

**ENTWURF UND IMPLEMENTIERUNG EINES  
WEBPORTALS ZUR REGISTRIERUNG UND VERWALTUNG  
VON WEITERBILDUNGSVERANSTALTUNGEN AUF BASIS  
VON MYSQL UND PHP**

Christian Schäfer  
Fachbereich Elektrotechnik  
FH Würzburg-Schweinfurt  
Abt. Schweinfurt

15.09.2006

## Inhaltsverzeichnis

1	Einleitung .....	6
2	Anforderungen .....	7
2.1	Umgebungs-Anforderungen .....	7
2.2	Nicht-Funktionelle Anforderungen .....	7
2.3	Funktionale Anforderungen .....	7
2.3.1	Frontend-Funktionalitäten .....	7
2.3.1.1	Veranstaltungskatalog browsen .....	8
2.3.1.2	Details einer Veranstaltung anzeigen .....	9
2.3.1.3	Veranstaltungskatalog durchsuchen .....	10
2.3.1.4	Registrierung .....	11
2.3.1.5	Auf Webseite einloggen .....	12
2.3.1.6	Zu einer Veranstaltung anmelden .....	13
2.3.1.7	Anmeldungen anzeigen .....	14
2.3.1.8	Veranstaltung vormerken .....	14
2.3.1.9	Vormerker anzeigen .....	15
2.3.1.10	Persönliche Daten bearbeiten .....	16
2.3.1.11	Kontakt aufnehmen .....	17
2.3.1.12	Informationen lesen .....	17
2.3.1.13	Veranstaltungs-Vorschau einsehen .....	18
2.3.1.14	Weitere Akademien anzeigen .....	19
2.3.2	Backend-Funktionalitäten .....	19
2.3.2.1	Akademie verwalten .....	20
2.3.2.2	Backend-Rechte zuweisen .....	21
2.3.2.3	Benutzer löschen .....	21
2.3.2.4	Passwort zurücksetzen .....	22
2.3.2.5	Veranstaltungen verwalten .....	23
2.3.2.6	Veranstaltung anlegen .....	23
2.3.2.7	Veranstaltung pflegen .....	24
2.3.2.8	Teilnehmer benachrichtigen .....	25
2.3.2.9	Instruktor benachrichtigen .....	26
2.3.2.10	Veranstaltung löschen .....	27
2.3.2.11	Anmeldungen einsehen .....	28
2.3.2.12	Vormerker einsehen .....	28
2.3.2.13	Kategorien verwalten .....	29
2.3.2.14	Kategorie anlegen .....	29
2.3.2.15	Kategorie pflegen .....	30
2.3.2.16	Kategorie löschen .....	31
3	Projektplanung .....	32
3.1	Ablauf der Bearbeitungszeit .....	32
3.2	Milestonesplan .....	32
4	Design .....	35
4.1	Screen-Design .....	35
4.1.1	Startseite .....	35
4.1.2	Veranstaltungs-Übersicht .....	36
4.1.2.1	Übersicht nach Kategorie .....	36
4.1.2.2	Übersicht nach Datum .....	37
4.1.2.3	Übersicht nach Alphabet .....	38
4.1.3	Suche .....	39

4.1.4	Informationen .....	41
4.1.5	Persönliches Profil.....	41
4.2	Software-Design.....	44
4.2.1	Allgemein .....	45
4.2.2	Drei-Schicht-Architektur.....	45
4.2.3	Datenmodell und Abstraktion .....	46
4.2.3.1	Abstraktion der Datenhaltung .....	46
4.2.3.2	Objekt- und Datenmodell .....	47
4.2.3.3	Data-Mapper.....	50
4.2.4	Business-Logik.....	51
4.2.5	Präsentations-Schicht .....	51
4.2.6	PlugIn-Mechanismus.....	52
5	System-Architektur .....	53
5.1	Webserver.....	54
5.2	Loadbalancer .....	54
5.3	NFS-Server.....	55
5.4	MySQL-Server .....	55
5.5	Netzwerk .....	56
5.6	Internet .....	56
5.7	Client .....	56
5.8	Weitere Komponenten.....	57
6	Implementierung .....	58
6.1	Ausführungsplan der Software.....	58
6.2	Software-Komponenten .....	60
6.2.1	Apache-HTTP-Server.....	60
6.2.2	PHP.....	60
6.2.3	MySQL.....	60
6.2.4	System-Programme .....	61
6.2.5	PHP-Core-Klassen .....	61
6.2.6	Daten-Schicht .....	63
6.2.7	Business-Schicht .....	65
6.2.8	Präsentations-Schicht .....	66
6.3	Sequenz-Diagramme .....	69
6.3.1	Übersicht .....	69
6.3.2	Sequenz-Diagramme der PlugIns.....	71
6.4	Statistiken.....	93
7	Installationsanleitung .....	94
7.1	Aufbau des Installationsverzeichnis .....	94
7.2	Installation.....	94
7.2.1	Datenbank-Initialisierung.....	94
7.2.2	Applikations-Initialisierung .....	95
7.2.3	Konfiguration der Programm-Dateien .....	95
7.2.4	Installation der Programm-Dateien .....	97
7.2.5	Einrichten der Cronjobs .....	97
7.2.6	Sicherung der Konfiguration.....	98
7.3	Anlegen einer weiteren Akademie .....	98
8	Ausblick .....	99
9	Literatur- und Abbildungs-Verzeichnis .....	100
9.1	Literaturverzeichnis.....	100
9.2	Abbildungsverzeichnis .....	101

10	Lizenzbestimmungen .....	102
----	--------------------------	-----

## Preface

Am Anfang dieses Dokuments seien diejenigen Personen explizit erwähnt, die zum positiven Verlauf des Projekts beigetragen haben:

*Für die gute Betreuung in den Vorgesprächen und der Vorbereitung des Projekts, sowie im Erfahrungsaustausch und der konstruktiven Diskussion während der Bearbeitung, sei Herrn Prof. Dr.-Ing. Ludwig Eckert ein herzlicher Dank ausgesprochen.*

*Zur Steigerung der Qualität und Sicherheit der Anwendung hat Herr Dipl.-Ing. (FH) Reiner Rottmann im Bereich des Reviews mit ausführlichen Tests, guten Anmerkungen und Hinweisen, in sehr hohem Maß beigetragen.*

*Frau Annika Achatz stand dem Autor für Fragen zur sprachlichen Gestaltung, zur Lektorierung der Texte und für den moralischen Beistand stets zur Verfügung.*

# 1 Einleitung

Das folgende Dokument beschäftigt sich mit dem Thema

*„Entwurf und Implementierung eines Webportals zur Registrierung und Verwaltung von Weiterbildungsveranstaltungen auf Basis von MySQL und PHP“.*

Die FH Schweinfurt möchte sich mit einer eigens dafür gegründeten Akademie stärker für die industrielle Weiterbildung stark machen. Dabei ist ihr nicht nur daran gelegen, neue Forschungserkenntnisse und Ideen der Hochschulen in die Wirtschaft einzubringen, sondern auch Erfahrungen der Wirtschaft an Hochschul-Studenten zu vermitteln. Um den Wissenstransfer breitbandig gestalten zu können soll dazu im Internet eine Plattform geschaffen werden, die das Angebot der ins Leben gerufenen Akademie präsentiert und mit der es gleichzeitig möglich ist, das dargestellte Angebot zu verwalten. Da die Internet-Affinität in den letzten Jahren in sehr starkem Maße zugenommen hat, wird Präsentation der Veranstaltungen, Registrierung und Anmeldung hauptsächlich über das zu entwerfende Portal abgewickelt. Dieses ist so ausgelegt, dass mehrere Akademien erstellt werden können und somit Synergien zwischen unterschiedlichen Einrichtungen besser nutzbar sind. In erster Instanz wird die Software jedoch ausschließlich von der Akademie der FH Schweinfurt genutzt. Die Verantwortlichen und Dozenten sollen die Möglichkeit haben, das Angebot einfach einzustellen, zu gestalten und zu pflegen.

## 2 Anforderungen

### 2.1 Umgebungs-Anforderungen

Als technische Basis der zu erstellenden Webapplikation ist die *Skriptsprache PHP* und die frei verfügbare *MySQL-Datenbank* zu wählen. Bei der Codierung ist darauf zu achten, dass die Software zur *PHP-Version 4.4.x* und zur *MySQL-Version 4.0.x* kompatibel gehalten wird um sie später auf dem Virtual-Host-Server [www.w3service.net](http://www.w3service.net) installieren zu können.

Bei der Implementierung der Applikation ist auf objektorientiertes Design und die Verwendung von Entwurfsmustern für Standard-Problemstellung zu setzen. Die Software soll sich ebenso im Allgemeinen durch eine modulare Struktur und einfache Erweiterbarkeit auszeichnen. Es soll auf einfache Weise möglich sein weitere Funktionen nach dem **Modul- bzw. PlugIn-Verfahren** einzuklinken. Bei Konzeption und Erstellung ist zudem auf Mandantenfähigkeit zu achten. Dies bedeutet, dass mehrere Akademien auf demselben Web- bzw. Applikations-Server gehostet und verwaltet werden können. Dadurch soll eine übergreifende Verwendbarkeit der Benutzer-Profile ermöglicht werden. Für die Verwaltung des Weiterbildungs-Portals ist ein Backend bereitzustellen, mit dem autorisierte Benutzer die Software per Weboberfläche betreuen können.

### 2.2 Nicht-Funktionelle Anforderungen

Die nicht-funktionellen Anforderungen lassen sich für eine Webapplikation in wenigen Sätzen zusammenfassen.

Im Allgemeinen wird unterstellt, dass eine „gute“ Webapplikation schnelle Ausführungs- und Antwortzeiten zwischen 0s und 3s besitzen sollte. Darüber hinaus wird vom Design der Oberfläche Übersichtlichkeit, benutzerfreundliche Positionierung der wichtigen Informationen, gute Menüführung und gute Erreichbarkeit der Eingabe- und Bedienelemente erwartet. An dieser Erwartungshaltung wird auch die zu erstellende Software gemessen werden.

