Hilfe von Use-Case Diagrammen definiert. Die Design Phase baut auf den Spezifikationen der Analyse Phase auf. Als Design Methode wird die in [Koch & Mandel,1999] beschriebene Methode verwendet, die auf OOHDM (Object-Oriented Hypermedia Design Methodology) basiert. Dies ist eine dreistufige Methode, bestehend aus der Modellierung des Domänenmodells, des Navigationsmodells und des Präsentationsmodells. Die Design Phase läuft als iterativer Prozeß ab, der solange wiederholt wird, bis er zum gewünschten Ziel führt (siehe Abbildung 1).

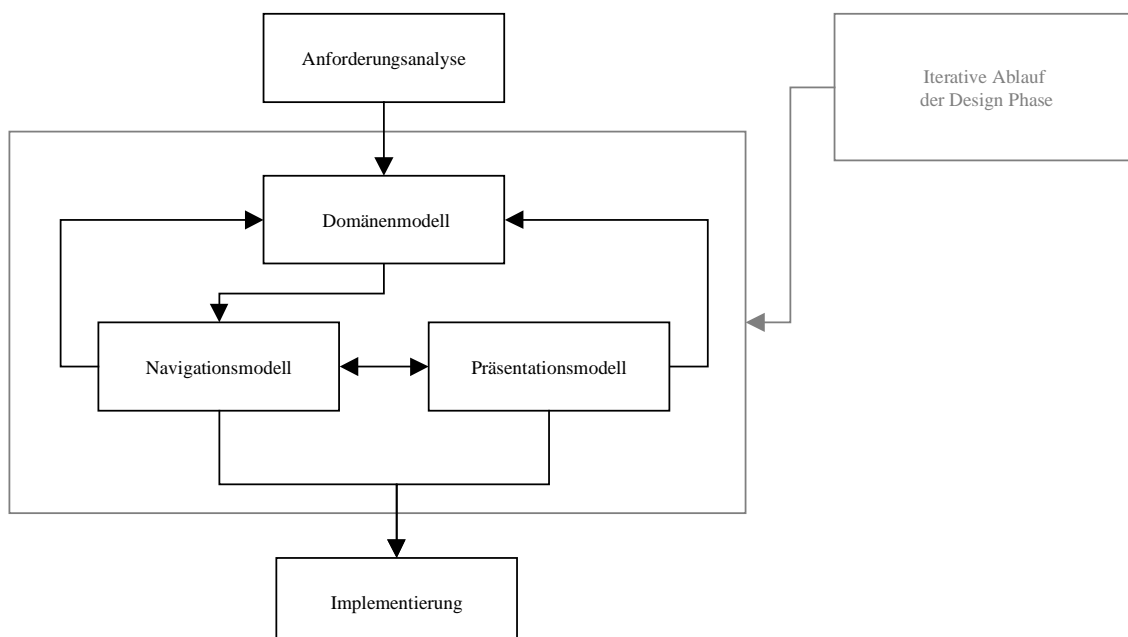


Abbildung 1: Iterativer Prozeß der Design Phase

Nach Abschluß der Design Phase ist die ganze Anwendung modelliert, d.h. es ist vor der Implementierung festgelegt, was die Anwendung leistet, wie sie zu bedienen ist, für welche Personengruppen sie entwickelt wird.

In Kapitel 6 wird die Implementierung beschrieben, es wird darauf eingegangen, wie das objektorientierte Modell der Design Phase in der Kombination bestehend aus einer prozeduralen Sprache und einem Datenbanksystem realisiert wurde.

Kapitel 7 befaßt sich dann mit dem Aspekt Wartung. Außerdem wird auf Datensicherung und -wiederherstellung eingegangen. Zum Schluß wird in Kapitel 8 auf einige Erweiterungsmöglichkeiten für die Anwendung verwiesen.

3 Beschreibung des Ist- und Soll-Zustands

3.1 Beschreibung des aktuellen Immobiliendatenbanksystems

Die Immobiliendatenbank ist entstanden, um die Vermittlung von Immobilienobjekten zu unterstützen. Sie ist über das Internet abfragbar, um möglichst vielen Interessenten einen einfachen Zugang zu den Objekten zu verschaffen. Man kann nach Immobilienobjekten suchen, wobei die Suche durch die wenigen Arten von Immobilienobjekten sehr eingeschränkt ist. Außerdem ist es nicht möglich mehrerer Arten, wie zum Beispiel 1- und 2 – Zimmerwohnungen, gleichzeitig zu suchen. Die Objekte sind außerdem zu ungenau beschrieben, um Interesse beim Anwender zu wecken. Der Vergleich mit anderen Produkten aus dieser Kategorie und die daraus resultierenden Schlüsse werden genauer im Abschnitt 4.1 beschrieben.

3.2 Beschreibung der zukünftigen Immobiliendatenbank

Bei der Neukonzeption der Immobiliendatenbank sollen noch zwei weitere Datenbanken implementiert werden: Grundstücks- und Mietdatenbank. Die Trennung in Grundstücke und Immobilien wird die Suche erleichtern. Das Hinzufügen von Mietobjekten rundet das Angebot ab. Alle drei Datenbanken sollen zur Vermittlung von Mietwohnungen und -häuser, Grundstücken und Kaufobjekte verwendet werden. Die Datenbanken sollen über das Internet abfragbar sein. Der Kunde kann sich so auf einfachem Weg über aktuelle Objekte informieren. Weiterhin soll dem Benutzer die Möglichkeit gegeben werden, seine persönlichen Suchkriterien abzuspeichern, damit bei häufiger Benutzung der Immobiliendatenbank, die Wahl der Suchkriterien wegfällt und die Navigation zum Ziel damit verkürzt wird.

3.3 Grund für die Neuprogrammierung

Die alte Immobiliendatenbank basierte auf der Mini–SQL Version 1.x. Die Version 2.0.x ist schneller, hat weniger Fehler und verfügt über einen größeren Funktionsumfang. Die Umstellung auf die neue Version bedingt aber eine Neuprogrammierung, da sich die Syntax der Programmiersprache geändert hat.

4 Anforderungsanalyse

4.1 Anforderungen an den Inhalt

Aus den Erfahrungen mit der alten Immobiliendatenbank und dem Vergleich mit anderen Produkten, wie z.B. Internet Immobilien (www.inim.de), Webserat (www.webserat.com) oder Immo-Broker (immo-broker.net) machte man nachfolgende Feststellungen:

Die Suchvorgänge nach Immobilienobjekten beschränken sich meistens auf den Preis oder die Wohnfläche. Wichtige Kriterien wie die Wohnungsart oder die Postleitzahlenregion fehlen oftmals. Nachdem man ein Objekt gefunden hat, sind die Angaben oftmals sehr ungenau. Es fehlen Angaben wie z.B. die Verkehrsanbindung, Stockwerk, Parkplatz u.s.w.. Außerdem vermißt man oftmals Bilder zum Objekt und Informationen zum Grundriß.

Genauere Suchfunktionen und mehr Informationen über die gefundenen Objekte werden mit Sicherheit das Kundeninteresse fördern.

Diese Erkenntnisse führten zu folgenden Vorgaben:

Die Datenbank wird aufgeteilt, so daß man je eine Datenbank für Immobilienobjekte, für Grundstücksobjekte und für Mietobjekte erhält.

Die Objekte sollen folgende Details beinhalten:

Für alle drei Datenbanken:	Zusätzlich für Miet- und Immobiliendatenbank:	Zusätzlich nur für die Mietdatenbank:
➤ Objektidentität	➤ Aufzug	➤ Raucher / Nichtraucher
➤ Art des Objekts	➤ Stockwerk	➤ Haustiere Ja/Nein
➤ Grundfläche	➤ Parkplatz	
➤ Preis / Miete	➤ Zimmeranzahl	
➤ Courtage / Provision		
➤ Postleitzahl		
➤ Ort		
➤ Beschreibung		
➤ Bild		
➤ Grundriß		
➤ Verkehrsanbindung		
➤ Lage		
➤ Ansprechpartner		
➤ Emailadresse		
➤ Telefonnummer		
➤ Faxnummer		

Anmerkung zur Art des Objekts:

Jedes Objekt gehört einer bestimmten Art an, z.B. 1-Zimmer-Wohnung oder 2-Zimmer-Wohnung. Jede Art wird in einer der Kategorien eingeteilt, wie z.B. Wohnungen, freistehende Häuser oder Gewerbeobjekte. Die Kategorien sowie die verschiedenen Arten von Objekten werden vorgegeben und dementsprechend in das Programm eingebunden. Die Vorgaben erhält man von den Firmen, die die Anwendung kaufen.

Für die Suche in den Datenbanken sollen dem Benutzer folgende Suchkriterien zur Verfügung stehen:

- Man kann nach mehreren Arten von Objekten oder Kategorien gleichzeitig suchen.
- Die Arten sollen Kategorien zugeordnet werden, um eine einfache Auswahl zu ermöglichen.
- Die Suchkriterien Preis, Wohnfläche (falls vorhanden), Grundfläche, Postleitzahl bleiben bestehen.
- Der Benutzer kann auswählen, wie das Ergebnis sortiert werden soll .

Wenn ein Objekt gefunden wird, soll es gleich angezeigt werden. Sind mehrere Objekte gefunden, soll der Benutzer die Möglichkeit haben, aus einer Liste auszuwählen, welches Objekt er am Bildschirm ansehen möchte.

Weiterhin soll der Benutzer die Möglichkeit haben, für jede Datenbank ein Benutzerprofil anzulegen, das seine persönlichen Suchkriterien speichert. Um diesen Dienst nutzen zu können, muß er sich aber registrieren. Das geschieht durch die Angabe seiner Anschrift und seiner Email-Adresse. Der registrierte Benutzer wird automatisch per Mail benachrichtigt, falls für ihn ein neues oder geändertes Objekt, das mit seinen persönlichen Suchkriterien übereinstimmt, in die Datenbank aufgenommen wird. Die Registrierung gilt für alle drei Datenbanken, wobei der Benutzer für jede Datenbank eigene Suchkriterien definieren kann. Außerdem hat er die Option, die Suchkriterien zu ändern, zu löschen oder die Registrierung komplett zu löschen.

Den Administratoren der Datenbanken sollen nachfolgende Aktionen über das Internet zur Verfügung stehen:

- Sie sollen in allen drei Datenbanken Objekte einfügen, ändern und löschen können.
- Die Objekte sind mit einem Paßwort versehen, so daß nicht jeder Administrator ein Objekt ändern kann.
- Sie sollen die Möglichkeit haben, bestehende Objekt aus den Internet zu nehmen, ohne diese zu löschen
- Jeder Administrator kann für jede der drei Datenbanken einen eigenen Mailtext eintragen.
- Der Administrator bekommt bei einer Objektänderung automatisch eine Liste über Personen, die registriert sind und deren gespeicherte Suchkriterien mit den Objektdaten übereinstimmen, per Mail zugesendet.

4.2 Anforderung an die Funktionalität

4.2.1 Allgemeine Beschreibung zu Use-Case Diagrammen

(dt. Anwendungsfalldiagramme)

Ein Anwendungsfalldiagramm visualisiert die Berührungspunkte zwischen der zu entwickelnden Software und dem Umfeld. Weiterhin beschreibt es die Aktivität oder die Funktionalität, die das entstehende Softwaresystem unterstützen soll. Use-Cases sind kein primärer Designansatz, sondern sie dienen zur Kommunikation zwischen Entwickler und Anwender, um das Verhalten des Systems, d.h. was es leistet und wie man mit ihm umgeht, für beide Seiten verständlich darzustellen.

Bei der Entwicklung eines Anwendungsfalldiagramms muß man als erstes die verschiedenen Akteure (Rollen) bestimmen. Jeder Benutzer, d.h. Anwender oder externes System, wird in

eine Klasse von Akteuren eingeteilt. Die Akteure sind an mindestens einem der in dem Anwendungsfall (siehe unten) beschriebenen Interaktionen beteiligt.

Die zu entwickelnde Software wird in verschiedene Anwendungsfälle unterteilt. Komplexe Diagramme werden hierarchisch gegliedert, d.h. ein Anwendungsfall in einem Diagramm wird durch ein zusätzliches, detaillierteres Anwendungsfalldiagramm beschrieben.

Zum bessern Verständnis werden teilweise Aktivitätsdiagramme hinzugefügt, da Aktivitätsdiagramme Zusammenhänge, die anwendungsfallübergreifend sind, besser beschreiben, als dies textuell oder grafisch in den Use-Cases möglich wäre.

4.2.2 Use-Case Diagramm zur Immobiliendatenbank

Aus den Angaben zur Immobiliendatenbank ergaben sich nachfolgende Rollen:

- Administrator
- Registrierter Benutzer, das sind Benutzer, die sich registrieren wollen oder registriert sind.
- Benutzer, das sind Benutzer, die sich weder registrieren wollen noch registriert sind.
- System

Außerdem resultieren aus den Angaben folgende Anwendungsfälle:

- Administration (das Subsystem Objekt bearbeiten, wird noch separat beschrieben)
- Objekte bearbeiten
- Benutzerregistrierung
- Standard Suche
- Mailsystem