### 2.3 Funktionale Anforderungen

#### 2.3.1 Frontend-Funktionalitäten

Das Frontend bildet diejenigen Funktionen ab, die allen Benutzern der Software zugänglich sein sollen und werden folglich als Frontend-Funktionalitäten bezeichnet.

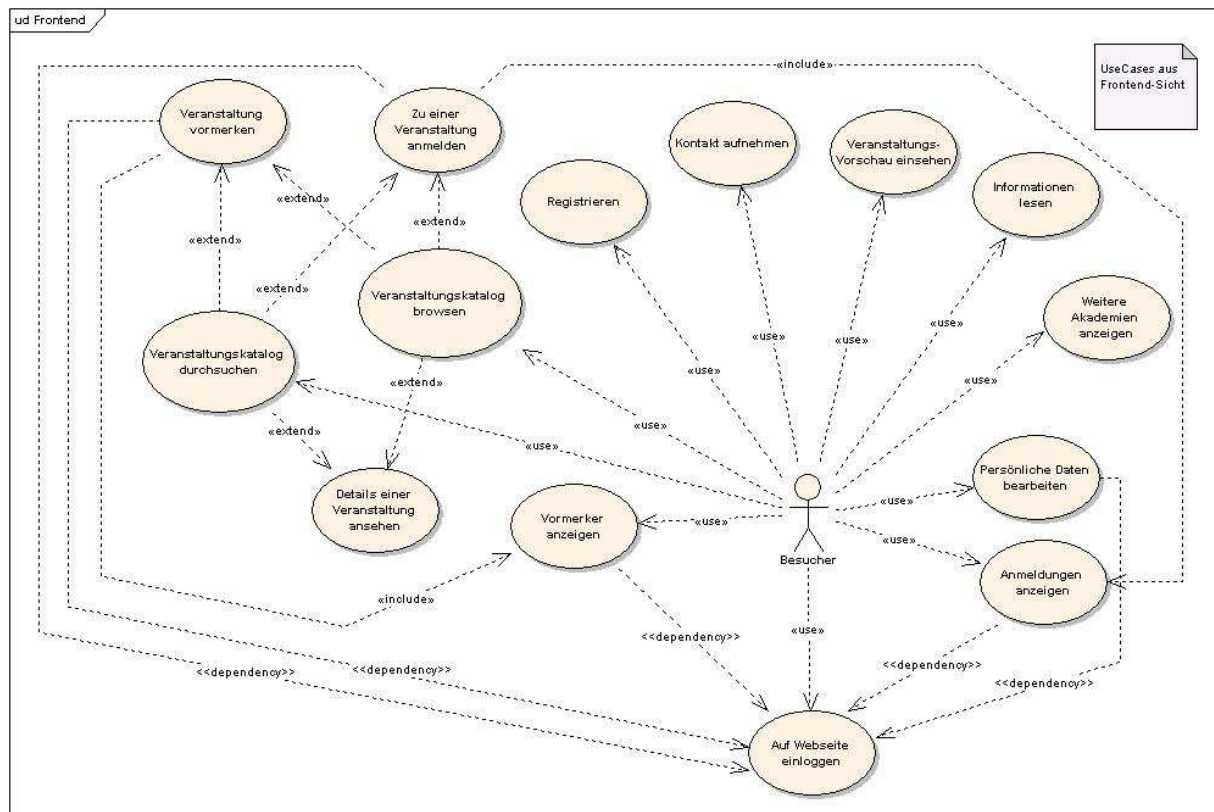


Abb. 2-1: Use Cases Frontend

Abbildung 2-1 zeigt die gewünschten Anforderungen in Form eines UML-Diagramms. Die einzelnen Use Cases werden im Folgenden näher betrachtet.

### 2.3.1.1 Veranstaltungskatalog browsen

System:	GUI
Use Case:	Veranstaltungskatalog browsen
Beschreibung:	Der Use Case <i>Veranstaltungskatalog browsen</i> repräsentiert die Anforderung, die Veranstaltungen in einer Übersicht anzeigen zu können. Dabei werden die jeweiligen Veranstaltungen mit den Attributen <i>Titel</i> , <i>Datum</i> , <i>Dozenten</i> und <i>Kurzbeschreibung</i> aufgeführt. Nähere Informationen können mit Klick auf einen Button <i>Details</i> eingesehen werden. Es muss dem Benutzer in dieser Darstellung ermöglicht werden die Liste nach <i>Kategorien</i> , <i>Datum</i> oder <i>Alphabet</i> sortiert anzeigen zu lassen.
Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Veranstaltungskatalog durchsuchen
Extensions:	<ul style="list-style-type: none"> <li>• Zu einer Veranstaltung anmelden</li> <li>• Eine Veranstaltung vormerken</li> <li>• Details einer Veranstaltung einsehen</li> </ul>
Includes:	-
Dependencies:	-
Vorbedingungen:	Der Besucher muss die Startseite der Akademie aufgerufen haben.
<b>Standard Workflow:</b>	



1. Der Benutzer wählt den Menüpunkt **Veranstaltungen**.
2. Er wählt ein Sortier-Kriterium (Kategorie, Datum, Alphabet) und bestätigt die Auswahl mit dem OK-Button.
3. Die Applikation ändert das Auswahlmenü für Kategorie, Datum und Alphabet und stellt die jeweils möglichen Werte dar.
4. Der Benutzer wählt einen Wert für Kategorie, Datum und Alphabet und quittiert die Auswahl mit OK.
5. Die Software zeigt die unter den gewählten Kriterien vorhandenen Veranstaltungen in einer Liste an.

**Alternativer Workflow:**

n/a

**Fehlerbehandlung:**

1. Sollte unter den gewählten Kriterien keine Veranstaltungen vorhanden sein, so wird eine entsprechende Meldung angezeigt.

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.2 Details einer Veranstaltung anzeigen

System:	GUI
Use Case:	Details einer Veranstaltung anzeigen
Beschreibung:	In der Detailansicht einer Veranstaltung werden alle Attribute einer Veranstaltung angezeigt. Zu diesen zählen <i>Titel, Kursnummer, Datum, Kategorie, Kurzbeschreibung, maximale / minimale Teilnehmerzahl, aktuelle Anmeldungszahlen</i> und <i>Detailbeschreibung</i> . <i>Kurzbeschreibung</i> und <i>Detailbeschreibung</i> müssen als frei gestaltbare Texte ausgelegt sein.
Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Veranstaltung in der Detailansicht ansehen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	Der Besucher muss eine Veranstaltungs-Liste unter den Menüpunkten Startseite, Veranstaltungen oder Suche aufgerufen haben.

**Standard Workflow:**

1. Der Benutzer klickt bei der gewünschten Veranstaltung auf „Details“.
2. Die Applikation zeigt die gewählte Veranstaltung in einer Detailansicht im Content-Bereich an.

**Alternativer Workflow:**

n/a	
<b>Fehlerbehandlung:</b>	
n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.3 Veranstaltungskatalog durchsuchen

System:	GUI
Use Case:	Veranstaltungskatalog durchsuchen
Beschreibung:	Um das Finden von Weiterbildungs-Seminaren zu erleichtern, muss eine Suche angeboten werden, mit der der Veranstaltungskatalog an Hand der Attribute <i>Titel</i> , <i>Dozent</i> , <i>Datum</i> und <i>Beschreibung</i> durchsucht werden kann. Dabei ist die Möglichkeit vorzusehen Wildcards wie „*“ ( <b>mehrere</b> beliebige Zeichen) und „?“ ( <b>ein</b> beliebiges Zeichen) in den Suchfeldern benutzen zu können. Die Suche ist technisch so auszulegen, dass die Ergebnisse der Einzelsuchen zu einem Gesamt-Ergebnis korreliert werden.
Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Veranstaltung in der Datenbank suchen
Extensions:	<ul style="list-style-type: none"> <li>• Details einer Veranstaltung ansehen</li> <li>• Zu einer Veranstaltung anmelden</li> <li>• Veranstaltung vormerken</li> </ul>
Includes:	-
Dependencies:	-
Vorbedingungen:	Der Besucher muss die Startseite der Akademie aufgerufen haben.
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer wechselt zum Menüpunkt <b>Suche</b>.</li> <li>2. Er füllt die Suchmaske mit beliebigen Werten für <i>Titel</i>, <i>Dozent</i>, <i>Datum</i> und <i>Beschreibung</i> aus.</li> <li>3. Die Applikation zeigt eine Liste der zu den Suchkriterien gefundenen Veranstaltungen an.</li> </ol>	
<b>Alternativer Workflow:</b>	
n/a	
<b>Fehlerbehandlung:</b> <ol style="list-style-type: none"> <li>1. Die Software zeigt einen Hinweis an, falls keine Veranstaltung zu den eingegebenen Suchkriterien gefunden wurde. Daraufhin kann der Besucher die Suchkriterien verändern und eine neue Suche starten.</li> </ol>	
Nachbedingungen:	-

Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.4 Registrierung

System:	GUI
Use Case:	Registrierung
Beschreibung:	<p>Personenbezogenen Funktionen wie <i>Anmelden</i>, <i>Vormerken</i> und <i>Bearbeiten der Personendaten</i> können nur nach einmaliger vorheriger Registrierung genutzt werden. Dabei müssen die persönlichen Daten in einem Profil hinterlegt werden, wobei die Firmen-E-Mail verpflichtend auszufüllen ist, da diese für alle weiteren Funktionen als primäre Kontakt-E-Mail eingesetzt werden soll. Zur Bestätigung der Registrierung ist eine E-Mail an die entsprechende Person zu versenden.</p> <p>Ein Profil muss folgende Daten beinhalten:</p> <ul style="list-style-type: none"> <li>- Anrede</li> <li>- Titel</li> <li>- Name</li> <li>- Vorname</li> <li>- Geburtsdatum</li> <li>- Privat-Adresse</li> <li>- Dienst-Adresse</li> <li>- Memo (Freitext)</li> <li>- Benutzer</li> <li>- Passwort</li> <li>- Gebuchte Veranstaltungen</li> <li>- Vorgemerkte Veranstaltungen</li> <li>- Rolle</li> </ul> <p>In Privat- und Dienst-Adresse sollen jeweils Adressen gespeichert werden können, die die Attribute</p> <ul style="list-style-type: none"> <li>- Straße</li> <li>- PLZ</li> <li>- Ort</li> <li>- Telefon</li> <li>- Mobil</li> <li>- Firma</li> <li>- E-Mail</li> <li>- Webseite</li> </ul> <p>enthalten.</p>
Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Registrierung durchführen um einen Account anzulegen
Extensions:	-
Includes:	-

Dependencies:	-
Vorbedingungen:	Der Besucher muss die Startseite der Akademie aufgerufen haben.
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer wechselt zum Menüpunkt <b>Registrierung</b>.</li> <li>2. Er füllt die Registrierungsmaske gemäß den Hinweistexten aus und bestätigt die Eingaben.</li> <li>3. Das System führt die Registrierung des Benutzers durch und verschickt eine Bestätigungse-Mail mit den eingegebenen Zugangsdaten und Hinweisen zur Benutzung der Plattform.</li> </ol> <b>Alternativer Workflow:</b> n/a	
<b>Fehlerbehandlung:</b> <ol style="list-style-type: none"> <li>1. Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder wird die Registrierung akzeptiert.</li> </ol>	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.5 Auf Webseite einloggen

System:	GUI
Use Case:	Auf Webseite einloggen
Beschreibung:	Ein Benutzer muss sich mit seinen Profildaten zu jeder Zeit auf dem Portal erneut einloggen können. Dazu ist eine auf allen Seiten verfügbare Login-Maske bereit zu stellen. Sollte ein Benutzer seine Zugangsdaten verloren haben, so muss er die Möglichkeit besitzen sich per E-Mail an einen Administrator zu wenden, um seine Zugangsdaten zurücksetzen zu lassen.
Akteur/Trigger:	Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Benutzer auf der Webseite einloggen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	Der Besucher muss die Startseite der Akademie aufgerufen haben.
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer gibt seine Zugangsdaten, die er bei der Registrierung gewählt hat, in die Felder <i>Benutzer</i> und <i>Passwort</i> ein.</li> <li>2. Die Zugangsdaten werden vom Benutzer mit Klick auf den Button <i>Login</i> bestätigt.</li> <li>3. Die Software zeigt das für die Rolle des Benutzers passende interne Menü.</li> </ol>	

**Alternativer Workflow:**

n/a

**Fehlerbehandlung:**

1. Wurden keine Daten eingegeben, so werden die betroffenen Felder rot umrandet. Hat der Benutzer falsche oder nicht vorhandene Benutzerdaten eingegeben, so wird ein Hinweistext ausgegeben, der den Benutzer auf diese Tatsache aufmerksam macht.

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

**2.3.1.6 Zu einer Veranstaltung anmelden**

System:	GUI
Use Case:	Zu einer Veranstaltung anmelden
Beschreibung:	Wurde das Interesse des Besuchers an einer Veranstaltung geweckt, kann er sich zu dieser per Klick anmelden. Voraussetzung ist, dass der Benutzer mit seinem Profil angemeldet ist. Die Veranstaltung erscheint dann im Sammel-Korb „gebuchte Veranstaltungen“.
Akteur/Trigger:	Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Zu einer Veranstaltung anmelden
Extensions:	-
Includes:	Anmeldungen anzeigen
Dependencies:	Auf Webseite einloggen
Vorbedingungen:	Der Besucher muss eine Veranstaltungs-Liste unter den Menüpunkten Startseite, Veranstaltungen oder Suche aufrufen.

**Standard Workflow:**

1. Der Teilnehmer klickt auf den Button *Anmelden*.
2. Das System meldet den Benutzer zur ausgewählten Veranstaltung an.
3. Dem Benutzer wird die Übersicht der angemeldeten Veranstaltungen angezeigt.

**Alternativer Workflow:**

1. Falls der Benutzer bereits zu dieser angemeldet ist wird ihm die Übersicht der angemeldeten Veranstaltungen angezeigt ohne eine weitere Anmeldung auszulösen.

**Fehlerbehandlung:**

n/a

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.7 Anmeldungen anzeigen

System:	GUI
Use Case:	Anmeldungen anzeigen
Beschreibung:	Der Use Case <i>Anmeldungen anzeigen</i> beschreibt die Funktionen des Anmeldungs-Korbs. Dieser zeigt eine Liste derjenigen Veranstaltungen, zu denen sich ein Benutzer angemeldet hat. Darin enthaltene Weiterbildungs-Seminare müssen vom Besucher gelöscht bzw. im Detail betrachtet werden können.
Akteur/Trigger:	Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Eigene Anmeldungen anzeigen
Extensions:	-
Includes:	-
Dependencies:	Auf Webseite einloggen
Vorbedingungen:	Der Besucher muss auf der Webseite eingeloggt sein.
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Teilnehmer, Instruktor, Akademie-Manager oder Administrator wählt den Menüpunkt <b>Anmeldungen anzeigen</b>.</li> <li>2. Die Software liefert dem Besucher die Übersicht der aktuell angemeldeten Veranstaltung zurück.</li> </ol>	
<b>Alternativer Workflow:</b> <ol style="list-style-type: none"> <li>1. Hat der Benutzer keine Anmeldungen zu verzeichnen wird eine Meldung ausgegeben.</li> </ol>	
<b>Fehlerbehandlung:</b>  n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.8 Veranstaltung vormerken

System:	GUI
Use Case:	Veranstaltung vormerken
Beschreibung:	Zusätzlich zur Anmeldungs-Funktion soll dem Benutzer eine Vormerken-Funktion angeboten werden. Diese erleichtert das Suchen nach Veranstaltungen, indem interessante Veranstaltungen zum Vormerker-Korb hinzugefügt und später weiter bearbeitet werden können.
Akteur/Trigger:	Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Veranstaltung vormerken
Extensions:	-
Includes:	Vormerker anzeigen
Dependencies:	Auf Webseite einloggen

Vorbedingungen:	Der Besucher muss eine Veranstaltungs-Liste unter den Menüpunkten Startseite, Veranstaltungen oder Suche aufgerufen. Es muss zudem ein erfolgreicher Login stattgefunden haben.
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer klickt auf den Button <b>Vormerken</b> in der Ansicht der gewünschten Veranstaltung.</li> <li>2. Das System legt die Veranstaltung in den Vormerker-Korb des Benutzers und zeigt die Übersicht der vorgemerkten Veranstaltungen an.</li> </ol> <b>Alternativer Workflow:</b> <ol style="list-style-type: none"> <li>1. Falls der Benutzer diese Veranstaltung bereits vorgemerkt hat wird ihm die Übersicht der vorgemerkten Veranstaltungen angezeigt ohne einen weiteren Vormerker auf diese Veranstaltung zu generieren.</li> </ol> <b>Fehlerbehandlung:</b> n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.9 Vormerker anzeigen

System:	GUI
Use Case:	Vormerker anzeigen
Beschreibung:	Ähnlich dem Anmeldungs-Korb werden hier die vorgemerkten Veranstaltungen angezeigt. Ein Button <i>Löschen</i> soll dem Benutzer ermöglichen nicht mehr relevante Seminare aus dem Korb zu entfernen, mit <i>Details</i> soll er die Veranstaltungsinformationen einsehen können.
Akteur/Trigger:	Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Vormerker des Benutzers anzeigen
Extensions:	-
Includes:	-
Dependencies:	Auf Webseite einloggen
Vorbedingungen:	Der Besucher muss erfolgreich auf der Akademie eingeloggt sein.
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer klickt auf den Menüpunkt <b>Vormerker verwalten</b>.</li> <li>2. Das System legt die Veranstaltung in den Vormerker-Korb des Benutzers und zeigt die Übersicht der vorgemerkten Veranstaltungen an.</li> </ol> <b>Alternativer Workflow:</b> <ol style="list-style-type: none"> <li>1. Hat der Benutzer keine Vormerker zu verzeichnen wird eine Meldung ausgegeben.</li> </ol>	

<b>Fehlerbehandlung:</b>	
n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.10 Persönliche Daten bearbeiten

System:	GUI
Use Case:	Persönliche Daten bearbeiten
Beschreibung:	Unter <i>Persönliche Daten bearbeiten</i> versteht sich die Pflege des eigenen Profils. Dem Benutzer muss die Möglichkeit gegeben werden, alle bei der Registrierung abgefragten Daten bearbeiten zu können.
Akteur/Trigger:	Teilnehmer, Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Persönliche Profil-Daten bearbeiten
Extensions:	-
Includes:	-
Dependencies:	Auf Webseite einloggen
Vorbedingungen:	Der Besucher muss erfolgreich auf der Akademie eingeloggt sein.

#### Standard Workflow:

1. Der Benutzer klickt auf den Menüpunkt **Persönliche Daten bearbeiten**.
2. Das System lädt die Daten des Profils in eine Bearbeitungsmaske und bietet dem Benutzer einen Button zur Bestätigung der Änderungen an.
3. Der Benutzer nimmt die Änderung seiner Daten in den vorgesehenen Feldern vor.
4. Die Änderungen werden vom Benutzer mit Klick auf den Button **Ändern** vom System gespeichert.

#### Alternativer Workflow:

n/a

#### Fehlerbehandlung:

1. Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder wird die Änderung akzeptiert.