Aus den Angaben ergibt sich, das in Abbildung 2 gezeigte Use-Case Diagramm

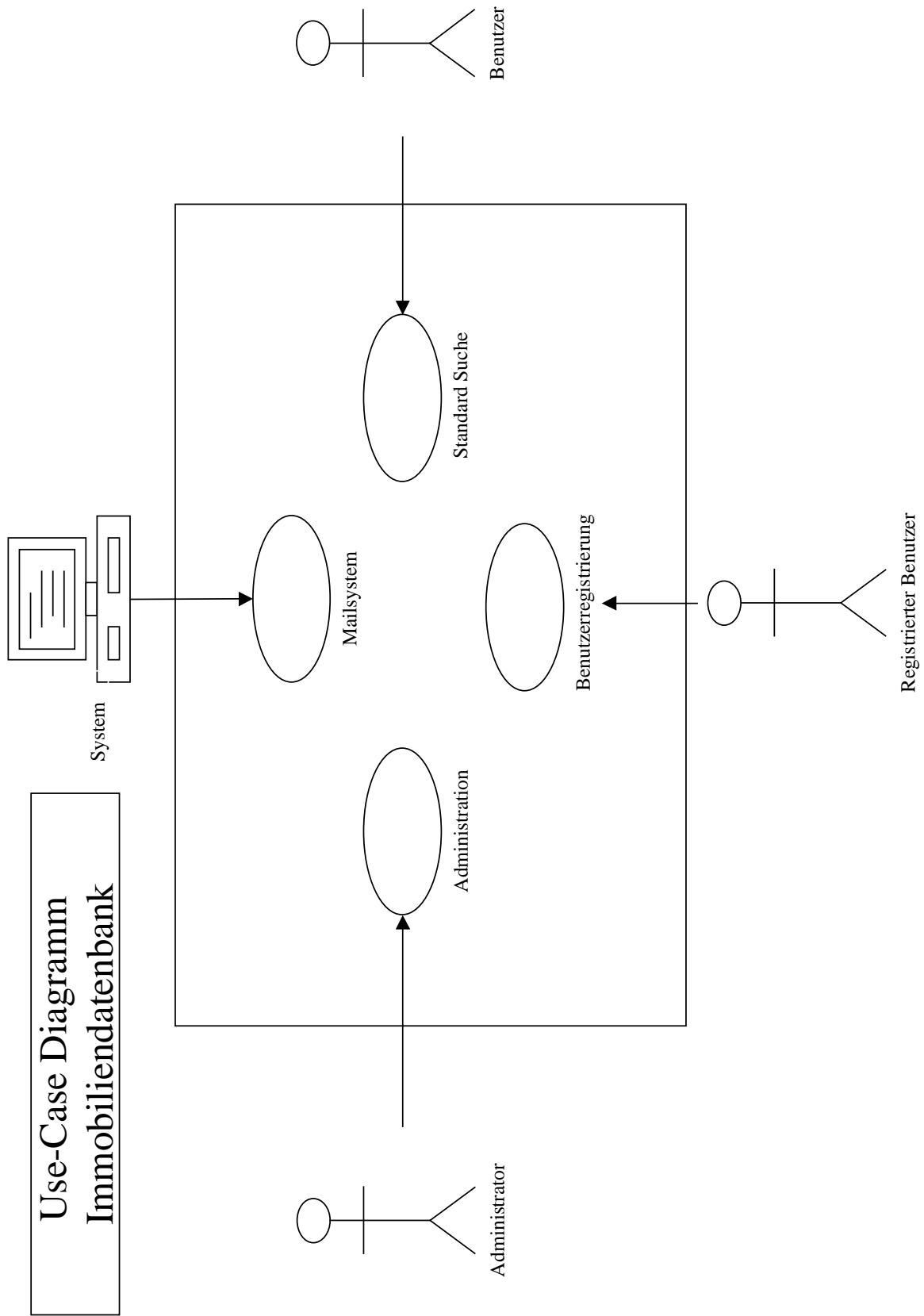


Abbildung 2: Das Use-Case Diagramm der Immobilienbank

Akteur: Administrator

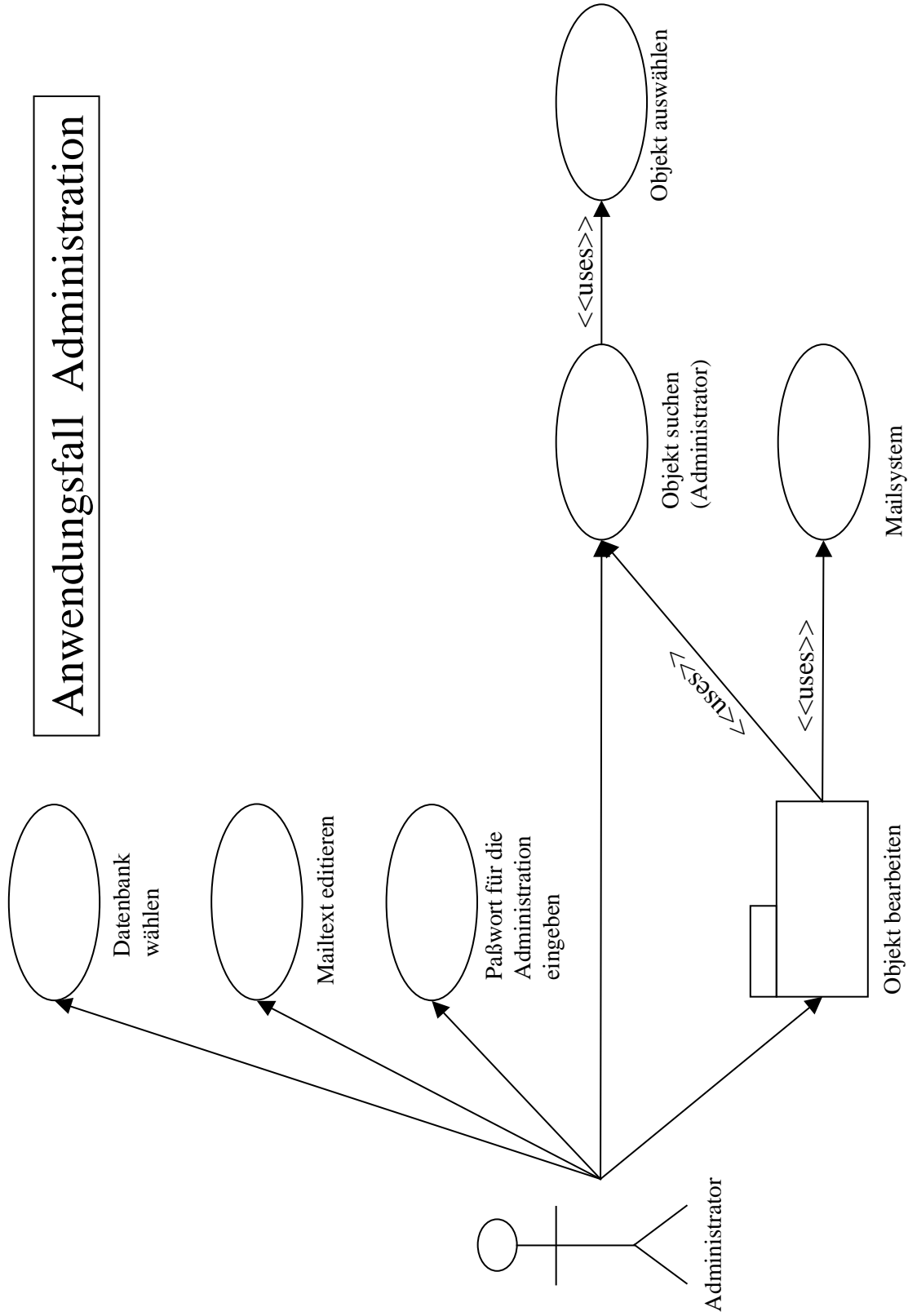


Abbildung 3: Das Anwendungsfalldiagramm für den Administrator

Aktivitätsdiagramm für den Administrator

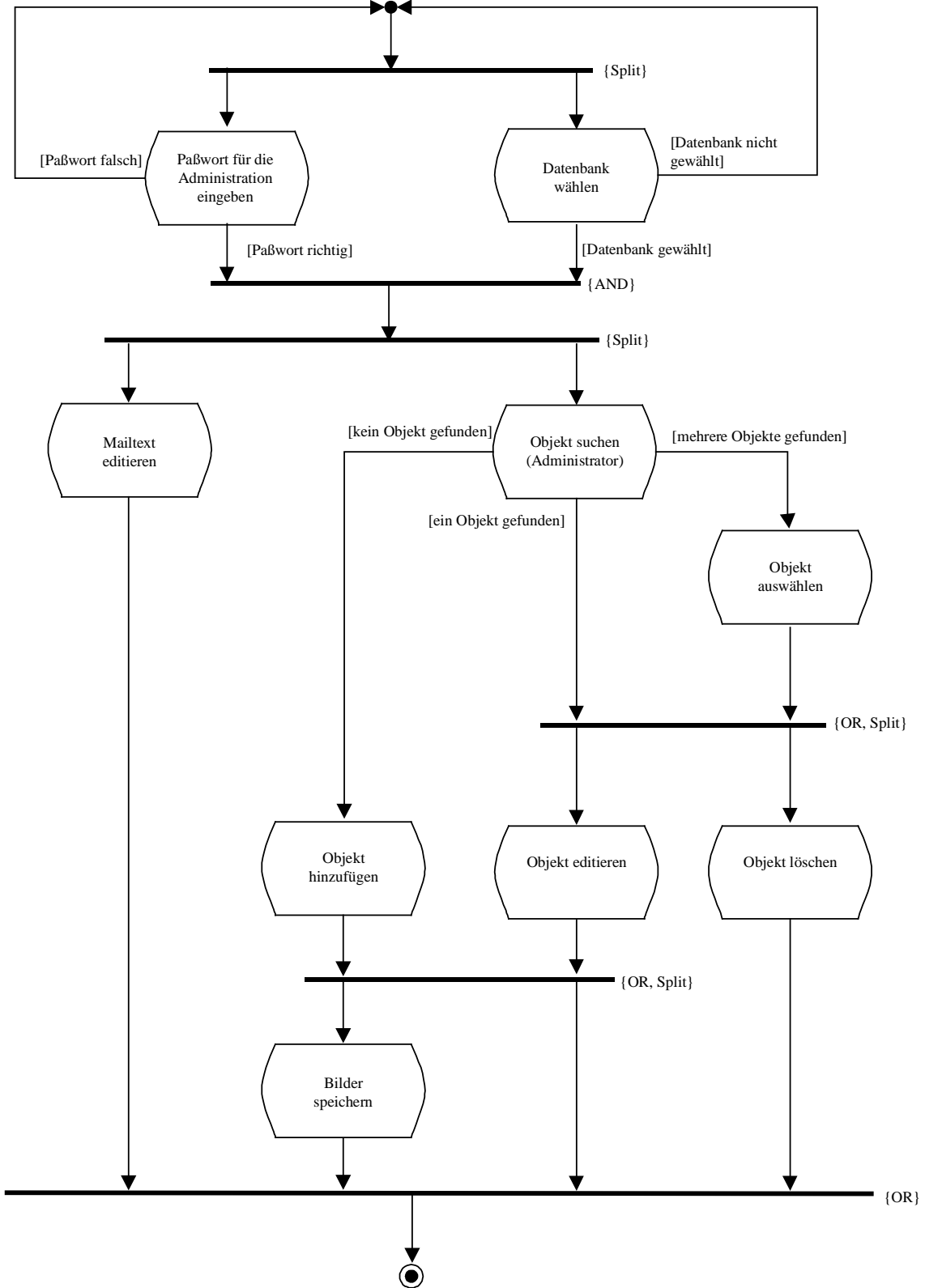


Abbildung 4: Aktivitätsdiagramm des Administrators

Anwendungsfall: Administration

Use-Case:	Paßwort für die Administration eingeben
Vorbedingung:	<ul style="list-style-type: none"> Keine
Beschreibung:	<ul style="list-style-type: none"> Die Administratorkennung wird eingegeben. Das Paßwort wird eingegeben. Das Paßwort und die Kennung werden überprüft.
Nachbedingung:	<ul style="list-style-type: none"> Das Paßwort für die Administration ist richtig.
Schnittstelle:	<ul style="list-style-type: none"> Keine

Use-Case:	Datenbank wählen
Vorbedingung:	<ul style="list-style-type: none"> Das Paßwort für die Administration ist richtig.
Beschreibung:	<ul style="list-style-type: none"> Der Administrator muß zwischen den drei Datenbanken auswählen.
Nachbedingung:	<ul style="list-style-type: none"> Eine Datenbank ist ausgewählt.
Schnittstelle:	<ul style="list-style-type: none"> Keine

Use-Case:	Mailtext Editieren
Vorbedingung:	<ul style="list-style-type: none"> Das Paßwort für die Administration ist richtig. Der Mailtext ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> Der Mailtext wird zum Editieren angezeigt. Der Administrator kann den Text editieren. Danach wird der Mailtext auf Knopfdruck abgespeichert.
Nachbedingung:	<ul style="list-style-type: none"> Der neue Mailtext ist gespeichert.
Schnittstelle:	<ul style="list-style-type: none"> Keine

Use-Case:	Objekt auswählen
Vorbedingung:	<ul style="list-style-type: none"> Es wurden mehrere Objekte anhand der eingegebenen Objektidentität gefunden. Das Paßwort für die Administration ist richtig.
Beschreibung:	<ul style="list-style-type: none"> Es wird eine Liste der gefundenen Objekte zur Auswahl ausgegeben. Nun wählt der Administrator ein Objekt aus. Das selektierte Objekt wird an den <i>Use-Case Objekt editieren</i> oder <i>Objekt löschen</i> übergeben.
Nachbedingung:	<ul style="list-style-type: none"> Ein Objekt ist selektiert
Schnittstelle:	<ul style="list-style-type: none"> Objekt Suchen

Use-Case:	Objekt suchen (Administrator)
Vorbedingung:	<ul style="list-style-type: none">• Das Paßwort für die Administration ist richtig.• Ganze oder teilweise Angaben der Objektidentität des Objektes, welches geladen werden soll.• Die Datenbank muß ausgewählt sein.
Beschreibung:	<ul style="list-style-type: none">• Nun wird ein Objekt mit passender oder ähnlicher Objektidentität in der ausgewählten Datenbank gesucht.
Nachbedingung:	<ul style="list-style-type: none">• Falls ein Objekt gefunden wird, wird das Ergebnis an den <i>Use-Case Objekt editieren</i> oder <i>Objekt löschen</i> übergeben.• Wenn mehrere Objekte gefunden werden, wird das Ergebnis an den <i>Use-Case Objekt auswählen</i> übergeben.• Andernfalls ist kein Objekt gefunden worden, somit wird ein neues Objekt mit dieser Objektidentität im <i>Use-Case Objekt hinzufügen</i> angelegt.
Schnittstelle:	<ul style="list-style-type: none">• Objekt auswählen• Objekt bearbeiten

Anwendungsfall: Objekt bearbeiten

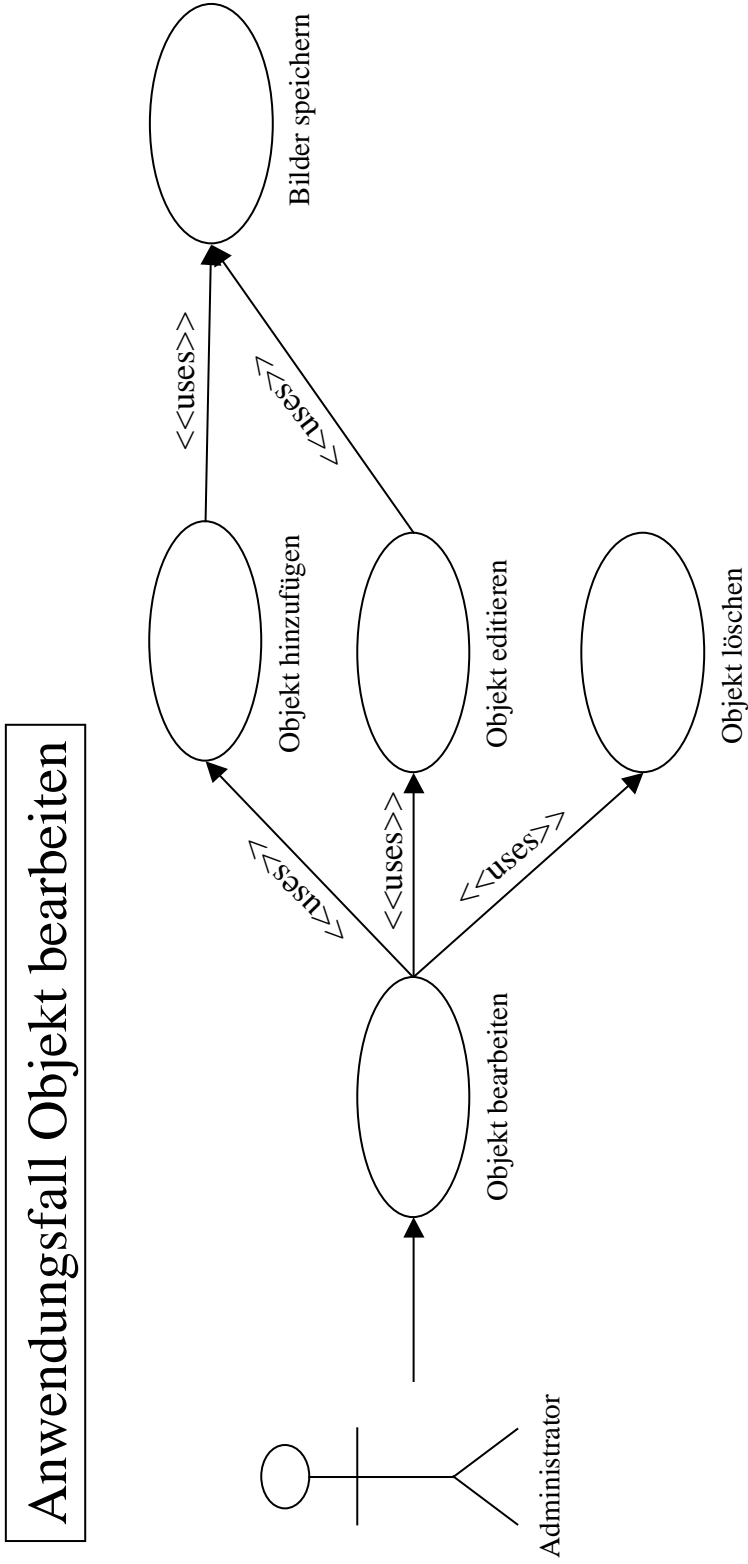


Abbildung 5: Anwendungsfall Objekt bearbeiten des Administrators

Use-Case:	Objekt editieren
Vorbedingung:	<ul style="list-style-type: none"> • Ein Objekt ist selektiert. • Das Paßwort für die Administration ist richtig.
Beschreibung:	<ul style="list-style-type: none"> • Das Objekt wird am Bildschirm zum Editieren angezeigt. • Der Administrator kann das Objekt nun bearbeiten. • Er muß für die Änderung das Paßwort zum Objekt eingeben. • Wenn das Paßwort richtig ist werden die neuen Daten eingetragen.
Nachbedingung:	<ul style="list-style-type: none"> • Das Paßwort muß überprüft und danach das Objekt eingetragen werden. • Wenn es außerdem Bilder zum Objekt gibt, wird die Objektidentität an den <i>Use-Case Bilder speichern</i> weiter gegeben
Schnittstelle:	<ul style="list-style-type: none"> • Bilder speichern • Objekt bearbeiten