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-



### 2.3.1.11 Kontakt aufnehmen

System:	GUI
Use Case:	Kontakt aufnehmen
Beschreibung:	Wie bereits erwähnt, ist eine standardisierte Methode vorzusehen, wie ein Besucher mit dem Betreiber der Plattform Kontakt aufnehmen kann. Dazu ist ein Formular einzubinden, das einem Besucher erlaubt, seine Anfrage per E-Mail an Verantwortliche zu versenden.
Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager
Haupt-Szenario:	Kontakt mit einem Administrator aufnehmen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer klickt auf den Menüpunkt <b>Kontakt</b>.</li> <li>2. Die Software zeigt dem Besucher, Teilnehmer, Instruktor oder Akademie-Manager ein Kontaktformular mit den Feldern Kontaktperson, Name, E-Mail, Betreff und Text an. Im Feld Kontaktperson werden die Administratoren der Software aufgelistet.</li> <li>3. Die Felder werden vom Benutzer ausgefüllt.</li> <li>4. Das System nimmt die Eingaben entgegen, generiert eine Kontakt-E-Mail und verschickt diese an den gewählten Administrator.</li> </ol> <b>Alternativer Workflow:</b> <p>n/a</p> <b>Fehlerbehandlung:</b> <ol style="list-style-type: none"> <li>1. Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder wird die Kontakt-E-Mail versendet.</li> </ol>	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.12 Informationen lesen

System:	GUI
Use Case:	Informationen lesen
Beschreibung:	<i>Informationen lesen</i> ist ein Platzhalter für alle auf dem Portal angebotenen Informationen. Hierzu zählen sämtliche veranstaltungsrelevanten Inhalte, sowie insbesondere Hilfe-Seiten. Letztere sind mit Hilfestellungen auszustatten, die dem Benutzer die Bedienung der Seite erleichtert.

Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager
Haupt-Szenario:	Informationen und Hilfetexte lesen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer klickt auf den Menüpunkt <b>Hilfe</b>.</li> <li>2. Die Software zeigt dem Besucher, Teilnehmer, Instruktor, Akademie-Manager oder Administrator je nach Rolle des Benutzers die für ihn relevanten Hilfetexte an.</li> </ol> <b>Alternativer Workflow:</b> n/a	
<b>Fehlerbehandlung:</b> n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.13 Veranstaltungsvorschau einsehen

System:	GUI
Use Case:	Veranstaltungsvorschau einsehen
Beschreibung:	Die Startseite des Portals ist mit einer Veranstaltungsvorschau für das Seminarportfolio auszustatten. Die genaue Anzahl der zu zeigenden Veranstaltungen ist noch abzustimmen, bzw. als konfigurierbare Ausgabe zu implementieren.
Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager
Haupt-Szenario:	Darstellung der Veranstaltungsvorschau
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	Der Besucher muss die Startseite der Akademie aufgerufen haben.
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Benutzer klickt auf den Menüpunkt <b>Startseite</b> oder befindet sich bereits auf diesem.</li> <li>2. Die Software unterbreitet dem Besucher eine Veranstaltungsvorschau über die nächsten drei Veranstaltungen.</li> </ol> <b>Alternativer Workflow:</b> n/a	

<b>Fehlerbehandlung:</b>	
1. Finden keine Veranstaltungen statt, deren Startdatum in der Zukunft liegt, so wird eine entsprechende Meldung angezeigt.	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.1.14 Weitere Akademien anzeigen

System:	GUI
Use Case:	Weitere Akademien anzeigen
Beschreibung:	Die Mehrmandantenfähigkeit bringt mit sich, dass mehrere Akademien auf der zu erstellenden Software betrieben werden. Um die Zusammenarbeit der einzelnen Projekte zu erleichtern, ist eine Übersicht bzw. Liste der auf der Plattform betriebenen Portale vorzusehen.
Akteur/Trigger:	Besucher, Teilnehmer, Instruktor, Akademie-Manager
Haupt-Szenario:	Weitere Akademien anzeigen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	Der Besucher muss die Startseite der Akademie aufgerufen haben.
<b>Standard Workflow:</b>	
<ol style="list-style-type: none"> <li>Der Benutzer klickt auf den Menüpunkt <b>Weitere Akademien</b>.</li> <li>Das System zeigt dem Benutzer die in der Datenbank angelegten Akademien in einer Liste an.</li> </ol>	
<b>Alternativer Workflow:</b>	
n/a	
<b>Fehlerbehandlung:</b>	
n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2 Backend-Funktionalitäten

Das Kapitel Backend-Funktionalitäten behandelt die zur Verwaltung der Software notwendigen Funktionen. Das beinhaltet manuell zu verrichtende Aufgaben sowie Automatismen.

Um die Verwaltungsaufgaben an verschiedene Personen mit unterschiedlichen Kompetenzen verteilen zu können, sind mehrere **feste** Rollen mit jeweils eigenen Rechten zu implementieren. Zum einen gibt es *Administratoren*, die alle Rechte auf das Backend haben und alle Aktionen ausführen können, außerdem *Akademie-Manager*, die alle Verwaltungsaufgaben innerhalb einer Akademie wahrnehmen können und *Instruktoren*, die lediglich Rechte auf ihre Veranstaltungen besitzen. Zum anderen ist die Rolle *CronJob* vorzusehen, die für die Erledigung der automatischen Aufgaben. Ein neu registrierter Benutzer soll zunächst die Rolle *Teilnehmer* erhalten, womit er lediglich Rechte auf sein eigenes Profil besitzt. Er kann daher keine Verwaltungsaufgaben wahrnehmen.

Im folgenden UML-Diagramm wird vorausgesetzt, dass der dort agierende Instruktor, Akademie-Manager oder Administrator eingeloggt ist. Diese Bedingung wird in den Use Case Templates deshalb nicht explizit erwähnt.

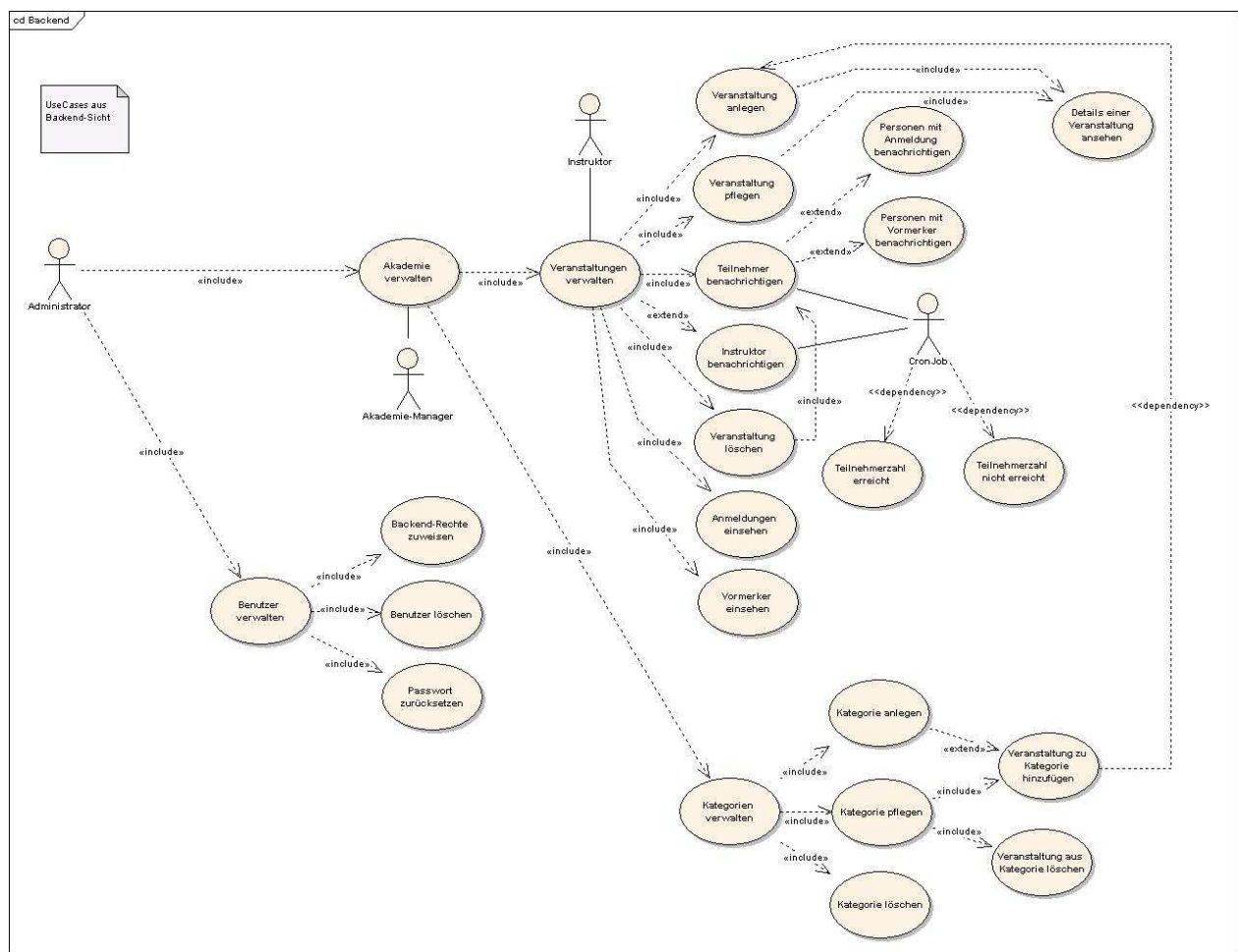


Abb. 2-2: Use Cases Backend

### 2.3.2.1 Akademie verwalten

Der Punkt *Akademie verwalten* gruppiert, wie in Abbildung 2-2 zu sehen ist, sämtliche Funktionen des Backends. Alle Personen der Rolle *Administrator* können diese Aktionen ausführen.

### 2.3.2.2 Backend-Rechte zuweisen

System:	GUI
Use Case:	Backend-Rechte zuweisen
Beschreibung:	Um Verwaltungsaufgaben in einer Akademie vornehmen zu können muss eine registrierte Person die Rollen <i>Administrator</i> , <i>Akademie-Manager</i> oder <i>Instruktor</i> innehaben. Der <i>Administrator</i> soll dazu eine Auswahl-Maske bereitgestellt bekommen, mit der er bestehenden Benutzern Rollen mit entsprechenden Zugriffsrechten auf das Backend zuweisen kann.
Akteur/Trigger:	Administrator
Haupt-Szenario:	Rechte auf Backend-Funktionen zuweisen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Administrator klickt auf den Menüpunkt <b>Rechte zuweisen</b>.</li> <li>2. Er wählt einen Benutzer aus, dem eine der Rollen Teilnehmer, Akademie-Manager oder Administrator zugewiesen werden soll und bestätigt die Auswahl.</li> <li>3. Die Software zeigt dem Administrator eine Sicherheitsabfrage an, die aus Sicherheitsgründen zur Zuweisung nochmals bestätigt werden muss.</li> <li>4. Das System weist bei positiver Bestätigung dem gewählten Benutzer die gewünschte Rolle zu und verschickt eine Benachrichtigungs-E-Mail.</li> </ol> <b>Alternativer Workflow:</b> <ol style="list-style-type: none"> <li>1. Verneint der Administrator die Bestätigung, so wird dem Benutzer die Rolle nicht zugewiesen und die Aktion abgebrochen.</li> </ol> <b>Fehlerbehandlung:</b> n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.3 Benutzer löschen

System:	GUI
Use Case:	Benutzer löschen
Beschreibung:	Veraltete oder nicht mehr gültige Benutzerprofile müssen bereinigt werden können. Dies soll mit Hilfe des Use Case <i>Benutzer löschen</i> möglich sein, wobei die Löschung eines Accounts aus

	Datenkonsistenzgründen alle profilbezogenen Daten beinhalten muss.
Akteur/Trigger:	Administrator
Haupt-Szenario:	Benutzer löschen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Administrator klickt auf den Menüpunkt <b>Benutzer löschen</b>.</li> <li>2. Er wählt einen Benutzer aus, den er löschen möchte bestätigt die Auswahl.</li> <li>3. Die Software zeigt dem Administrator eine Sicherheitsabfrage an um die Absicht des Benutzers nochmals abzufragen.</li> <li>4. Das System löscht bei positiver Bestätigung den gewählten Benutzer aus der Datenbank und verschickt eine Benachrichtigungs-E-Mail.</li> </ol> <b>Alternativer Workflow:</b> <ol style="list-style-type: none"> <li>1. Verneint der Administrator die Bestätigung, so wird der Benutzer nicht gelöscht.</li> </ol> <b>Fehlerbehandlung:</b> n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.4 Passwort zurücksetzen

System:	GUI
Use Case:	Passwort zurücksetzen
Beschreibung:	Eine häufige administrative Tätigkeit ist das Zurücksetzen von Zugangsdaten.
Akteur/Trigger:	Administrator
Haupt-Szenario:	Passwort zurücksetzen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Administrator klickt auf den Menüpunkt <b>Passwort rücksetzen</b>.</li> <li>2. Er wählt den Benutzer aus, vergibt im Passwort-Feld ein neues Zugangs-Passwort und quittiert die Auswahl.</li> <li>3. Das System speichert die neuen Zugangsdaten und verschickt eine Benachrichtigungs-E-Mail.</li> </ol>	

**Alternativer Workflow:**

n/a

**Fehlerbehandlung:**

1. Versucht der Administrator ein leeres Passwort zu vergeben wird die Eingabe als ungültig markiert. Die Passwort-Änderung wird erst bei Eingabe eines Strings mit Zeichenlänge größer Null vom System akzeptiert.

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.5 Veranstaltungen verwalten

*Veranstaltungen verwalten* gruppiert weitere Use Cases, die die genauen Funktionalitäten beschreiben. Dieser Use Case und alle darunter befindlichen sollen auch einem *Instruktor* zugänglich sein, der nur die von ihm erstellten Veranstaltungen pflegen kann.

### 2.3.2.6 Veranstaltung anlegen

System:	GUI
Use Case:	Veranstaltung anlegen
Beschreibung:	<p>Personen, die der Rolle <i>Instruktor</i>, <i>Akademie-Manager</i> oder <i>Administrator</i> angehören, sollen Veranstaltungen anlegen und diese einer bereits bestehenden Kategorie hinzufügen können.</p> <p>Es sind folgende Attribute beim Anlegen auszufüllen:</p> <ul style="list-style-type: none"><li>- Titel</li><li>- Kurs-Nummer (z.B. FHSW-2006-78)</li><li>- Kategorie</li><li>- Start (Datum)</li><li>- Ende (Datum)</li><li>- Dozent</li><li>- Kurzer Beschreibungstext (Freitext-Feld)</li><li>- Detailbeschreibungstext (Freitext-Feld)<ul style="list-style-type: none"><li>▪ Kursform</li><li>▪ Ziel</li><li>▪ Inhalte</li><li>▪ Zielgruppe</li><li>▪ Voraussetzungen</li><li>▪ Kursform</li><li>▪ Teilnehmer-Modus</li><li>▪ Ort / Raum</li></ul></li></ul>

	<ul style="list-style-type: none"> <li>▪ Zeit / Zyklus</li> <li>▪ Kursgebühr</li> <li>▪ Ansprechpartner</li> <li>▪ Weitere Infos</li> <li>- Minimal erforderliche Teilnehmerzahl</li> <li>- Maximal zulässige Teilnehmerzahl</li> <li>- Aktuelle Anmeldungsanzahl</li> </ul>
Akteur/Trigger:	Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Veranstaltung anlegen
Extensions:	-
Includes:	Details einer Veranstaltung anzeigen
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Instruktor, Akademie-Manager bzw. der Administrator klickt auf den Menüpunkt <b>Veranstaltung anlegen</b>.</li> <li>2. Er füllt das angezeigte Formular mit den für die Veranstaltung relevanten Informationen und bestätigt die Eingabe mit Klick auf den Button <b>Anlegen</b>.</li> <li>3. Das System speichert die Daten der Veranstaltung und zeigt die Veranstaltungs-Detailansicht an.</li> </ol> <b>Alternativer Workflow:</b> n/a	
<b>Fehlerbehandlung:</b> <ol style="list-style-type: none"> <li>1. Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder wird die Veranstaltung gespeichert.</li> </ol>	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.7 Veranstaltung pflegen

System:	GUI
Use Case:	Veranstaltung pflegen
Beschreibung:	Alle Veranstaltungen einer Akademie müssen via Oberfläche pflegbar sein. Benutzer, die der Rolle <i>Administrator</i> angehören, können alle Veranstaltungen bearbeiten, Benutzer der Rolle <i>Akademie-Manager</i> sehen nur Veranstaltungen einer Akademie und <i>Instruktoren</i> nur ihre eigenen Veranstaltungen.
Akteur/Trigger:	Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Veranstaltung pflegen



Extensions:	-
Includes:	Details einer Veranstaltung anzeigen
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Instruktor, Akademie-Manager bzw. der Administrator klickt auf den Menüpunkt <b>Veranstaltung pflegen</b>.</li> <li>2. Er wählt eine der aufgelisteten Veranstaltungen aus der Liste aus und bestätigt die Auswahl.</li> <li>3. Die Software zeigt daraufhin die Bearbeitungsmaske der Veranstaltung an.</li> <li>4. Der Benutzer hat nun die Möglichkeit die Daten der Veranstaltung zu ändern. Am Ende der Bearbeitung bestätigt er die Eingaben mit Klick auf den Button <b>Bearbeiten</b>.</li> <li>5. Das System speichert die Daten der Veranstaltung und zeigt die Veranstaltungs-Detailansicht an.</li> </ol> <b>Alternativer Workflow:</b> <p>n/a</p> <b>Fehlerbehandlung:</b> <ul style="list-style-type: none"> <li>• Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder wird die bearbeitete Veranstaltung gespeichert.</li> </ul>	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.8 Teilnehmer benachrichtigen

System:	GUI
Use Case:	Teilnehmer benachrichtigen
Beschreibung:	Im Rahmen von Terminverschiebungen, Änderung von Inhalten, Absagen oder Zusagen muss ein Dozent die Möglichkeit haben, diejenigen Teilnehmer per Verwaltungs-Oberfläche zu benachrichtigen, welche die aktuelle Veranstaltung im Vormerker-Korb abgelegt oder sich zu dieser bereits angemeldet haben. Die Benachrichtigung soll technisch gesehen per E-Mail erfolgen.
Akteur/Trigger:	Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Teilnehmer benachrichtigen
Extensions:	<ul style="list-style-type: none"> <li>• Personen mit Anmeldung benachrichtigen</li> <li>• Personen mit Vormerker benachrichtigen</li> </ul>
Includes:	-
Dependencies:	-

Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Instruktor, Akademie-Manager bzw. der Administrator klickt auf den Menüpunkt <b>Teilnehmer benachrichtigen</b>.</li> <li>2. Er wählt im angezeigten Dialog, ob er bereits angemeldete Personen, oder solche, die die Veranstaltung vorgemerkt haben, oder beide Personenkreise benachrichtigen möchte und füllt das Feld Betreff und den Text der E-Mail mit Inhalt.</li> <li>3. Durch bestätigen der Eingabe mit dem Button <b>Absenden</b> erfolgt entsprechend die Benachrichtigung per E-Mail.</li> </ol> <b>Alternativer Workflow:</b> n/a	
<b>Fehlerbehandlung:</b> <ol style="list-style-type: none"> <li>1. Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder wird die Benachrichtigung versandt.</li> </ol>	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.9 Instruktor benachrichtigen

System:	Konsole
Use Case:	Instruktor benachrichtigen
Beschreibung:	Benachrichtigungen sind bei den Events „Teilnehmerzahl innerhalb der Anmelde-Frist nicht erreicht“ und „Teilnehmerzahl innerhalb der Anmelde-Frist erreicht“ jeweils an den oder die Dozenten zu versenden. Hierzu ist ein CronJob einzurichten, der das Auftreten der genannten Events in einem definierten Zeitintervall für alle Veranstaltungen prüft.
Akteur/Trigger:	CronJob
Haupt-Szenario:	Teilnehmer benachrichtigen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Cronjob prüft alle Veranstaltungen auf die beschreiben Events.</li> <li>2. Die Instruktoren der betroffenen Veranstaltungen werden per E-Mail auf eine erreichte oder nicht erreichte Teilnehmerzahl aufmerksam gemacht.</li> </ol> <b>Alternativer Workflow:</b>	

n/a

**Fehlerbehandlung:**

n/a

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.10 Veranstaltung löschen

System:	GUI
Use Case:	Veranstaltung löschen
Beschreibung:	Hat eine Veranstaltung bereits statt gefunden oder wurde sie abgesagt, so kann diese gelöscht werden.
Akteur/Trigger:	Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Veranstaltung löschen
Extensions:	-
Includes:	Teilnehmer benachrichtigen
Dependencies:	-
Vorbedingungen:	-

**Standard Workflow:**

1. Der Instruktor, Akademie-Manager oder Administrator klickt auf den Menüpunkt **Veranstaltung löschen**.
2. Er wählt eine zu löschende Veranstaltung aus und bestätigt die Auswahl.
3. Die Software zeigt dem Benutzer eine Sicherheitsabfrage an um die Absicht des Benutzers nochmals abzufragen.
4. Das System löscht bei positiver Bestätigung die gewählte Veranstaltung aus der Datenbank und verschickt eine Benachrichtigungs-E-Mail.