Use-Case:	Objekt löschen
Vorbedingung:	<ul style="list-style-type: none"> • Ein Objekt ist selektiert. • Das Paßwort für die Administration ist richtig. • Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> • Das Objekt wird am Bildschirm angezeigt. • Der Administrator muß das Paßwort eingeben, damit er das Objekt löschen kann. • Wenn das Paßwort richtig ist, wird das Objekt gelöscht.
Nachbedingung:	<ul style="list-style-type: none"> • Das Paßwort muß überprüft und danach das Objekt gelöscht werden.
Schnittstelle:	<ul style="list-style-type: none"> • Objekt bearbeiten

Use-Case:	Objekt hinzufügen
Vorbedingung:	<ul style="list-style-type: none"> • Eine eindeutige Objekt ID, die noch nicht existiert. • Das Paßwort für die Administration ist richtig. • Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> • Das Eingabeformular wird am Bildschirm angezeigt. • Der Administrator gibt nun die Daten ein. • Er muß zum Objekt ein Paßwort eingegeben. • Die neuen Daten werden eingetragen.
Nachbedingung:	<ul style="list-style-type: none"> • Wenn Bilder zum Objekt vorhanden sind, wird die Objekt-ID an den <i>Use-Case Bilder speichern</i> weiter gegeben.
Schnittstelle:	<ul style="list-style-type: none"> • Bilder speichern • Objekt bearbeiten

Use-Case:	Bilder speichern
Vorbedingung:	<ul style="list-style-type: none">• Ein Objekt ist selektiert.• Das Paßwort für die Administration ist richtig.• Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none">• Der Administrator muß den Pfad des Bildes angeben.• Das Bild zum Objekt wird abgespeichert .
Nachbedingung:	<ul style="list-style-type: none">• Falls weitere Bilder zum Objekt vorhanden sind, erneutes Aufrufen des <i>Use-Case Bilder speichern</i>.
Schnittstelle:	<ul style="list-style-type: none">• Bilder speichern• Objekt editieren• Objekt hinzufügen

Aktor: Benutzer

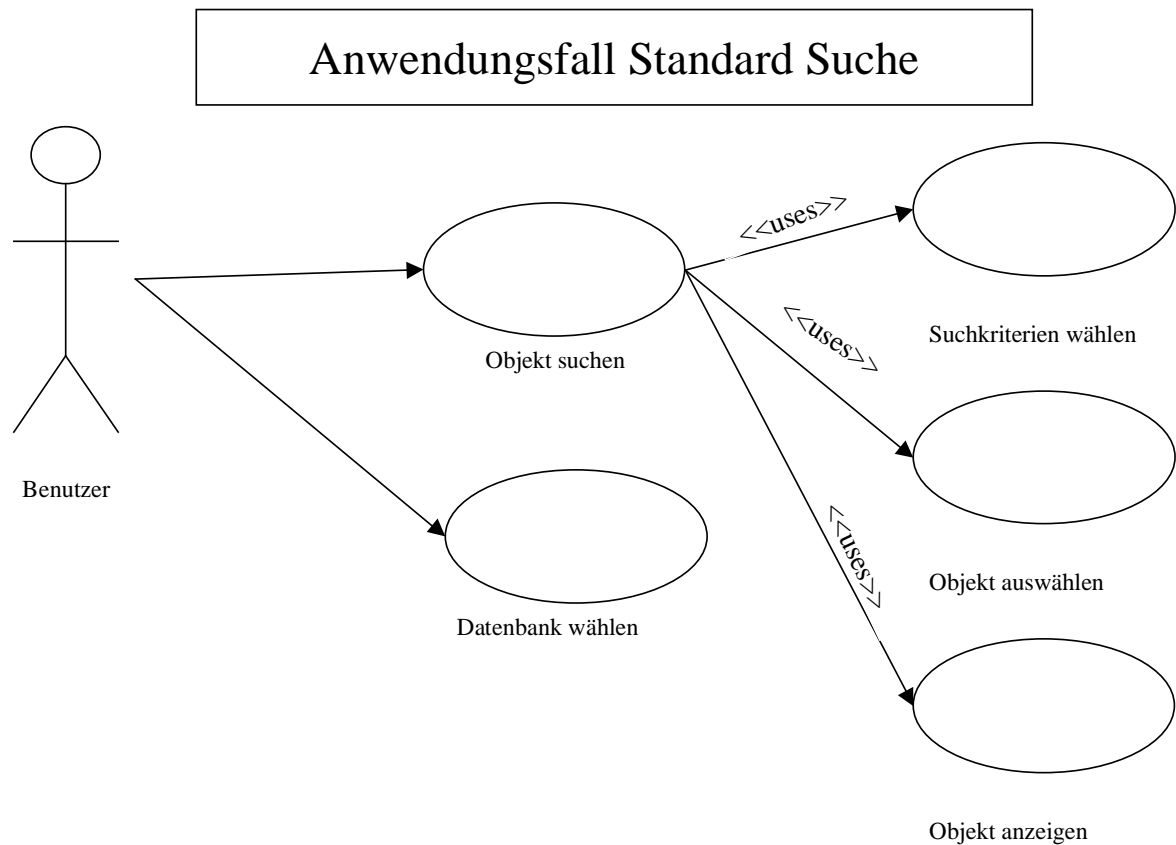


Abbildung 6: Der Anwendungsfall Standard Suche des (nicht registrierten) Benutzers

Anwendungsfall: Standard Suche

Use-Case:	Datenbank wählen
Vorbedingung:	<ul style="list-style-type: none"> Keine
Beschreibung:	<ul style="list-style-type: none"> Der Benutzer muß zwischen den drei Datenbanken auswählen.
Nachbedingung:	<ul style="list-style-type: none"> Eine Datenbank ist ausgewählt.
Schnittstelle:	<ul style="list-style-type: none"> Keine.

Use-Case:	Objekt suchen
Vorbedingung:	<ul style="list-style-type: none"> Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> Die Suchkriterien werden zur Auswahl angezeigt. Der Benutzer wählt die Suchkriterien aus (<i>Use-Case Suchkriterien wählen</i>). Danach wird in der Datenbank anhand der Kriterien gesucht.
Nachbedingung:	<ul style="list-style-type: none"> Falls ein Objekt gefunden wird, wird das Ergebnis an den <i>Use-Case Objekt anzeigen</i> übergeben. Wenn mehrere Objekte gefunden werden, wird das Ergebnis an den <i>Use-Case Objekt auswählen</i> übergeben. Andernfalls ist kein Objekt gefunden worden. Der <i>Use-Case Objekt Suchen</i> wird erneut aufgerufen.
Schnittstelle:	<ul style="list-style-type: none"> Objekt auswählen Objekt anzeigen Suchkriterien wählen

Use-Case:	Suchkriterien wählen
Vorbedingung:	<ul style="list-style-type: none"> Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> Die einzelnen Kategorien mit ihren Arten werden an dem Bildschirm angezeigt. Selektionsfelder für die verschiedenen Wertebereiche werden angezeigt. Der Benutzer wählt die Suchkriterien aus.
Nachbedingung:	<ul style="list-style-type: none"> Die Suchkriterien sind gewählt.
Schnittstelle:	<ul style="list-style-type: none"> Objekt suchen

Use-Case:	Objekt anzeigen
Vorbedingung:	<ul style="list-style-type: none"> Ein Objekt ist selektiert.
Beschreibung:	<ul style="list-style-type: none"> Das Objekt wird am Bildschirm angezeigt.
Nachbedingung:	<ul style="list-style-type: none"> Objekt suchen. Objekt auswählen.

Use-Case:	Objekt auswählen
Vorbedingung:	<ul style="list-style-type: none">• Es wurden mehrere Objekte anhand der gewählten Suchkriterien gefunden.
Beschreibung:	<ul style="list-style-type: none">• Es wird eine Liste der gefundenen Objekte zur Auswahl ausgegeben.• Nun wählt der Benutzer ein Objekt aus.• Das selektierte Objekt wird an den <i>Use-Case Objekt anzeigen</i> übergeben.
Nachbedingung:	<ul style="list-style-type: none">• Ein Objekt ist selektiert.
Schnittstelle:	<ul style="list-style-type: none">• Objekt suchen.• Objekt anzeigen.

Aktor: Registrierter Benutzer

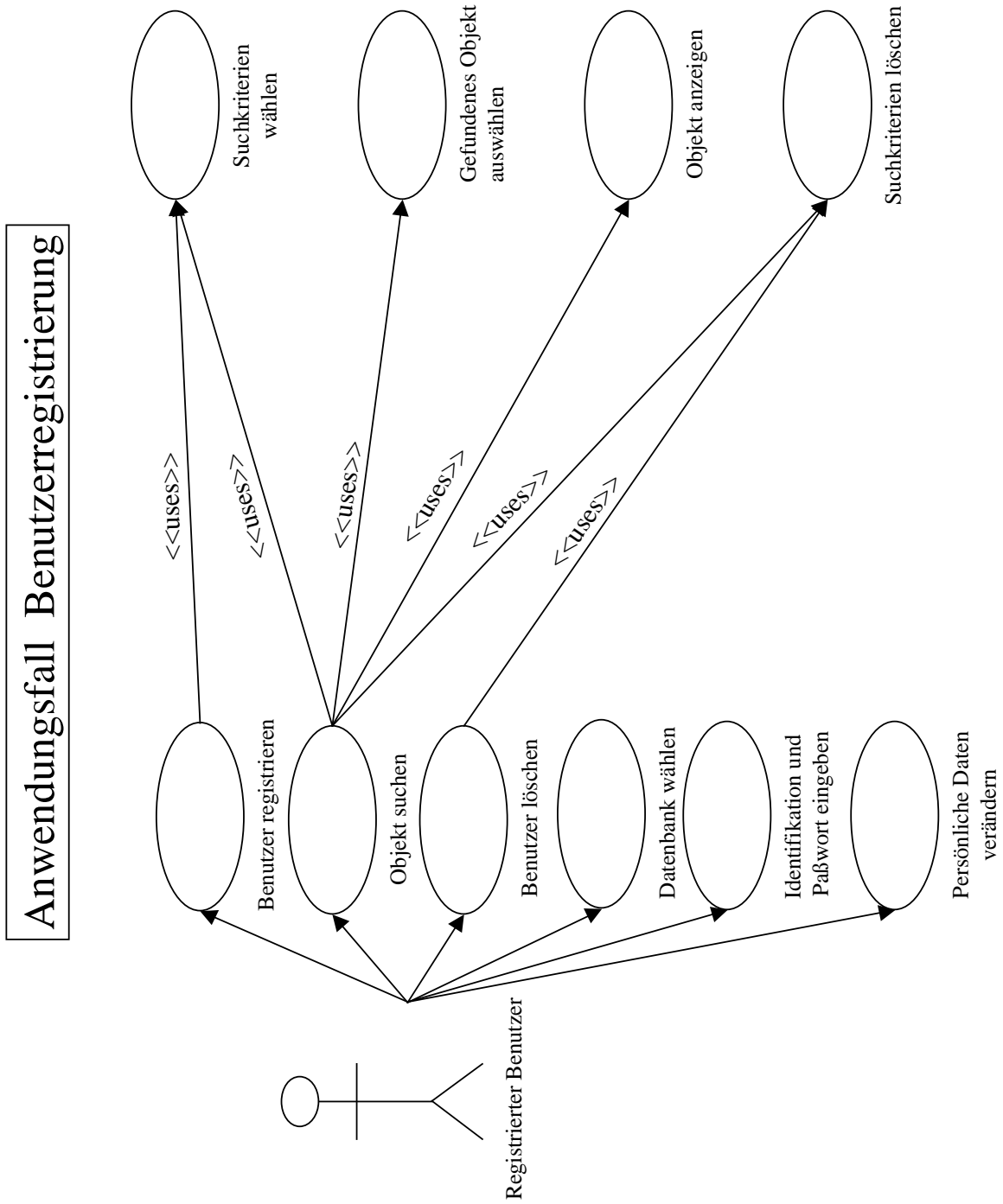


Abbildung 7: Anwendungsfall Benutzerregistrierung des registrierten Benutzers

Aktivitätsdiagramm für den registrierten Benutzer

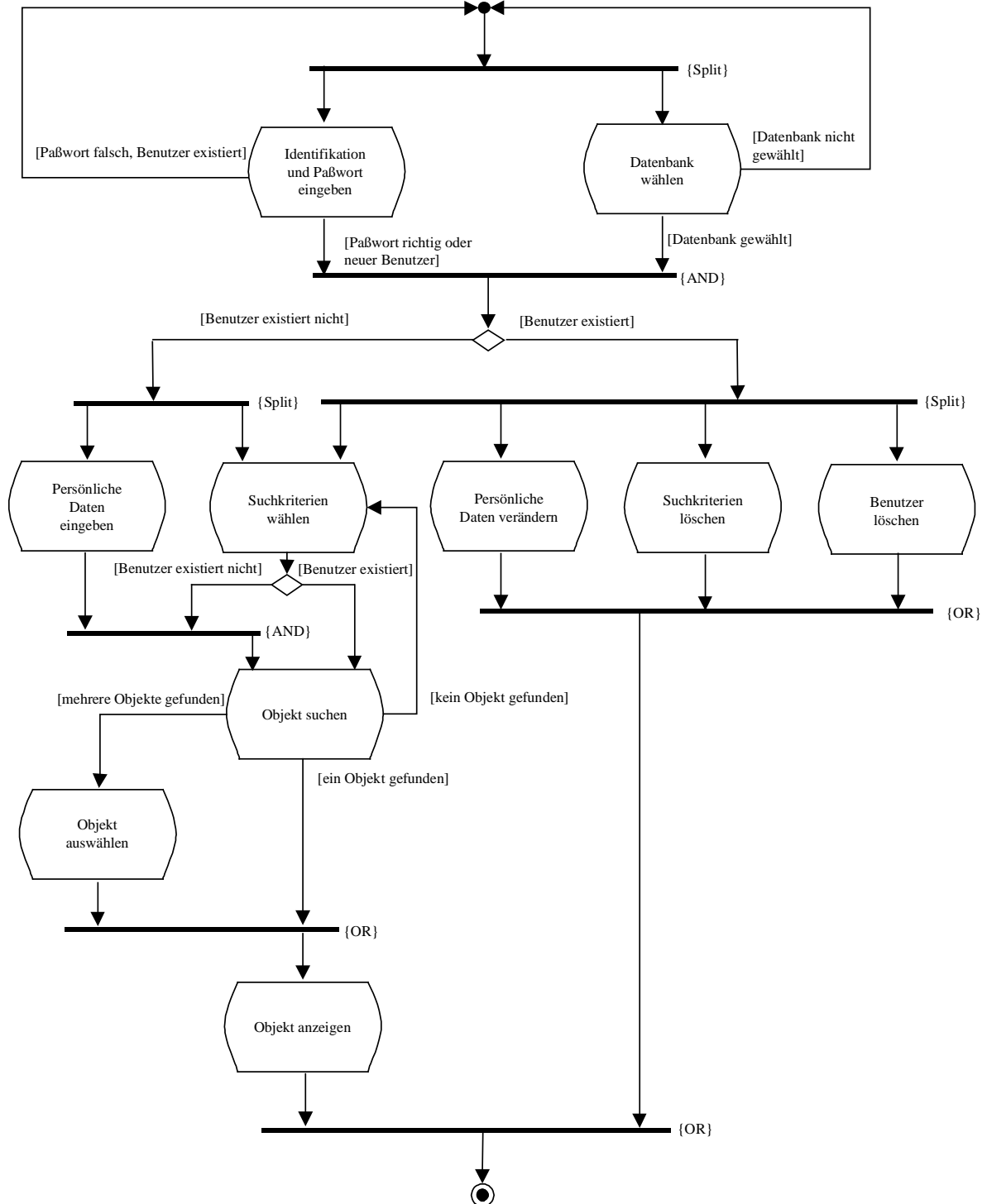


Abbildung 8: Aktivitätsdiagramm des registrierten Benutzers

Anwendungsfall: Benutzerregistrierung

Use-Case:	Benutzer registrieren
Vorbedingung:	<ul style="list-style-type: none"> • Der Benutzer hat seine ID eingegeben. • Der Benutzer hat sein Paßwort eingegeben. • Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> • Ein Eingabeformular für die persönlichen Daten wird ausgegeben (<i>Use-Case Persönliche Daten eingeben</i>). • Die Suchkriterien werden zur Auswahl angezeigt. Der Benutzer wählt die Suchkriterien aus (<i>Use-Case Suchkriterien wählen</i>). • Nachdem der Benutzer seine persönlichen Daten eingegeben und die Suchkriterien ausgewählt hat, wird beides abgespeichert.
Nachbedingung:	<ul style="list-style-type: none"> • Die Suchkriterien werden an den <i>Use-Case Objekt suchen</i> übergeben.
Schnittstelle:	<ul style="list-style-type: none"> • Objekt suchen. • Suchkriterien wählen.

Use-Case:	Identifikation und Paßwort eingeben
Vorbedingung:	<ul style="list-style-type: none"> • Keine.
Beschreibung:	<ul style="list-style-type: none"> • Der Benutzer bekommt je ein Eingabefeld für seine Identifikation und sein Paßwort. • Nach der Eingabe wird beides überprüft.
Nachbedingung:	<ul style="list-style-type: none"> • Falls die ID und das Paßwort stimmen, wird die ID an den <i>Use-Case Objekt suchen</i> übergeben. • Falls die ID noch nicht existiert, wird sie an den <i>Use-Case Benutzer registrieren</i> übergeben. • Fall ID und Paßwort nicht übereinstimmen, wird der <i>Use-Case Identifikation und Paßwort eingeben</i> erneut aufgerufen.
Schnittstelle:	<ul style="list-style-type: none"> • Objekt suchen. • Identifikation und Paßwort eingeben. • Benutzer registrieren

Use-Case:	Datenbank wählen
Vorbedingung:	<ul style="list-style-type: none"> • Keine
Beschreibung:	<ul style="list-style-type: none"> • Der Benutzer muß zwischen den drei Datenbanken auswählen.
Nachbedingung:	<ul style="list-style-type: none"> • Eine Datenbank ist ausgewählt.
Schnittstelle:	<ul style="list-style-type: none"> • Keine.

Use-Case:	Persönliche Daten eingeben
Vorbedingung:	<ul style="list-style-type: none"> • Der Benutzer hat seine ID und sein Paßwort eingegeben.
Beschreibung:	<ul style="list-style-type: none"> • Es wird ein Eingabeformular für die persönlichen Daten angezeigt. • Der Benutzer gibt nun die Daten ein. • Die Benutzerdaten werden auf Vollständigkeit geprüft.
Nachbedingung:	<ul style="list-style-type: none"> • Die Benutzerdaten sind eingegeben.
Schnittstelle:	<ul style="list-style-type: none"> • Keine.

Use-Case:	Persönliche Daten verändern
Vorbedingung:	<ul style="list-style-type: none"> • Der Benutzer hat seine ID eingegeben. • Der Benutzer hat sein Paßwort eingegeben.
Beschreibung:	<ul style="list-style-type: none"> • Es werden die persönlichen Daten des Benutzers am Bildschirm angezeigt. • Der Benutzer kann die Daten jetzt verändern. • Die persönlichen Daten werden auf Vollständigkeit geprüft. • Danach werden die persönlichen Daten abgespeichert.
Nachbedingung:	<ul style="list-style-type: none"> • Alle persönlichen Daten sind eingegeben.
Schnittstelle:	<ul style="list-style-type: none"> • Keine.

Use-Case:	Objekt suchen
Vorbedingung:	<ul style="list-style-type: none"> • Der Benutzer hat seine ID eingegeben. • Der Benutzer hat sein Paßwort eingegeben. • Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> • Falls die Suchkriterien nicht schon gewählt sind, werden die Suchkriterien des Benutzers geladen. Der Benutzer kann nun die Suchkriterien verändern (<i>Use-Case Suchkriterien wählen</i>). • Nun hat der Benutzer die Auswahl, ob er mit den veränderten Suchkriterien nur suchen will oder ob er sie auch gleichzeitig abspeichern will. • Danach wird anhand der gewählten Suchkriterien in der gewählten Datenbank gesucht. • Außerdem hat der Benutzer hier die Möglichkeit, die Suchkriterien für die gewählte Datenbank zu löschen.
Nachbedingung:	<ul style="list-style-type: none"> • Falls der Benutzer seine Suchkriterien löschen will, werden die Benutzer ID und der Datenbankname an den <i>Use-Case Suchkriterien löschen</i> übergeben. • Falls ein Objekt gefunden wird, wird das Ergebnis an den <i>Use-Case Objekt anzeigen</i> übergeben. • Wenn mehrere Objekte gefunden werden, wird das Ergebnis an den <i>Use-Case Objekt auswählen</i> übergeben. • Andernfalls ist kein Objekt gefunden worden. Der <i>Use-Case Objekt suchen</i> wird erneut aufgerufen.
Schnittstelle:	<ul style="list-style-type: none"> • Suchkriterien wählen • Suchkriterien löschen • Objekt anzeigen • Objekt auswählen • Objekt suchen

Use-Case:	Benutzer löschen
Vorbedingung:	<ul style="list-style-type: none"> • Der Benutzer hat seine ID eingegeben. • Der Benutzer hat sein Paßwort eingegeben.
Beschreibung:	<ul style="list-style-type: none"> • Der Benutzer wird aufgefordert, zu bestätigen, daß er die Registrierung löschen möchte. • Nach der Bestätigung werden alle Suchkriterien gelöscht (<i>Use-Case Suchkriterien löschen</i>). • Danach werden die Benutzerdaten gelöscht und der Benutzer ausgeloggt.
Nachbedingung:	<ul style="list-style-type: none"> • Es existieren keinerlei benutzerbezogene Daten mehr.
Schnittstelle:	<ul style="list-style-type: none"> • Suchkriterien löschen

Use-Case:	Suchkriterien löschen
Vorbedingung:	<ul style="list-style-type: none"> • Der Benutzer hat seine ID eingegeben. • Der Benutzer hat sein Paßwort eingegeben.
Beschreibung:	<ul style="list-style-type: none"> • Falls eine Datenbank gewählt ist, die Suchkriterien dieser Datenbank löschen. • Falls keine Datenbank gewählt ist, alle Suchkriterien löschen.
Nachbedingung:	<ul style="list-style-type: none"> • Die Suchkriterien einer oder aller Datenbanken sind gelöscht
Schnittstelle:	<ul style="list-style-type: none"> • Benutzer löschen. • Objekt suchen.

Use-Case:	Suchkriterien wählen
Vorbedingung:	<ul style="list-style-type: none"> • Der Benutzer hat seine ID eingegeben. • Der Benutzer hat sein Paßwort eingegeben. • Eine Datenbank ist ausgewählt.
Beschreibung:	<ul style="list-style-type: none"> • Falls der Benutzer schon persönliche Suchkriterien für die gewählte Datenbank hat, werden diese geladen. • Falls der Benutzer keine persönlichen Suchkriterien für die gewählte Datenbank hat, werden die Standard Suchkriterien geladen. • Die einzelnen Kategorien mit ihren Arten werden an dem Bildschirm angezeigt. • Selektionsfelder für die verschiedenen Wertebereiche werden angezeigt. • Der Benutzer wählt die Suchkriterien aus.
Nachbedingung:	<ul style="list-style-type: none"> • Die Suchkriterien werden zurückgegeben.
Schnittstelle:	<ul style="list-style-type: none"> • Objekt suchen. • Benutzer registrieren.

Aktor: System

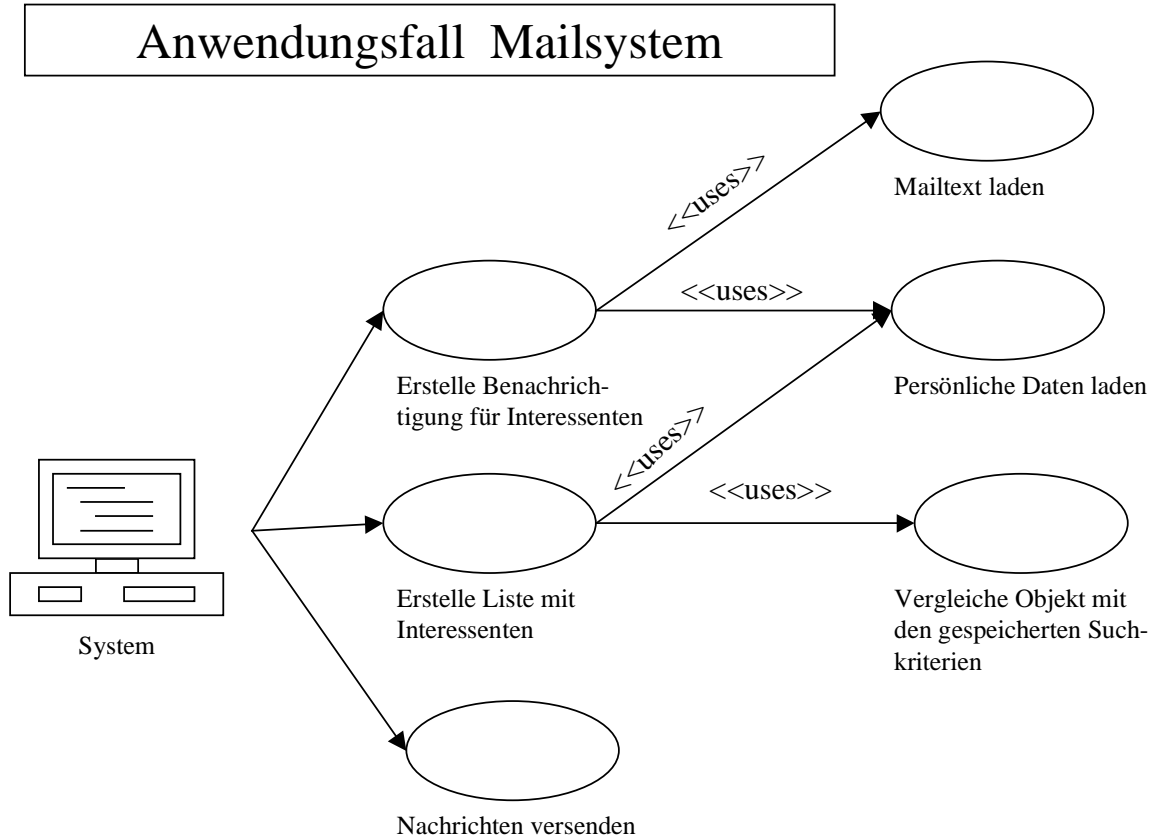


Abbildung 9: Anwendungsfall Mailsystem; Akteur: System

Anwendungsfall: Mailsystem

Use-Case:	Erstelle Benachrichtigung für Interessenten
Vorbedingung:	<ul style="list-style-type: none"> • Ein Objekt in einer der drei Datenbanken wurde geändert oder hinzugefügt. • Name des Administrators der das Objekt eingegeben hat.
Beschreibung:	<ul style="list-style-type: none"> • Der Mailtext des Administrators wird geladen (<i>Use-Case Lade Mailtext</i>). • Die Liste mit Interessenten wird vom System übergeben. • Anhand der Liste werden die Email Adressen der einzelnen Personen geladen (<i>Use-Case Persönliche Daten laden</i>). • Die fertigen Mails werden an das System übergeben.
Nachbedingung:	<ul style="list-style-type: none"> • Alle Mails für die Interessenten sind erstellt.
Schnittstelle:	<ul style="list-style-type: none"> • Mailtext laden. • Persönliche Daten laden.

Use-Case:	Persönliche Daten laden
Vorbedingung:	<ul style="list-style-type: none"> • Die ID des Benutzers • Die Art der gewünschten Informationen
Beschreibung:	<ul style="list-style-type: none"> • Die Benutzer Daten werden geladen. • Die gewünschten Informationen werden übergeben.
Nachbedingung:	<ul style="list-style-type: none"> • Die Informationen sind übergeben.
Schnittstelle:	<ul style="list-style-type: none"> • Erstelle Benachrichtigung für Interessenten. • Erstelle Liste mit Interessenten.

Use-Case:	Mailtext laden
Vorbedingung:	<ul style="list-style-type: none"> • Name des Administrators • Name der Datenbank, der das Objekt angehört.
Beschreibung:	<ul style="list-style-type: none"> • Anhand des Namens des Administrators und der Datenbank wird der entsprechende Mailtext geladen.
Nachbedingung:	<ul style="list-style-type: none"> • Der Mailtext wird zurückgegeben.
Schnittstelle:	<ul style="list-style-type: none"> • Erstelle Benachrichtigung für Interessenten.

Use-Case:	Erstelle Liste mit Interessenten
Vorbedingung:	<ul style="list-style-type: none"> • Ein Objekt in einer der drei Datenbanken wurde geändert oder hinzugefügt. • Name der Datenbank, der das Objekt angehört.
Beschreibung:	<ul style="list-style-type: none"> • Vergleiche die Daten des Objekts mit den Suchkriterien der registrierten Benutzer (<i>Use-Case Vergleiche Objekt mit den gespeicherten Suchkriterien</i>). • Für jede gefundene Benutzer ID die persönlichen Daten laden, die für die Liste benötigt werden (<i>Use-Case Persönliche Daten laden</i>). • Den Benutzer zu der Liste hinzufügen. • Die Liste an das System zurückgeben.
Nachbedingung:	<ul style="list-style-type: none"> • Rückgabe der Liste mit Interessenten.
Schnittstelle:	<ul style="list-style-type: none"> • Persönliche Daten laden. • Vergleiche Objekt mit den gespeicherten Suchkriterien.

Use-Case:	Nachricht versenden
Vorbedingung:	<ul style="list-style-type: none"> • Der zu versendende Text wurde übergeben. • Die Email Adresse wurde übergeben.
Beschreibung:	<ul style="list-style-type: none"> • Die Mail wird erstellt und versendet.
Nachbedingung:	<ul style="list-style-type: none"> • Die Mail ist versendet.
Schnittstelle:	<ul style="list-style-type: none"> • Keine

Use-Case:	Vergleiche Objekt mit den gespeicherten Suchkriterien
Vorbedingung:	<ul style="list-style-type: none"> • Name der Datenbank, der das Objekt angehört.
Beschreibung:	<ul style="list-style-type: none"> • Die Objektdaten werden mit den Suchkriterien der registrierten Benutzer verglichen. • Bei Übereinstimmung wird jede Benutzer Identität zurückgegeben.
Nachbedingung:	<ul style="list-style-type: none"> • Liste mit gefundenen Benutzer Identitäten wird zurückgegeben.
Schnittstelle:	<ul style="list-style-type: none"> • Erstelle Liste mit Interessenten.