**Alternativer Workflow:**

1. Verneint der Administrator die Bestätigung, so wird die Veranstaltung nicht gelöscht.

**Fehlerbehandlung:**

n/a

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.11 Anmeldungen einsehen

System:	GUI
Use Case:	Anmeldungen einsehen
Beschreibung:	<i>Anmeldungen einsehen</i> zeigt die Übersicht der aktuellen Anmeldungen zu einer ausgewählten Veranstaltung. Der Benutzer hat ebenso die Möglichkeit, die angezeigte Liste nach Excel zu exportieren, oder eine Druckansicht zu generieren.
Akteur/Trigger:	Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Anmeldungen einsehen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Instruktor, Akademie-Manager oder Administrator klickt auf den Menüpunkt <b>Anmeldungen einsehen</b>.</li> <li>2. Er wählt eine Veranstaltung aus der Liste der ihm zugeordneten Veranstaltungen aus und bestätigt die Auswahl.</li> <li>3. Das Programm zeigt dem Benutzer die aktuell zur gewählten Veranstaltung angemeldeten Personen.</li> </ol> <b>Alternativer Workflow:</b> <ol style="list-style-type: none"> <li>1. Nach Auswahl der Veranstaltung und Anzeigen der angemeldeten Personen kann der Benutzer die Liste per <b>Exportieren</b> nach Excel exportieren, oder mit dem Button <b>Drucken</b> eine Druckansicht generieren.</li> </ol> <b>Fehlerbehandlung:</b> n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.12 Vormerker einsehen

System:	GUI
Use Case:	Vormerker einsehen
Beschreibung:	Hier können die Merker einer ausgewählten Veranstaltung eingesehen werden.
Akteur/Trigger:	Instruktor, Akademie-Manager, Administrator
Haupt-Szenario:	Vormerker einsehen
Extensions:	-
Includes:	-
Dependencies:	-

Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Instruktor, Akademie-Manager oder Administrator klickt auf den Menüpunkt <b>Vormerker einsehen</b>.</li> <li>2. Er wählt eine aus der Liste der ihm zugeordneten Veranstaltungen aus und bestätigt die Auswahl.</li> <li>3. Das Programm zeigt dem Benutzer einen Vormerker auf den gewählten Kurs gesetzt haben.</li> </ol> <b>Alternativer Workflow:</b> n/a <b>Fehlerbehandlung:</b> n/a	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.13 Kategorien verwalten

Die Veranstaltungen einer Akademie können in unterschiedliche, auf diese beschränkte, Kategorien gruppiert werden. Diese dienen der logischen Gruppierung und der Sortierung bei der Anzeige im Veranstaltungs-Browser.

### 2.3.2.14 Kategorie anlegen

System:	GUI
Use Case:	Kategorie anlegen
Beschreibung:	Eine Kategorie muss unabhängig von einer Veranstaltung angelegt werden können. Während des Anlege-Vorgangs sollte es möglich sein, bereits existente Seminare hinzuzufügen.
Akteur/Trigger:	Akademie-Manager, Administrator
Haupt-Szenario:	Kategorie anlegen
Extensions:	<ul style="list-style-type: none"> <li>• Veranstaltung zu Kategorie hinzufügen</li> </ul>
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Akademie-Manager oder Administrator klickt auf den Menüpunkt <b>Kategorie anlegen</b>.</li> <li>2. Er füllt die Felder Titel und Beschreibung aus und wählt Veranstaltungen aus der Liste</li> </ol>	

- aus um diese der Kategorie zuzuordnen.
- Der Benutzer bestätigt das Anlegen der Kategorie mit **Anlegen**.
  - Das Programm speichert die neue Kategorie und ordnet die gewählten Veranstaltungen dieser zu.

**Alternativer Workflow:**

n/a

**Fehlerbehandlung:**

- Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder wird die Kategorie gespeichert.

Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.15 Kategorie pflegen

System:	GUI
Use Case:	Kategorie pflegen
Beschreibung:	Pflegen einer Kategorie soll die Bearbeitung der Attribute <i>Name</i> und <i>Beschreibung</i> , sowie der <i>assoziierten Veranstaltungen</i> umfassen.
Akteur/Trigger:	Akademie-Manager, Administrator
Haupt-Szenario:	Kategorie pflegen
Extensions:	-
Includes:	<ul style="list-style-type: none"> <li>Veranstaltung zu Kategorie hinzufügen</li> <li>Veranstaltung aus Kategorie löschen</li> </ul>
Dependencies:	-
Vorbedingungen:	-

**Standard Workflow:**

- Der Akademie-Manager oder Administrator klickt auf den Menüpunkt **Kategorie pflegen**.
- Er bearbeitet die Felder Titel und Beschreibung und wählt Veranstaltungen aus der Liste aus um diese der Kategorie zuzuordnen.
- Der Benutzer bestätigt das Anlegen der Kategorie mit **Bearbeiten**.
- Das Programm speichert die bearbeitete Kategorie ab und ordnet dieser die gewählten Veranstaltungen zu.

**Alternativer Workflow:**

n/a

**Fehlerbehandlung:**

1. Die Software macht den Benutzer im Falle nicht ausgefüllter Pflicht- oder falsch ausgefüllter Felder mit roten Umrandungen oder Hinweistexten auf Fehleingaben aufmerksam. Der Benutzer hat daraufhin die Möglichkeit die Eingaben zu korrigieren. Erst bei einwandfreier Befüllung der Felder kann die Kategorie erfolgreich bearbeitet werden.	
Nachbedingungen:	-
Spezifische Anforderungen:	-
Offene Punkte:	-

### 2.3.2.16 Kategorie löschen

System:	GUI
Use Case:	Kategorie löschen
Beschreibung:	Ist eine Kategorie nicht mehr aktuell, muss diese vom <i>Akademie-Manager</i> gelöscht werden können. Die darin enthaltenen Veranstaltungen sind nach der Löschung keiner anderen Kategorie zuzuordnen.
Akteur/Trigger:	Akademie-Manager, Administrator
Haupt-Szenario:	Kategorie löschen
Extensions:	-
Includes:	-
Dependencies:	-
Vorbedingungen:	-
<b>Standard Workflow:</b> <ol style="list-style-type: none"> <li>1. Der Akademie-Manager oder Administrator klickt auf den Menüpunkt <b>Kategorie löschen</b>.</li> <li>2. Er wählt eine zu löschende Kategorie aus und bestätigt die Auswahl.</li> <li>3. Die Software zeigt dem Benutzer eine Sicherheitsabfrage an um die Absicht des Benutzers nochmals abzufragen.</li> <li>4. Das System löscht bei positiver Bestätigung die gewählte Kategorie aus der Datenbank.</li> </ol>	
<b>Alternativer Workflow:</b>  n/a	
<b>Fehlerbehandlung:</b>  n/a	
Nachbedingungen:	Die Veranstaltungen die vor der Löschung der gelöschten Kategorie angehörten sind nun keiner Kategorie zugeordnet.
Spezifische Anforderungen:	-
Offene Punkte:	-

## 3 Projektplanung

### 3.1 Ablauf der Bearbeitungszeit

Das vorliegende Kapitel widmet sich der Projektplanung und beschreibt die Milestones des Software-Erstellungsprozesses. Es werden Zeitpunkte beschreiben, zu denen sowohl Quellcode als auch Dokumentation geliefert wird. In den Prozess sind folgende Personen involviert:

#### Betreuender Professor (Projektleitung)

Prof. Dr.-Ing. Ludwig Eckert

0171 / \*\*\*\*\*

[leckert@fh-sw.de](mailto:leckert@fh-sw.de)

#### Reviewer (Test und Abnahme)

Dipl.-Ing. (FH) Reiner Rottmann

0160 / \*\*\*\*\*

[rottmann@atix.de](mailto:rottmann@atix.de)

#### Diplomand (Design und Programmierung)

Christian Schäfer

0171 / \*\*\*\*\*

[christian.schaefer2@messe-muenchen.de](mailto:christian.schaefer2@messe-muenchen.de)

### 3.2 Milestonesplan

Der vorliegende *Milestones-Plan* sieht einen Bearbeitungs-Zeitraum von 3 Monaten ab dem 26.04.2006 vor und zeigt den aktuellen Status. Für das in der Liste aufgeführte Review werden jeweils 2 Tage veranschlagt, wobei die Rückmeldung in schriftlicher Form erwartet wird. Tauchen während des Reviews Fragen auf, so sind diese schriftlich, oder idealerweise per Telefon, zeitnah an den Diplomanten zu richten.

Nr.	Beschreibung	Termin	Lieferumfang	Status
1	Klärung der restlichen Eckpunkte <ul style="list-style-type: none"><li>• Welche Art der Dokumentation (nur Quellcode-Doku oder auch Prosa)?</li><li>• Welcher Umfang der Dokumentation der Implementierung?</li><li>• Ansprechpartner für Tests (Unittest, sowie iterative Funktionstests, da horizontales und vertikales Prototyping)? Ggf. Vorbereitung der</li></ul>	12.05. (6 Tage)	Schriftliches Feedback an den Diplomanden	Erledigt (10.06.)