4.3 Anforderung an die Präsentation

Die fertige Anwendung soll bei verschiedenen Sparkassen und Banken verwendet werden. Sie soll in dem bestehenden Web-Auftritt eines Instituts leicht integrierbar sein.

Derzeit wird bei den bestehen Webanwendungen die Präsentation durch Server Side Include Dateien bestimmt, das heißt der Webserver verändert automatisch das Erscheinungsbild der Anwendung. Somit werden zum Beispiel bei einer Sparkasse der mosaikartige Hintergrund, die Kopfleisten oben, die Navigationsleiste unten, die Schriften usw. automatisch angepaßt oder eingefügt. Dieses Verfahren wird auch bei dieser Anwendung eingesetzt.

5 Design

5.1 Allgemeine Beschreibung zum Design

Für den Designvorgang wird die in [Koch & Mandel,1999] beschriebene Notation verwendet. Diese Methode besteht aus drei Phasen: Conceptual Design, Navigational Design und Presentational Design. Im Conceptual Design werden die Datenobjekte und Ihre Beziehungen zueinander erfaßt. Das Ergebnis des Conceptual Designs ist ein Domänenmodell.

Das Navigationsmodell beschreibt zwei Sichtweisen auf das Domänenmodell: Das Navigationsklassenmodell und das Navigationsstrukturmodell. Das Navigationsklassenmodell zeigt, welche Objekte man durch Navigation erreichen kann. Das Navigationsstrukturmodell beschreibt wie man die Objekte erreichen kann.

Das Präsentationsmodell definiert die Darstellung der Navigationsobjekte. Das Präsentationsmodell teilt sich in zwei Teilmodelle: Statisches und dynamisches Präsentationsmodell. Für jedes Navigationsobjekt gibt es mindestens ein Präsentationsobjekt, das die Darstellung beschreibt. Das dynamische Präsentationsmodell gibt dann an, wie sich ein solches Objekt verhält.

5.2 Domänenmodell

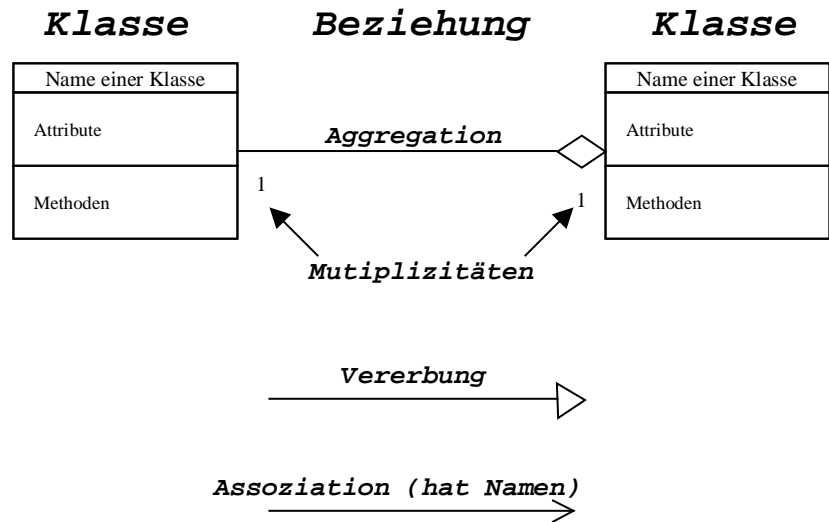
5.2.1 Allgemeine Beschreibung des Domänenmodells

Im Domänenmodell werden die Datenobjekte erfaßt, die für die Applikation relevant sind. Danach werden die Beziehungen zwischen den Objekten modelliert. Somit entsteht ein Modell, das die komplette Applikation, d.h. die einzelnen Teilbereiche aus den Use-Cases, zusammenfaßt.

Das Modell wird in UML-Notation dargestellt. Die einzelnen Klassen werden noch separat beschrieben.

5.2.2 Anmerkung zum Domänenmodell

Symbolerklärungen:



Bemerkung:

Das Domänenmodell besteht aus folgenden drei Subsystemen: Benutzerprofile, Immobilien und Verwaltung. Da die einzelnen Subsysteme sehr klein sind, wurden sie zu einem Modell zusammen gefaßt: Die Benutzerprofile sind für benutzerspezifische Informationen zuständig. Das Subsystem der Immobilien beschreibt die Datenstruktur der drei verschiedenen Arten von Immobilienobjekten und der Bereich Verwaltung schildert den administrativen Teil der Anwendung.

5.2.3 Das Domänenmodelldiagramm

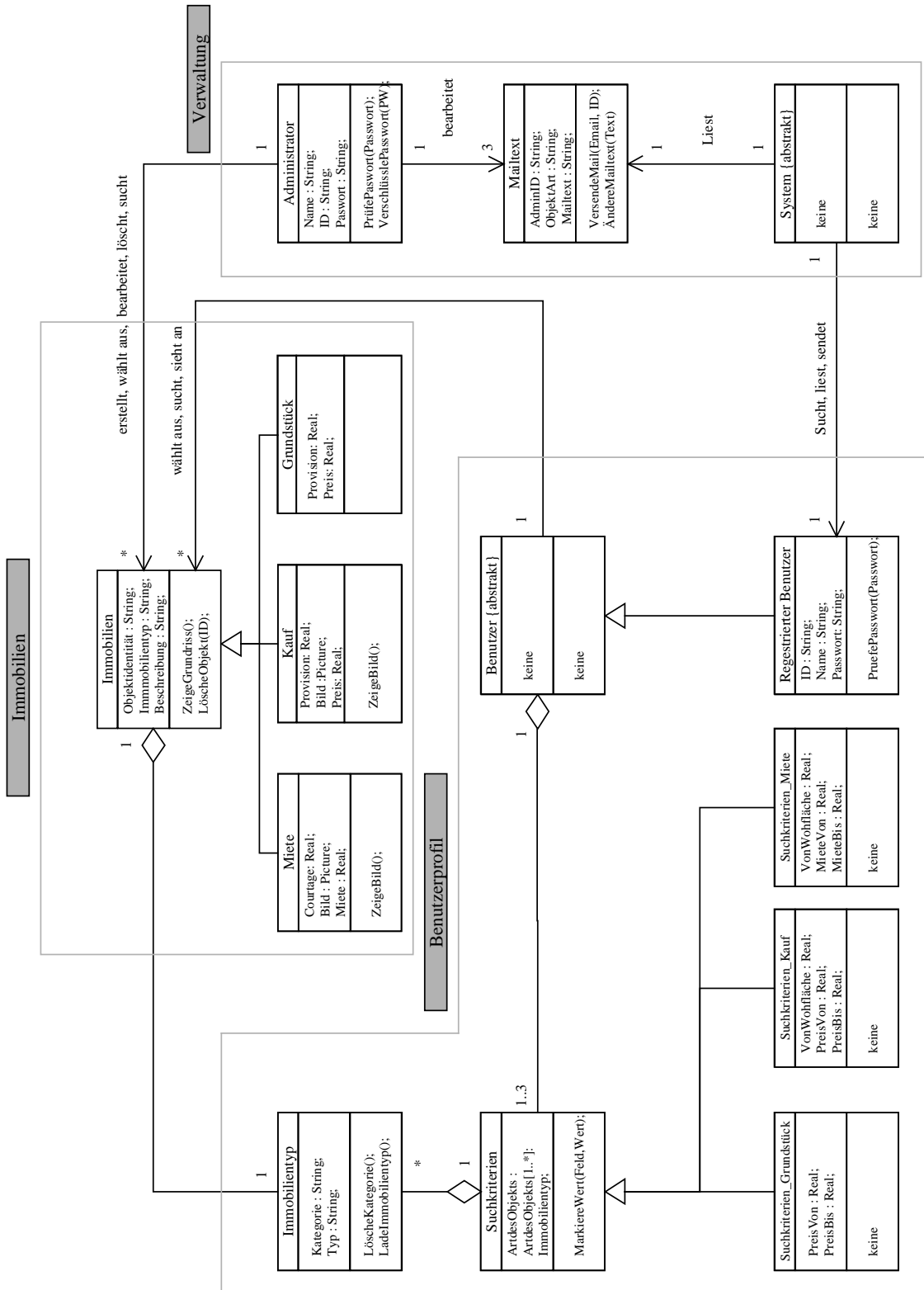


Abbildung 10: Das Domänenmodell

5.2.4 Beschreibung der einzelnen Klassen

Klassenname:	Benutzer{ abstrakt }		
Attribute:	-		
Methode:	-		
Beziehungen:	Name	Art	Klasse
	Sucht	Assoziation	Objekte
	wählt aus	Assoziation	Objekte
	sieht an	Assoziation	Objekte
	-	Aggregation	Suchkriterien
Vererbungsstruktur:	Obertyp von Registrierter Benutzer		
Beschreibung:	Diese abstrakte Klasse ist für den nicht registrierten Benutzer. Er hat nur die Standard Suchkriterien zur Auswahl.		

Klassenname:	Registrierter Benutzer		
Attribute:	ID : String; Name : String; Vorname : String; Straße : String; Ort: String; PLZ: Real; Paßwort : String; Telefon : String; Fax : String;		
Methode: Art	PruefePasswort(Passwort), ZeigeBenutzerdatenAn(), VerschlüsselePasswort(ID,Passwort)		
Beziehungen:	Name	Art	Klasse
	-	-	-
Vererbungsstruktur:	Untertyp von Benutzer		
Beschreibung:	Diese Klasse ist für die Registrierung und die Speicherung der Suchkriterien eines Benutzer zuständig.		

Klassenname:	Suchkriterien		
Attribute:	ArtdesObjekts[1..*] : Immobilientyp; GrundflächeVon : Real; GrundflächeBis : Real; Postleitzahl : Integer; Ort : String; Markierung[1..n] : String;		
Methode:	MarkiereWert(Feld,Wert), LöscheMarkierung(Feld,Wert), BildeAuswahlmenue(),		
Beziehungen:	Name	Art	Klasse
	-	Aggregation	Registrierte Benutzer
	-	Aggregation	Immobilientyp
Vererbungsstruktur:	Obertyp von Suchkriterien_Miete, Suchkriterien_Kauf		
Beschreibung:	Mit dieser Klasse wird festgelegt, welche Suchkriterien zugelassen sind.		

Klassenname:	Suchkriterien_Kauf		
Attribute:	VonWohfläche : Real; BisWohfläche : Real; PreisVon : Real; PreisBis : Real;		
Methode:			
Beziehungen:	Name	Art	Klasse
	-	-	-
Vererbungsstruktur:	Untertyp von Suchkriterien		
Beschreibung:	-		

Klassenname:	Suchkriterien_Grundstück		
Attribute:	PreisVon : Real; PreisBis : Real;		
Methode:			
Beziehungen:	Name	Art	Klasse
	-	-	-
Vererbungsstruktur:	Untertyp von Suchkriterien		
Beschreibung:	-		

Klassenname:	Suchkriterien_Miete		
Attribute:	VonWohfläche : Real; BisWohfläche : Real; MieteVon : Real; MieteBis : Real;		
Methode:			
Beziehungen:	Name	Art	Klasse
	-	-	-
Vererbungsstruktur:	Untertyp von Suchkriterien		
Beschreibung:	-		

Klassenname:	Immobilientyp		
Attribute:	Kategorie : String; Typ : String;		
Methode:	SetzeKategorie(Wert), SetzeTyp(Typ), LeseWerte(), LöscheKategorie, ErstelleListeMitKategorien(), GeneriereTypenEinerKategorie(), LadeImmobilientyp();		
Beziehungen:	Name	Art	Klasse
	-	Aggregation	Suchkriterien
	-	Aggregation	Objekt
Vererbungsstruktur:	Keine		
Beschreibung:	Diese Klasse legt fest, welcher Wohnungstyp in welche Kategorie kommt.		

Klassenname:	Immobilie		
Attribute:	ID: String; Besitzer : String, Freigegeben: Boolean; ArtDesObjekts : Immobilientyp; Grundfläche : String; Postleitzahl : Integer; Ort : String; Beschreibung : String; Grundriß : Image; Verkehrsanbindung : String; Lage : String; Ansprechpartner : String; Emailadresse : String; Telefonnummer : String; Faxnummer : String		
Methode:	PrüfePasswort(Passwort), LöscheObjekt(ID), VerschlüssePasswort(Passwort),		
Beziehungen:	Name	Art	Klasse
	-	Aggregation	Immobilientyp
	erstellt	Assoziation	Administrator
	sucht	Assoziation	Administrator
	wählt aus	Assoziation	Administrator
	bearbeitet	Assoziation	Administrator
	sucht	Assoziation	Benutzer
	wählt aus	Assoziation	Benutzer
	sieht an	Assoziation	Benutzer
Vererbungsstruktur:	Obertyp von Kauf, Miete, Grundstück		
Beschreibung:	In dieser Klasse werden die zu speichernden Attribute eines Immobilienobjekts festgelegt.		

Klassenname:	Kauf		
Attribute:	Aufzug : String, Stockwerk : Integer, Parkplatz : String, Zimmeranzahl : Real; Preis : Real; Provision : Real; Bild : Image		
Methode:	ZeigeBild();		
Beziehungen:	Name	Art	Klasse
	-	-	-
Vererbungsstruktur:	Untertyp von Immobilie		
Beschreibung:	-		

Klassenname:	Grundstück		
Attribute:	Preis : Real; Provision : Real		
Methode:	-		
Beziehungen:	Name	Art	Klasse
	-	-	-
Vererbungsstruktur:	Untertyp von Immobilie		
Beschreibung:	-		

Klassenname:	Miete		
Attribute:	Raucher : String; Haustiere : String; Miete : Real; Courtage: Real; Bild : Image;		
Methode:	ZeigeBild();		
Beziehungen:	Name	Art	Klasse
	-	-	-
Vererbungsstruktur:	Untertyp von Immobilie		
Beschreibung:	-		

Klassenname:	Administrator		
Attribute:	Name : String; Vorname : String; ID : String; Email : String; Passwort : String		
Methode:	PrüfePasswort(Passwort), VerschlüssePasswort(Passwort), SetzeWert(Feld,Wert)		
Beziehungen:	Name	Art	Klasse
	bearbeitet	Assoziation	Mailtext
	erstellt	Assoziation	Immobilie
	wählt aus	Assoziation	Immobilie
	bearbeitet	Assoziation	Immobilie
	sucht	Assoziation	Immobilie
Vererbungsstruktur:	Keine		
Beschreibung:	Hier werden die einzelnen Administratoren verwaltet.		

Klassenname:	Mailtext		
Attribute:	AdminID : String; Objekt_Kategorie : String; Mailtext : String;		
Methode:	ZeigeMailtextAn(), VersendeMail(Email,ID), ÄndereMailtext(Text)		
Beziehungen:	Name	Art	Klasse
	bearbeitet	Assoziation	Administrator
Vererbungsstruktur:	Keine		
Beschreibung:	Jeder Administrator kann einen Mailtext für jede der drei Datenbanken abspeichern und bearbeiten. Dieser Mailtext wird dann automatisch versendet, sobald ein Benutzer Interesse an einem Immobilienobjekt haben könnte.		

5.3 Navigationsmodell

5.3.1 Allgemeine Beschreibung des Navigationsmodells

Das Navigationsmodell stellt grafisch dar, wie man in der fertigen Anwendung navigieren wird. Im Klassenmodell werden die wichtigsten Navigationsknoten, die sich aus dem Domänenmodell ableiten lassen, beschrieben. Im Navigationsstrukturmodell wird das Klassenmodell verfeinert. Hierfür werden verschiedenen Kontexte und die Art, wie man auf Datenobjekte zugreift, geschildert.

5.3.2 Navigationsklassenmodell

Beim Navigationsmodell werden nur noch die Klassen vom Domänenmodell aufgeführt, die für die Navigation relevant sind. Es gibt nur noch Assoziationen zwischen den Klassen. Attribute und Methoden können aus anderen Klassen abgeleitet werden, sie werden durch “/” gekennzeichnet.

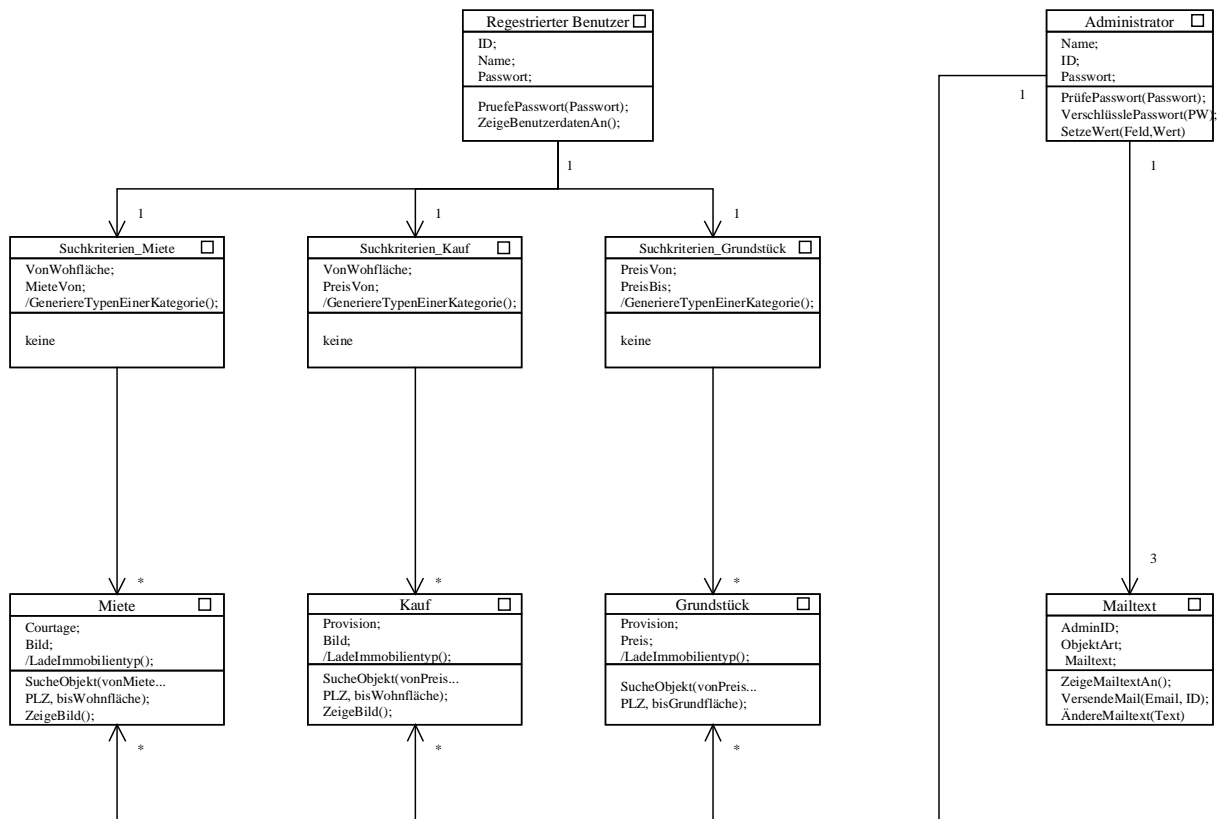


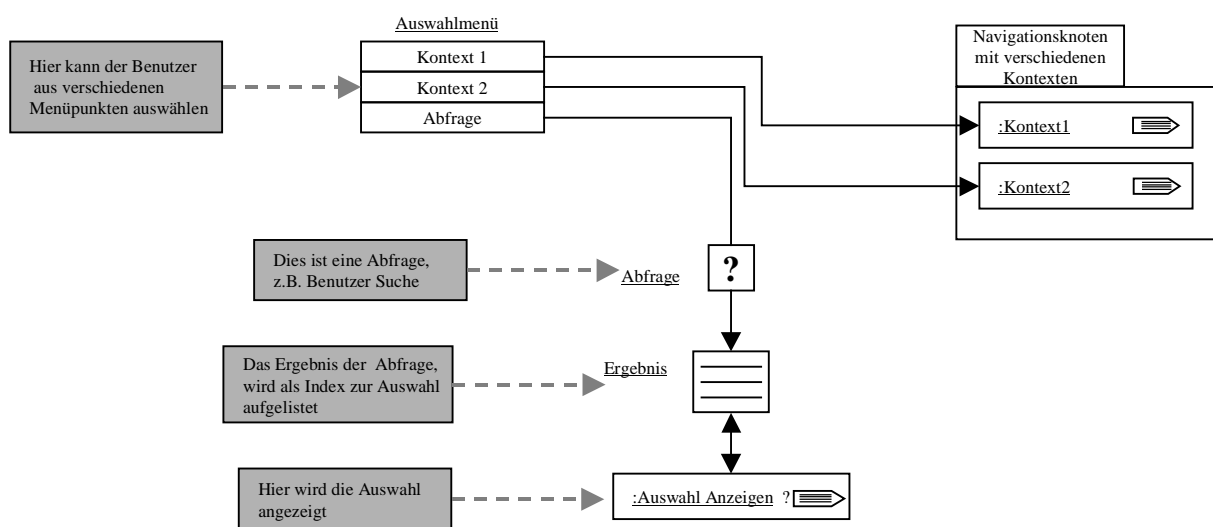
Abbildung 11: Das Navigationsklassenmodell

5.3.3 Navigationsstrukturmodell

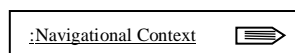
5.3.4 Bemerkung zum Navigationsstrukturmodell

Bei dieser Anwendung „verändert“ sich die Navigationsstruktur, d.h. die Anwendung erkennt ob ein Benutzer registriert ist oder nicht und paßt die Navigationsstruktur dementsprechend an. Aus diesem Grund ist die Navigationsstruktur am Anfang dreigeteilt, um die einzelnen Fälle zu unterscheiden. Die drei Diagramme werden am Ende dieses Abschnittes zusammengefaßt (Abbildung 15).

5.3.5 Erklärung der Notation

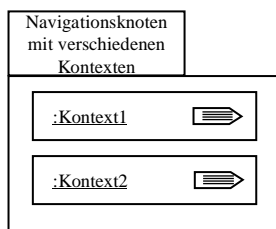


Navigationskontext (Navigational Context):



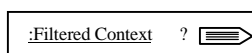
Navigationskontext kann durch eine oder mehrere Sequenz(en) vom Navigationsknoten erreicht werden. Dadurch ist es möglich, durch unterschiedliche Navigation den selben Inhalt zu erreichen. Der Inhalt eines Navigationskontextes besteht aus den Attributen einer Navigationsklasse.

Zusammengefaßte Navigationskontexte (Package of Navigational Contexts):



Die verschiedenen Navigationskontexte einer Navigationsklasse werden zur Übersichtlichkeit zu Packages zusammengefaßt.

Gefilterte Kontexte (Filtered Context):

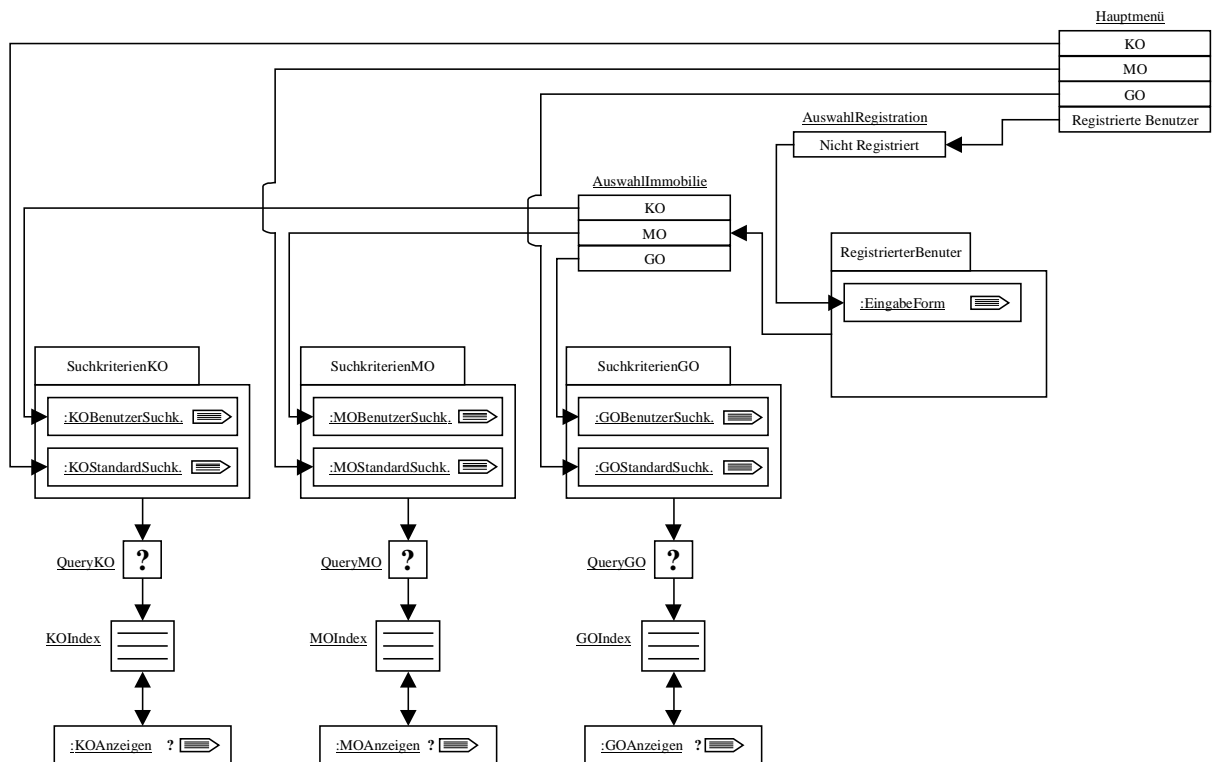


Der gefilterte Kontext ermöglicht es, den Inhalt des Kontextes dynamisch anzupassen. Es werden nur Elemente angezeigt, die ein bestimmtes Kriterium erfüllen.

Die ersten zwei Fälle beschreiben die Navigation des registrierten Benutzers (Abbildung 12) und die Navigation des Benutzers der sich nicht registrieren möchte (Abbildung 13). Der nächste Fall ist die Administration. Dieser wurde zur bessern Übersichtlichkeit aufgegliedert (Abbildung 14).

Fall: Der nicht registrierte Benutzer

Das System erkennt automatisch, ob der Benutzer sich registriert hat oder nicht (ob die Benutzererkennung schon vorhanden ist). Deswegen kommt man von dem Menüpunkt Registrierte Benutzer direkt zum Eingabeformular. Die Auswahl Registration ist im Prinzip hinfällig, da man nur einen Punkt zur Auswahl hat.

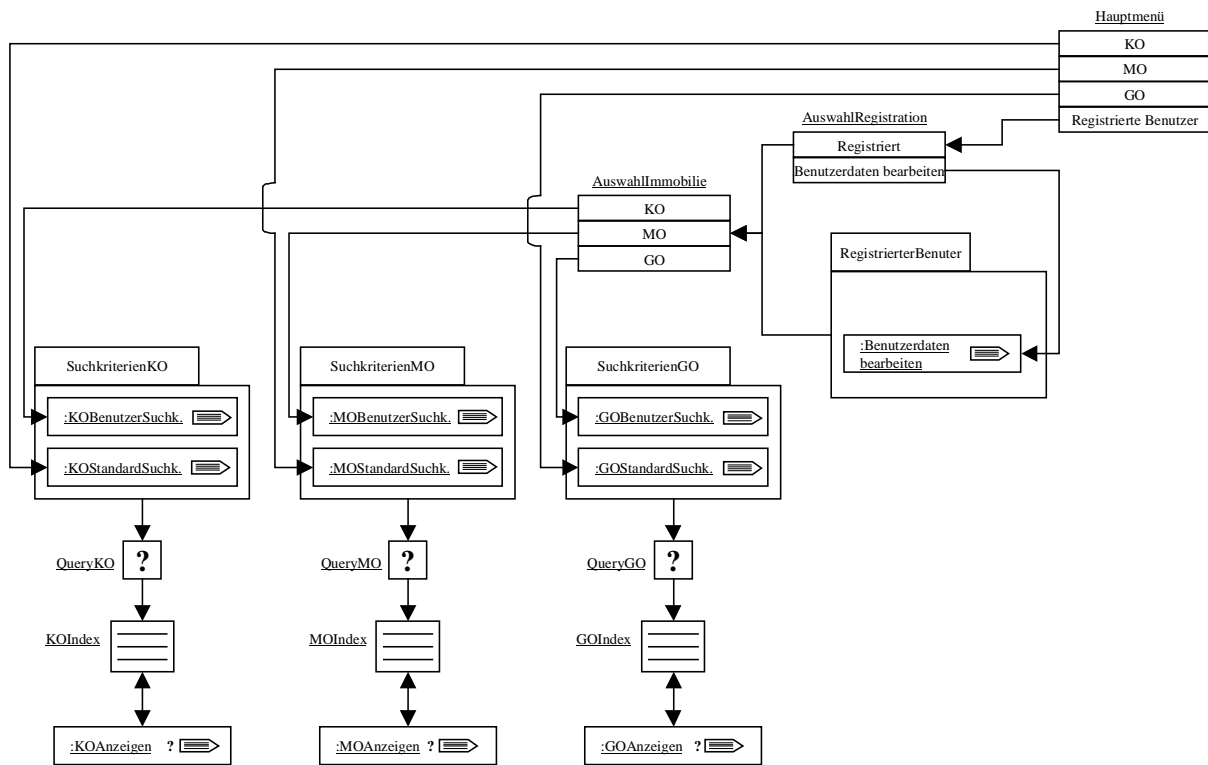


Bemerkung:
KO steht für Kaufobjekte
MO steht für Mietobjekte
GO steht für Grundstücksobjekte

Abbildung 12: Navigationsstrukturmodell; Fall des nicht registrierten Benutzers

Fall: Der registrierte Benutzer

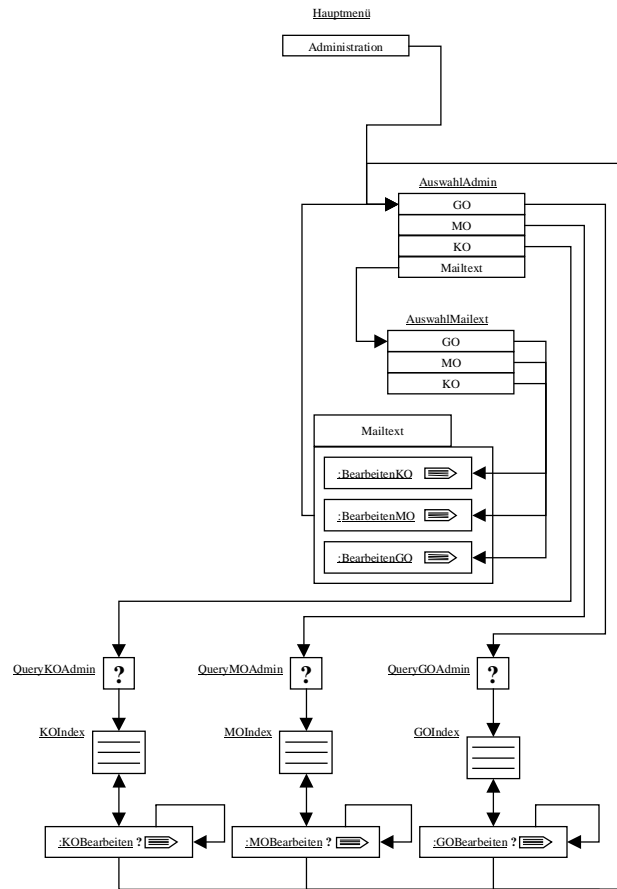
Im Menü Auswahl Registration ist das Eingabeformular jetzt weggefallen, da der Benutzer schon registriert ist. Dafür hat er jetzt aber die Möglichkeit, seine Benutzerdaten zu bearbeiten oder direkt mit der Suche weiterzumachen.