	Test-Person auf die Tests (Anlesen von Grundkenntnissen etc.) <ul style="list-style-type: none"> <li>Ist Zeitplan für alle Beteiligten ok?</li> </ul>			
2	Fertigstellung der Dokumentation des Designs der Software	18.05. (6 Tage)	Dokumentation an Reviewer und Professor	Erledigt (19.05.)
3	<b>Review M 2</b>	20.05. (2 Tage)	Schriftliches Feedback an Diplomanden und Reviewer	Erledigt (19.06.)
4	Fertigstellung der Grundlagen-Komponenten gemäß Software-Design	28.05. (8 Tage)	Dokumentation, Quellcodes und Unit-Tests für Grundkomponenten als Installation auf dem Shared-Hosting-Account <a href="http://www.w3service.net">www.w3service.net</a> . Information an Professor und Reviewer.	Erledigt (29.05.)
5	<b>Review M 4</b>	30.05. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (19.06.)
6	Fertigstellung der GUI-Funktionalitäten <ul style="list-style-type: none"> <li>Öffentliches Menü + Login</li> <li>Backend-Menü</li> <li>Content-Bereich</li> <li>Kontaktformular</li> </ul>	04.06. (5 Tage)	Dokumentation und Quellcodes als Installation auf dem Shared-Hosting-Account <a href="http://www.w3service.net">www.w3service.net</a> . Information an Professor und Reviewer.	Erledigt (04.06.)
7	<b>Review M 6</b>	06.06. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (19.06.)
8	Fertigstellung der Funktionen zu einem Profil: <ul style="list-style-type: none"> <li>Registrierung</li> <li>Persönliches Profil bearbeiten</li> <li>Zugangsdaten bearbeiten</li> <li>Rechte zuweisen</li> <li>Passwort zurücksetzen</li> <li>Benutzer löschen</li> </ul>	10.06. (4 Tage)	Dokumentation und Quellcodes als Installation auf dem Shared-Hosting-Account <a href="http://www.w3service.net">www.w3service.net</a> . Information an Professor und Reviewer.	Erledigt (11.06.)
9	<b>Review M 8</b>	12.06. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (23.06.)
10	Fertigstellung der Basis-Funktionen zu einer Veranstaltung: <ul style="list-style-type: none"> <li>Kategorie anlegen</li> <li>Kategorie pflegen</li> <li>Kategorie löschen</li> <li>Veranstaltung anlegen (beinhaltet <i>Veranstaltung zu einer Kategorie hinzufügen</i>)</li> <li>Veranstaltung bearbeiten (beinhaltet <i>Kategorie zu Veranstaltung hinzufügen</i> <b>und</b> <i>Veranstaltung aus Kategorie löschen</i>)</li> <li>Veranstaltung anzeigen (Use Case <i>Details einer Veranstaltung ansehen</i>)</li> <li>Veranstaltung löschen</li> </ul>	22.06. (10 Tage)	Dokumentation und Quellcodes als Installation auf dem Shared-Hosting-Account <a href="http://www.w3service.net">www.w3service.net</a> . Information an Professor und Reviewer.	Erledigt (22.06.)
11	<b>Review M 10</b>	24.06. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (23.06.)
12	Fertigstellung der erweiterten Frontend-Funktionen zu einer Veranstaltung: <ul style="list-style-type: none"> <li>Veranstaltungsvorschau einsehen (Startseite)</li> <li>Veranstaltungskatalog browsen</li> <li>Veranstaltungskatalog durchsuchen</li> <li>Veranstaltung vormerken</li> </ul>	30.06. (6 Tage)	Dokumentation und Quellcodes als Installation auf dem Shared-Hosting-Account <a href="http://www.w3service.net">www.w3service.net</a> . Information an Professor und Reviewer.	Erledigt (30.06.)

	<ul style="list-style-type: none"> <li>• Zu einer Veranstaltung anmelden</li> <li>• Vormerker anzeigen</li> <li>• Anmeldungen anzeigen</li> <li>• Andere Akademieliste anzeigen</li> </ul>			
<b>13</b>	<b>Review M 12</b>	02.07. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (03.07.)
<b>14</b>	Fertigstellung der erweiterten Backend-Funktionen: <ul style="list-style-type: none"> <li>• Teilnehmer benachrichtigen</li> <li>• Anmeldungen einsehen</li> <li>• Vormerker einsehen</li> </ul>	05.07. (3 Tage)	Dokumentation und Quellcodes als Installation auf dem Shared-Hosting-Account <a href="http://www.w3service.net">www.w3service.net</a> . Information an Professor und Reviewer.	Erledigt (05.07.)
<b>15</b>	<b>Review M 14</b>	07.07. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (10.07)
<b>16</b>	Fertigstellung der automatisierten Benachrichtigungen: <ul style="list-style-type: none"> <li>• Teilnehmer benachrichtigen</li> <li>• Instruktor benachrichtigen</li> </ul>	12.07. (5 Tage)	Dokumentation und Quellcodes als Installation auf dem Shared-Hosting-Account <a href="http://www.w3service.net">www.w3service.net</a> . Information an Professor und Reviewer.	Erledigt (12.07.)
<b>17</b>	<b>Review M 16</b>	14.07. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (15.07.)
<b>18</b>	Nacharbeiten (optional)	24.07. (10 Tage)	Geänderte Funktionalitäten in der Form der eigentlichen Auslieferung incl. aller Punkte des Reviews. Info an Professor und Reviewer.	Erledigt (25.07.)
<b>19</b>	Abnahme Nacharbeiten (optional)	26.07. (2 Tage)	Schriftliches Feedback an Diplomanden und Professor.	Erledigt (29.07.)
<b>20</b>	Puffer	14 Tage	-	-
<b>21</b>	Fertigstellung Dokumentation	25.08. (12 Tage)	Dokumentation, komplette Quellcodes und Installationsanleitung an Professor und Reviewer.	Erledigt (25.08.)
<b>22</b>	Fertigstellung Diplomarbeit	04.09. (10 Tage)	Komplette Dokumentation und Quellcodes in digitaler und analoger Form sowie die Dokumentation in Papierform an Professor.	Erledigt (05.09.)

## 4 Design

Unter Design versteht sich der Entwurf der Software in zweierlei Hinsicht. Das Screen-Design beschreibt den visuellen Entwurf, verbildlicht Abläufe und Funktionen. Das Software-Design behandelt die Struktur und die in den Anforderungen beschriebenen Funktionalitäten.

### 4.1 Screen-Design

Wie bereits in der Einleitung angesprochen, ist Screen-Design eine grafische Entwurfsmethode, die die Oberfläche der Software beschreibt und in einer Serie von Bildschirm-Ausschnitten Funktionalitäten implizit aufzeigt. Die vorher dokumentierten Anforderungen können so an Hand von GUI-Ausschnitten vertiefend diskutiert und gegebenenfalls korrigiert werden.

Insbesondere dem Kunden können Screens<sup>1</sup> einen ersten Eindruck der zukünftigen Software vermitteln, sie bieten Gelegenheit bereits im Vorfeld Aussehen, Anordnung und Funktionen abzugleichen und eventuelle Design-Fehler noch vor dem Verfassen der Anwendung in Quelltext zu beheben. Daraus ergibt sich ein iterativer Reifeprozess, durch den sich die Idee und das Konzept der Software stetig verbessern sollen.

In diesem Kapitel wird exemplarisch das Screen-Design<sup>2</sup> und die zugehörige Diskussion im Bereich des **Frontends** dargestellt.

#### 4.1.1 Startseite

Der Startseiten-Screen beschreibt bereits viele Eigenschaften der grafischen Struktur. In Abbildung 4-1 lässt sich erkennen, dass die Seite im klassischen Zwei-Spalten-Design mit Header aufgebaut ist. Der Header-Bereich wird zur Präsentation des Logos im linken oberen Bereich verwendet, das je nach Akademie differiert.

Der mittlere obere Bereich kann zur ergänzenden Darstellung eines weiteren Logos oder ergänzender Stör-Elemente benutzt werden, wurde hier jedoch frei gelassen. Die Seite skaliert sowohl horizontal, mit der Größe des Browserfensters, als auch vertikal mit der Länge des Seiteninhalts. Dadurch wird eine größtmögliche Kompatibilität für die verschiedenen Bildschirm-Auflösungen der Besucher geschaffen.

Der linke Randbereich dient zur Aufnahme des Menüs des Front- und Backends. Weiterhin befindet sich dort die Login-Maske und – nach erfolgreichem Login – das Menü des angemeldeten Benutzers.

Der mittlere untere Bereich ist dem Content der Seite vorbehalten. Unter Content versteht sich jeglicher Inhalt der auf der Seite der Akademie präsentiert wird.

Im Content-Bereich der Startseite befinden sich, wie in Abbildung 4-1 zu erkennen, allgemeine Informationen zur Akademie und die in den Anforderungen definierte Veranstaltungs-Vorschau über die nächsten Veranstaltungen.

---

<sup>1</sup> Screen (engl.): Bildschirmauszug, Veranschaulichung des zukünftigen „Aussehens“ der Software (erhebt nicht den Anspruch auf Vollständigkeit)

<sup>2</sup> **Anmerkung:** Die hier aufgeführten Abbildungen werden von der endgültigen Implementierung der Software möglicherweise leicht abweichen, da die Screens nicht aus der letzten Diskussions-Phase des Entwurfs stammen.



Abb. 4-1: Screen Startseite

## 4.1.2 Veranstaltungs-Übersicht

Unter dem Menüpunkt *Veranstaltungen* verbirgt sich der Veranstaltungs-Browser. Ein Besucher kann, wie auf den Abbildungen 4-2, 4-3 und 4-4 angedeutet, die Veranstaltungen einer Akademie nach den Kriterien *Kategorie*, *Datum* und *Alphabet* geordnet anzeigen lassen. Es wird ihm angeboten das Sortier-Kriterium per Formular auszuwählen.