Bemerkung:
KO steht für Kaufobjekte
MO steht für Mietobjekte
GO steht für Grundstücksobjekte

Abbildung 13: Navigationsstrukturmodell; Fall des registrierten Benutzers

Fall: Administration



Bemerkung:
KO steht für Kaufobjekte
MO steht für Mietobjekte
GO steht für Grundstücksobjekte

Abbildung 14: Navigationsstrukturmodell; Fall der Administration

In Abbildung 15 sieht man das zusammengefaßte Navigationsstrukturmodell. Im Punkt MainMenue wird „unterschieden“ zwischen den einzelnen Aktoren (siehe Use-Case). Die Auswahl KO/MO/GO wird für die Standard Suche (Aktor: Benutzer) benötigt. Dem Actor Administrator steht die Auswahl Administration zur Verfügung und dem registrierten Benutzer der letzte Auswahlpunkt des Menüs. Von den Menüpunkten ausgehende Pfeile beschreiben, wie die einzelnen Akteure navigieren können und welche Auswahlmöglichkeiten ihnen zur Verfügung stehen.

Exemplarisch wird hier noch der Suchvorgang nach einem Kaufobjekt erklärt (siehe Abbildung 15). Nachdem der Benutzer den Menüpunkt KO (Kaufobjekte suchen) gewählt hat, werden ihm die Suchkriterien :KOSTandardSuchk. angezeigt. Nach dem Festlegen der Suchkriterien durch den Benutzer findet die Suche QueryKO statt. Aus der Ergebnisliste KOIndex können einzelne Objekte zum Anzeigen (KOAnzeigen) ausgewählt werden.

Das gesamte Navigationsstrukturmodell

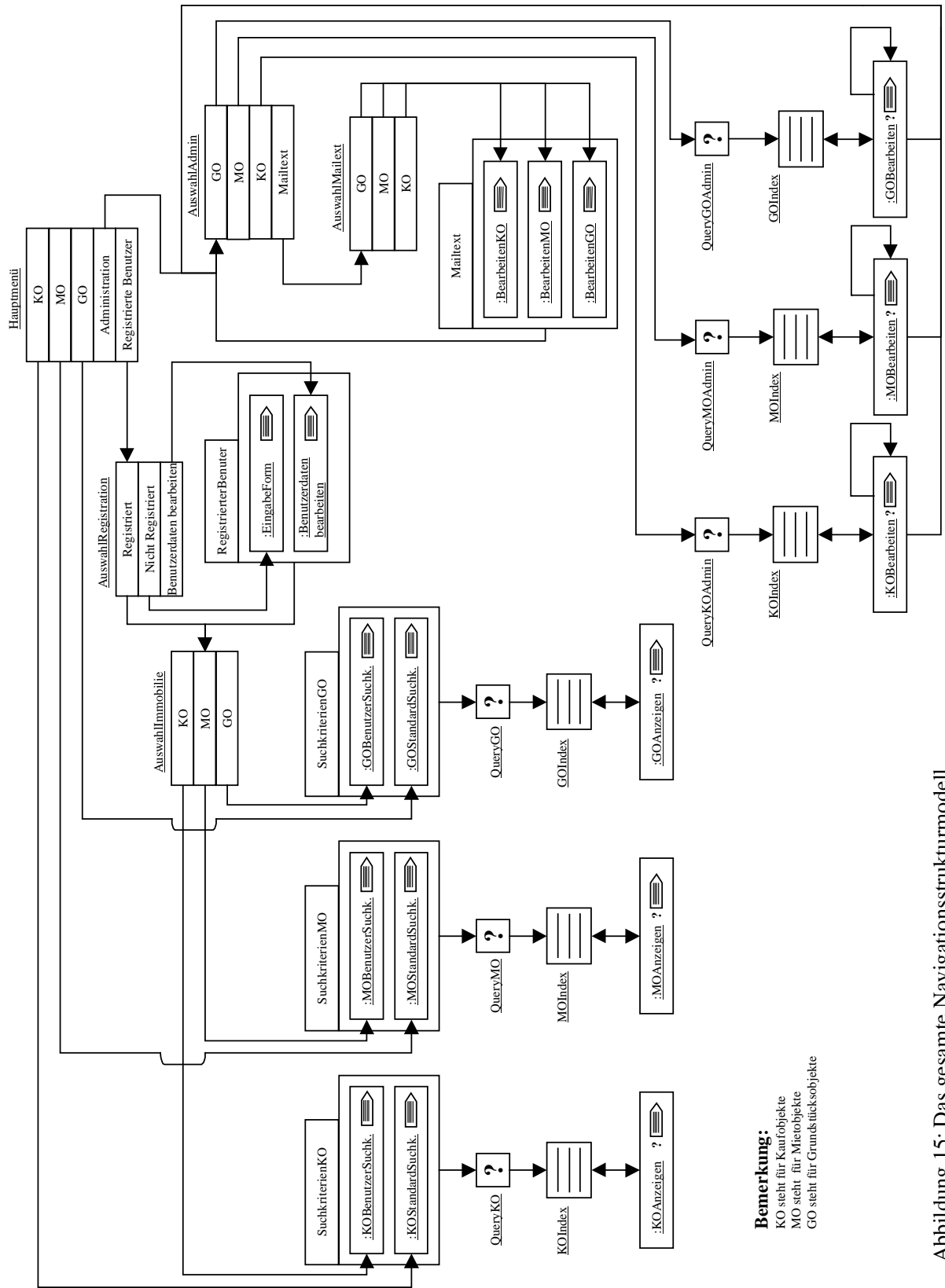


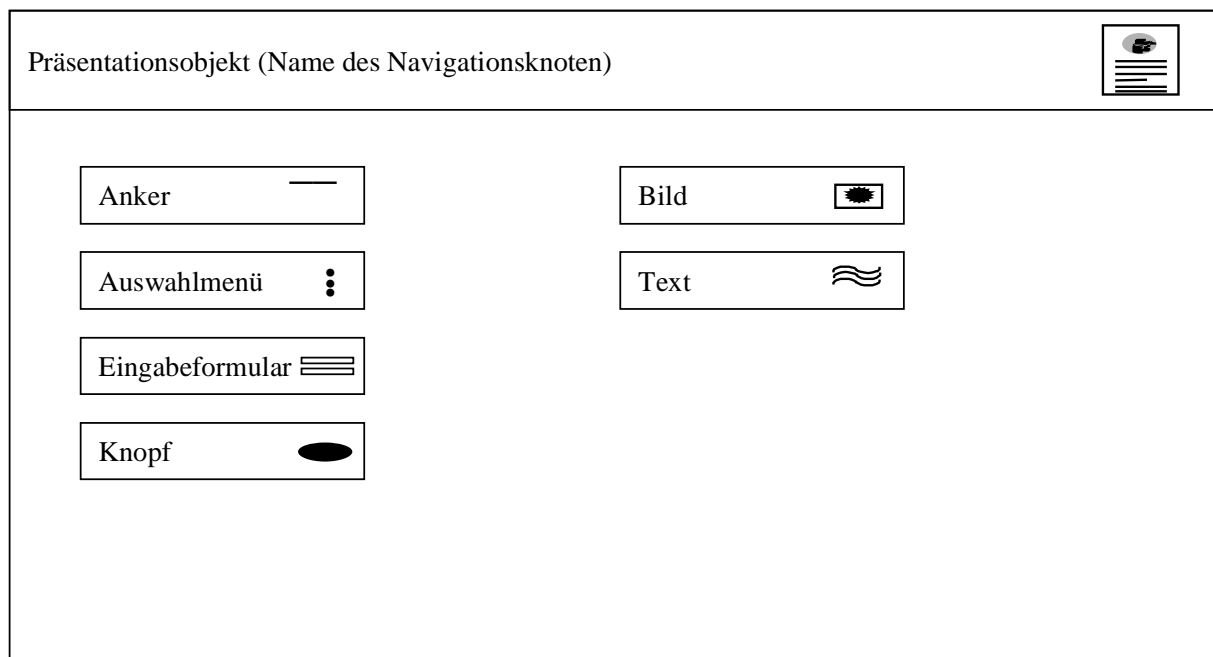
Abbildung 15: Das gesamte Navigationsstrukturmodell

5.4 Präsentationsmodell

Das Präsentationsmodell wird zum Modellieren einer abstrakten Benutzerschnittstelle verwendet. Es gliedert sich in zwei Teile, den Statischen und den Dynamischen. Mit dem statischen Präsentationsmodell modelliert man das Erscheinungsbild eines Navigationsknotens. Das dynamische Präsentationsmodell beschreibt das Verhalten einzelner Objekte auf Benutzereingaben. Ein Beispiel hierfür wäre das Abspielen eines Filmes auf Knopfdruck. Da solche interaktiven Elemente bei dieser Anwendung nicht verwendet werden, wird das dynamische Modell nicht modelliert.

5.4.1 Statisches Präsentationsmodell

Erklärung der Notation



Jedes Attribut eines Navigationsobjektes wird als Objekt im Präsentationsmodell dargestellt. Dadurch wird ein abstrakter Überblick über das Erscheinungsbild der einzelnen Navigationsknoten gegeben.

Das statische Präsentationsmodell der Anwendung

Das Präsentationsmodell wird hier nur ansatzweise gezeigt, da die meisten Präsentationsobjekte sehr ähnlich aufgebaut sind. Die Navigation beginnt mit dem Hauptmenü (Abbildung 16):

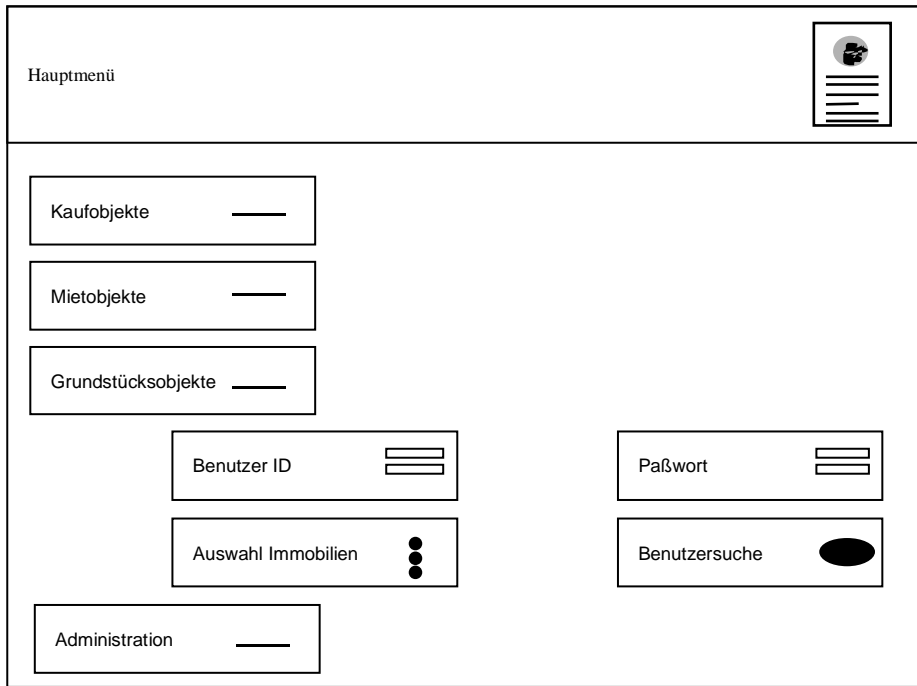


Abbildung 16: Navigationsknoten MainMenue

Von dem Hauptmenü aus kann man zum AuswahlAdmin navigieren (Abbildung 17)

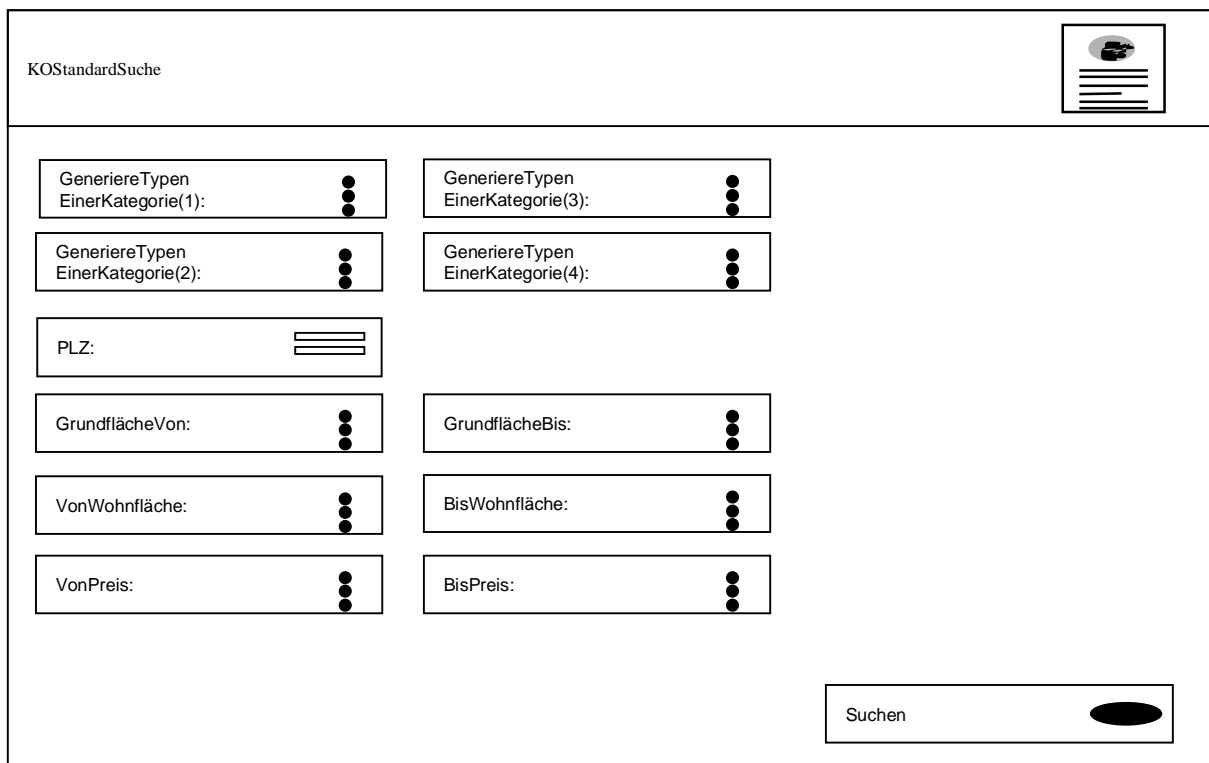


Abbildung 17: Navigationsknoten KOSTandardSuche

Außerdem kann man von dem MainMenue zur Suche von Kauf-, Miet- und Grundstücksobjekten navigieren. Hier wird nur die Suche von den Kaufobjekten gezeigt.

The image shows a screenshot of a web application interface titled "KOSTandardSuche". In the top right corner, there is a small icon of a house with a magnifying glass. Below the title, there are several search filters arranged in a grid. Each filter is a rectangular box containing text and a vertical ellipsis icon (three dots). The filters are: "GeneriereTypen EinerKategorie(1):", "GeneriereTypen EinerKategorie(2):", "GeneriereTypen EinerKategorie(3):", "GeneriereTypen EinerKategorie(4):", "PLZ:" (with a horizontal ellipsis icon), "GrundflächeVon:", "GrundflächeBis:", "VonWohnfläche:", "BisWohnfläche:", "VonPreis:", and "BisPreis:". At the bottom right of the interface, there is a button labeled "Suchen" with a black oval icon to its right.

Abbildung 18: Navigationsknoten KOSTandardSuche

t

Das Ergebnis der Suche ist eine Liste mit Objekten, aus der dann ein Objekt ausgewählt werden kann. Das Objekt wird dann angezeigt. Das Anzeigen eines Objektes und das Bearbeiten eines Objektes ist sehr ähnlich, weswegen hier nur das Anzeigen eines Kaufobjektes modelliert wird (Abbildung 19).

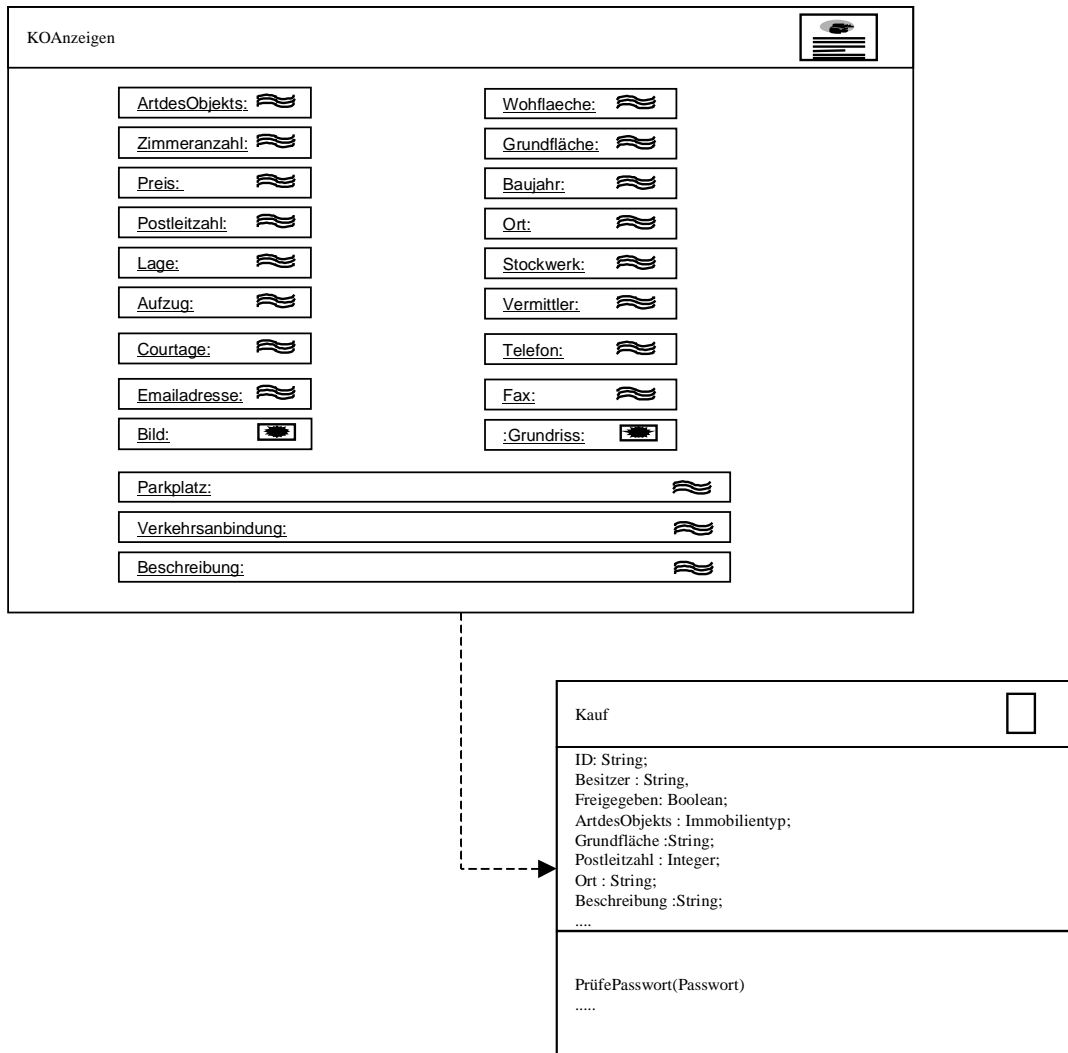


Abbildung 19: Navigationsknoten KO Anzeigen

6 Implementierung

6.1 Allgemeine Aspekte der Implementierung

Die Immobiliendatenbank wurde in der prozeduralen Programmiersprache Lite implementiert. Dies hat mehrere Gründe. Zum einen war die Verwendung dieser Programmiersprache eine Anforderung des Kunden, da bereits die alte Immobiliendatenbank und andere Produkte des Kunden in Lite programmiert wurden. Außerdem wird die Lösung von Problemen dadurch erleichtert, da man nicht auf einen Spezialisten angewiesen ist, sondern auf bereits vorhandenes Konw-How der Mitarbeiter im Umgang mit dieser Programmiersprache zurückgreifen kann. Weiterhin entstehen keine Zusatzkosten für die Beschaffung von Lizenzen, da die Progemmiersprache mit dem Datenbanksystem mitgeliefert wird.

Das Datenbanksystem wurde gewählt, da Erfahrung im Umgang mit diesem System vorhanden ist.

Die Betrachtung des Preis/Leistungsverhältnisses legt ebenfalls die Entscheidung für diese Datenbank nahe (zum Preis siehe Webseite von Huges Technology). Es wird nicht der gleiche Funktionsumfang zur Verfügung gestellt wie von Oracle oder Informix. Der Leistungsumfang ist aber durchaus ausreichend für die Immobiliendatenbank, wie die Erfahrungen mit der bisherigen Immobiliendatenbank und anderen Produkten zeigen.

In der verwendeten Immobiliendatenbank befinden sich derzeit im Schnitt 100 Datensätze. Die W3-MSQL Datenbank ist bei anderen Anwendungen des Kunden im Online Betrieb schon mit bis zu 2000 Datensätzen betrieben worden und erreichte bei entsprechenden Operationen (suchen, auflisten u.s.w) akzeptable Antwortzeiten. Die Leistungsfähigkeit der Datenbank ist somit auch für die neue Anwendung gewährleistet.

Der Import vorhandener Datensätze in die neue Datenbank wird durch die Verwendung des gleichen Systems ebenfalls erleichtert.

6.2 Probleme bei der Implementierung

Aus den allgemeinen Aspekten der Implementierung entstand nachfolgende Problematik:

Der objektorientierte Entwurf der Anwendung wird mit der prozeduralen Programmiersprache Lite und der Datenbank W3-MSQL realisiert, d.h., das Klassendiagramm mit seinen Klassen, Attributen, Methoden und Beziehungen mußte in ein prozedurales Schema umgewandelt werden.

Folgende generelle Strategie wurde angewendet:

- Klassen des Objektdiagramms wurden in Tabellen umgewandelt.
- Vererbungsstrukturen werden aufgelöst, die Attribute und Methoden des Obertypen werden in jedem Untertyp separat implementiert (keine Generalisierung mehr).
- Methoden werden als Funktionen mit Operationen auf Tabellen realisiert.
- Beziehungen der Klassen werden Beziehungen zwischen den Tabellen. Dies wird durch Verwendung gemeinsamer Primar- und Fremdschlüssel realisiert. Multiplizitäten müssen bei Einfüge- und Änderungsfunktionen berücksichtigt werden.
- Verwendung von tabellenübergreifenden Funktionen, um Aggregationen zu realisieren.

In einigen Fällen mußte von dieser Strategie abgewichen werden, da sie nur grob die Vorgehensweise beschreibt und Probleme, die sich im Detail ergeben, nicht berücksichtigt.

Ein anderes Problem bei der Realisierung war die Option des Administrators Bilder zu einem Immobilienobjekt hinzuzufügen. Das Hinaufladen der Bilder (upload) über das Internet stellt

sich als problematisch heraus, da der Datenbankserver beim Dateien hinaufladen abstürzte. Deswegen wurde das Hinaufladen der Bilder aus der Administrator Seite herausgenommen und als separate Webseite innerhalb der Administration realisiert. Der Vorgang des Hinaufladens wurde in der Programmiersprache Perl realisiert.

6.3 Zusammenfassung der Technischen Daten

- Plattform : Suse - Linux Version 5.x
- Webserver Apache 1.2
- Datenbankmanagement System : W3-MSQL
- Programmiersprachen : embedded WWW scripting language Perl und Lite

7 Wartung und Sicherheit

7.1 Wartung des Programms

Die Wartung der Immobiliendatenbank wird sowohl von der Firma, die die Anwendung betreibt, als auch von den Administratoren der einzelnen Immobilien durchgeführt. Der Administrator hat die Aufgabe, seine Immobilienobjekte zu pflegen, d.h. jegliche Änderungen an den Attributen eines Objekts nimmt der Administrator selbst vor.

Die Wartungsarbeiten der Firma sind zweigeteilt in das Bearbeiten der Immobilientypen und die Wartung des Datenbanksystems.

Das Bearbeiten der Instanzen der Klasse der Immobilientypen ist mit dem Bearbeiten der Tabelle Immobilientyp gleichzusetzen, da Klassen als Tabellen realisiert wurden. Das Hinzufügen, Löschen und Bearbeiten von Datensätzen dieser Tabelle wird manuell erledigt, da dies sehr selten vorkommt (Erfahrungswert von der alten Immobiliendatenbank) und der Aufwand, dafür eine eigene Benutzeroberfläche zu programmieren, nicht gerechtfertigt war. Die Wartung des Datenbanksystems wird nur dann durchgeführt, wenn von der Firma eine Fehlerbehebung bereitgestellt wird, die das Programm betrifft.

7.2 Systemrecovery

Die Vorgehensweise beim Wiederherstellen eines abgestürzten Systems ist abhängig davon, wie das System abgestürzt ist und welche Art der Sicherung verwendet wird. Bei der Immobiliendatenbank wird ein dreistufiges Backup verwendet:

- Das Protokollieren von Änderungen auf den Tabellen
- Das tägliche Erstellen einer Sicherungskopie von den Protokollen auf ein Backupmedium
- Die wöchentliche Komplettsicherung auf ein Backupmedium

Beim Protokollieren von Änderungen auf den Tabellen wird jede Änderung als SQL (Simply Query Language) Statement in eine Textdatei abgespeichert. Die SQL Statements ermöglichen es, die einzelnen Änderungen sehr schnell und einfach zu reproduzieren.

Die nächsten zwei Stufen des Backups werden von einer Backupsoftware automatisch ausgeführt. Bei der täglichen Sicherung wird das Backup des Vortages mit dem aktuellen Stand der Textdatei verglichen und die Differenz gespeichert. Die Komplettsicherung der Datenbank wird einmal in der Woche ausgeführt. Dazu ist es nötig, die Datenbank komplett in eine Textdatei abzuspeichern. Dies ist mit Hilfe des Befehls `msqldump <Datenbankname>`

möglich. Der Befehl wird einmal in der Woche automatisch ausgeführt, bevor das Backupprogramm mit der wöchentlichen Sicherung beginnt.

Bei einem Systemversagen ist es auf jeden Fall möglich, den Stand des Vortages wiederherzustellen. Als erstes muß man auf dem Server das Datenbanksystem installieren, danach stellt man den Stand der letzten Komplettsicherung wieder her. Danach spielt man die Backup Kopien der einzelnen Tage wieder ein. Durch die Backup Kopien der einzelnen Tage, kann man, mit Hilfe der SQL Statements, die Operationen auf den Daten jedes einzelnen Tages durchführen. Ob man den Zustand des aktuellen Tages wiederherstellen kann, ist davon abhängig, ob die Textdatei, in der die Änderungen mit protokolliert sind, beschädigt ist oder nicht. Falls sie nicht beschädigt ist, kann man den aktuellen Stand des Tages wiederherstellen.

Eine andere Methode die Daten zu sichern, wäre die tägliche Komplettsicherung. Diese Art der Sicherung hat den entscheidenden Nachteil, daß man nur den Datenstand eines Tages und nicht die einzelnen Änderungen wiederherstellen kann. Somit wäre die Möglichkeit einzelne Änderungsoperationen zu verändern oder zu löschen, genommen.

8 Erweiterbarkeit

Bei der neuen Immobiliendatenbank muß der Benutzer für jedes Objekt eine Mail oder ein Fax an den Administrator schicken, um ihm sein Interesse mitzuteilen. Eine Möglichkeit, die Immobiliendatenbank für den Benutzer noch komfortabler zu gestalten, ist die bestehende Anwendung um ein Warenkorbsystem zu erweitern. Dies gibt dem Anwender die Option jedes Immobilienobjekt, für das er sich interessiert, dem Warenkorb hinzuzufügen. Sobald er alle für ihn interessanten Objekte hinzugefügt hat, müßte er nur noch **eine** Mitteilung versenden, in der dann alle Objekte enthalten sind.

9 Zusammenfassung

Die Immobiliendatenbank war das erste Projekt, daß ich mit Hilfe von Design Methoden entwickelt habe. Da ich Erfahrungen mit ähnlichen Projekten habe, die ohne Design Methoden realisiert wurden, war das ein sehr interessanter Vergleich für mich.

Der Zeitaufwand der für das Design nötig ist, ist sehr hoch. Das relativiert sich aber wieder, da sich der Zeitaufwand für die Implementierung drastisch verkürzt, da nachfolgende Situationen nicht mehr auftreten:

- Mißverständnisse zwischen Anwender und Entwickler. Dies ist durch die einzelnen Phasen der Entwicklung ausgeschlossen, da dem Anwender bei jeder Phase des Designs Informationsmaterial gezeigt wird, das man auch fachfremden Personen verständlich machen kann.
- Die Anwendung hat nicht den gewünschten Funktionsumfang und muß nachträglich erweitert werden. Dies wird durch die Anforderungsanalyse verhindert.
- Der Anwender ist mit der Bedienbarkeit der Anwendung nicht zufrieden. Der Fall kann nicht eintreten, da vor der Implementierung festgelegt wird, wie der Anwender die Anwendung zu bedienen hat. Es werden also nachträglichen Änderungen minimiert.

Die oben genannten Situationen treten bei einer Implementierung ohne Design Methoden immer wieder auf. Die Realisierung wird dadurch oftmals unnötig komplex, da die Änderungen im Konzept nicht berücksichtigt waren und komplette Module neu geschrieben oder existierende Module angepaßt werden müssen. Die nachträglichen Änderungen führen oftmals auch zu unstrukturierten Quelltexten, die dann sehr schwer verständlich sind.

Mein Fazit ist, daß Entwickeln mit Design Methoden erleichtert das Erstellen von Anwendungen und der Entwicklungsprozeß an sich läuft strukturierter ab.

10 Literaturverzeichnis

- Daniel Schwabe (schwabe@inf.puc-rio.br), The Object-Oriented Hypermedia Design Model,
<http://www.telemidia.puc-rio.br/oohdm/oohdm.html>.
- Bernd Oesterreich, Objektorientierte Softwareentwicklung, 4. aktual.Auflage,
<http://www.oose.de/uml>.
- Ivar Jacobson, Magnus Christerson, Patrik Jonsson, Gunnar Overgaard:
Object-Oriented Software Engineering – A Use Case Driven Approach,
Addison-Wesley, 1992.
- Nora Koch & Luis Mandel, Using UML to Design Hypermedia Applications,
Technischer Bericht 9901, Institut für Informatik an der Ludwig – Maximilians –
Universität München, März 1999.
- UML Semantics, 1997, Sanata Clara, USA, Rational Inc., Version 1.1
- Rumbaugh J., Blaha M., Premerlani W., Eddy F., Lorensen W., 1999, Object-
Oriented Modeling and Design, Prentice-Hall Editions.
- <http://www.omg.org>

11 Anhang A: Auszüge aus dem Quelltext

12 Anhang B: Beispiel für die fertige Applikation