### 4.1.2.1 Übersicht nach Kategorie

In der Übersicht nach Kategorien wird ein Auswahlménü angezeigt, mit dem der Benutzer die Veranstaltungen der jeweils ausgewählten Kategorien anzeigen lassen kann.



Abb. 4-2: Screen Übersicht Veranstaltungen nach Kategorie

#### 4.1.2.2 Übersicht nach Datum

Auch hier werden die Veranstaltungen in einer Liste aufgeführt. Sollten pro Monat sehr viele Einträge gezeigt werden, so ergibt sich eine sehr lange Liste von Seminaren. Abhilfe kann hier der Einsatz eines Pagers schaffen, der von jeder Kategorie immer nur eine bestimmte Anzahl von Veranstaltungen anzeigt. Dieses, die Usability verbessernde Feature, wird jedoch erst für spätere Versionen aufgenommen, da darauf in den Anforderungen kein Wert gelegt wurde.

Unter dem Link *Details* kann die Detailansicht der Veranstaltung aufgerufen werden.



Abb. 4-3: Screen Übersicht Veranstaltung nach Datum

### 4.1.2.3 Übersicht nach Alphabet

Als weiteres Ordnungskriterium kann das Alphabet eingesetzt werden. Zur Bestimmung der Veranstaltungs-Reihenfolge wird der erste Buchstabe des Titels verwendet. Um die Übersichtlichkeit zu verbessern, werden nur diejenigen Buchstaben des Alphabets im Auswahlmenü angezeigt, für die Veranstaltungen vorhanden sind.

Die Auswahl einer Veranstaltung funktioniert damit wie folgt:

Ein Besucher wählt das Ordnungs-Kriterium *Alphabet* aus und klickt auf OK. Anschließend wählt er einen Buchstaben und bestätigt die Auswahl. Daraufhin wird ihm eine Veranstaltungsliste angezeigt, welche diejenige Seminare enthält, deren Titel mit dem gewählten Buchstaben beginnen. Der Benutzer kann nun innerhalb der Liste navigieren oder den Buchstaben ändern.



Abb. 4-4: Screen Übersicht Veranstaltung nach Alphabet

### 4.1.3 Suche


Abbildung 4-5 zeigt den Aufbau des Suchformulars, mit dem der Besucher nach beliebigen *Titeln*, *Dozenten*, *Zeiträumen* oder *Beschreibungen* suchen kann. Das Suchfeld *Titel* soll gegen das Veranstaltungs-Attribut Titel geprüft werden, *Dozenten* gegen das Attribut Dozent, *Datum* gegen das Kurs-Datum/den Kurs-Zeitraum und *Beschreibung* gegen alle Inhalte der Kurz- und Detailbeschreibung eines Seminars.

Möchte ein Teilnehmer Kurse über „Linux“ angezeigt bekommen, so erzielt er Treffer in dem er unter Zuhilfenahme von Wildcards nach „LINUX\*“ oder auch „\*LINUX\*“ sucht.

Die einzelnen Suchkriterien werden mit ODER verknüpft, so dass das Treffervolumen maximiert wird. Das bedeutet beispielsweise, dass zwei eindeutige Treffer aus dem Bereich Titel mit drei eindeutigen Treffern aus dem Bereich Datum, zu fünf eindeutigen Treffern in der Ergebnis-Liste verbunden werden. Abbildung 4-6 zeigt das Verhalten der GUI, wenn der Benutzer den Button Suchen anklickt.

Als zusätzliche Funktion kann es darüber hinaus sinnvoll sein, eine Relevanz-Ermittlung durchzuführen und die Liste nach der resultierenden Relevanz der Treffer absteigend zu sortieren. Dies ist jedoch nicht Teil der Anforderungen und könnte in einer weiteren Version implementiert werden.





Akademie der

...training on the job

...get some more knowledge

Startseite

Veranstaltungen

Suche

Registrierung

Hilfe

Kontakt

Impressum

Weitere Akademien

Login:

Tragen Sie hier Ihre Benutzerkennung ein um sich zu Ihrem persönlichen Profil anzumelden.

Anmelden

## Suche

Nutzen Sie die Suche um zu Ihren Anforderungen eine passende Veranstaltung zu finden. Die Suche verbindet die Einzelergebnisse jeweils mit ODER und stellt diese in der Ergebnis-Liste dar.

**Tipp:** Verwenden Sie "?" um ein einzelnes Zeichen und "\*" um beliebige Zeichen in den Such-Feldern für Titel, Dozent oder Beschreibung zu adressieren.

**Titel:**

**Dozent:**

**Datum:**

01

01

2004

bis

01

01

2004

**Beschreibung:**

Suchen

Abb. 4-5: Screen Suchformular



Akademie der

...training on the job

...get some more knowledge

Startseite

Veranstaltungen

Suche

Registrierung

Hilfe

Kontakt

Impressum

Weitere Akademien

Login:

Tragen Sie hier Ihre Benutzerkennung ein um sich zu Ihrem persönlichen Profil anzumelden.

Anmelden

## Suche

Nutzen Sie die Suche um zu Ihren Anforderungen eine passende Veranstaltung zu finden. Die Suche verbindet die Einzelergebnisse jeweils mit ODER und stellt diese in der Ergebnis-Liste dar.

**Tipp:** Verwenden Sie "?" um ein einzelnes Zeichen und "\*" um beliebige Zeichen in den Such-Feldern für Titel, Dozent oder Beschreibung zu adressieren.

**Titel:**

**Dozent:**

**Datum:**

01

01

2004

bis

01

01

2004

**Beschreibung:**

Suchen

Es wurden folgende Veranstaltungen gemäß Ihren Eingaben gefunden:

**LINUX for Beginner 2**

Dozent(en):  
Hans Meier,

Beschreibung:  
Der Kurs möchte eine Einführung in LINUX geben. Es werden Grundfertigkeiten mit dem Umgang der Kommandozeile erlernt. Besprochen werden auch unterschiedliche Techniken der Verwaltung. U.a. Perl-Programmierung hilfreich.

Abb. 4-6: Screen Suchergebnisse der eigenen Akademie



#### 4.1.4 Informationen

Abbildung 4-7 zeigt schemenhaft die Darstellung von allgemeinen Inhalten. Eine Content-Seite kann Texte und Bilder sowie Links beinhalten. Die Gestaltung der Informations- und Hilfe-Seiten wird nach Fertigstellung der Applikation nochmals auf Vollständigkeit geprüft. Der hier abgebildete Screen ist deshalb bewusst unvollständig.

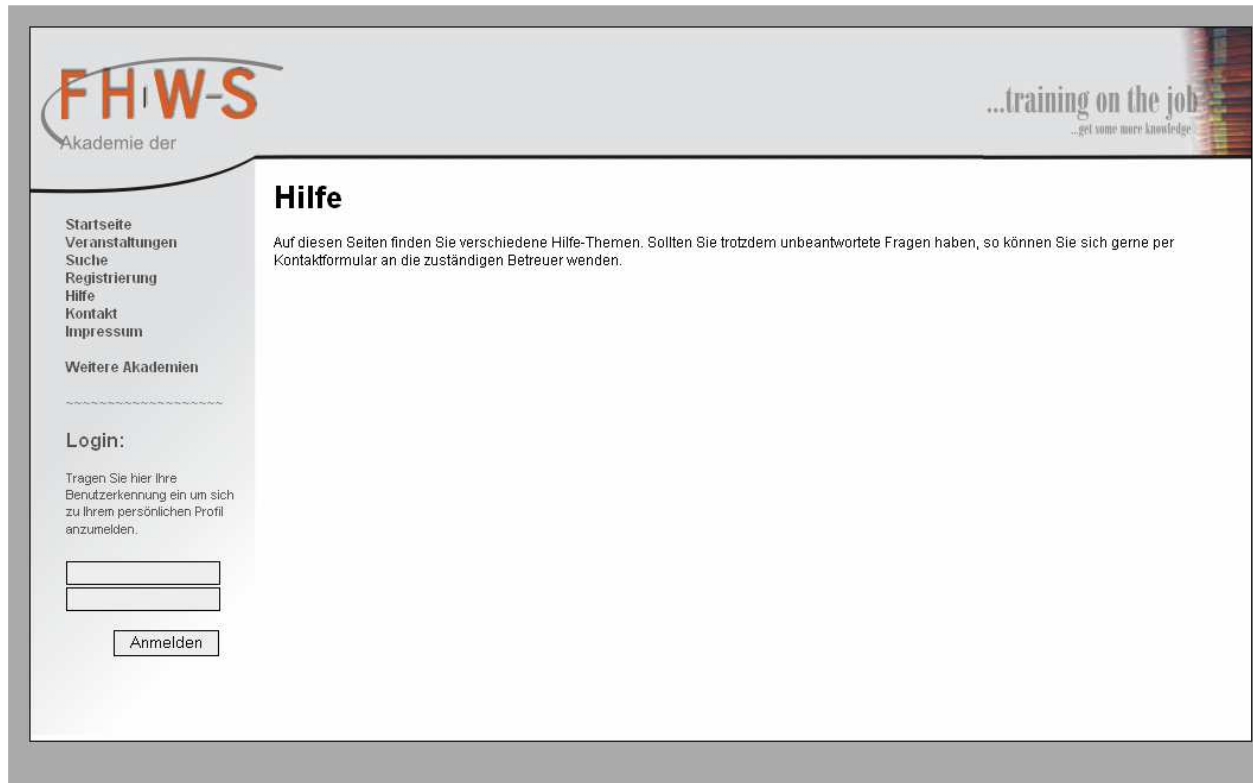


Abb. 4-7: Screen allgemeine Informationen

#### 4.1.5 Persönliches Profil

Das persönliche Profil umfasst mehrere Funktionen, die sich analog den Anforderungen in die Bereiche Frontend und Backend gliedern. Auf Grund der Rollen-Struktur und der sich daraus ergebenden Rechte unterscheidet sich das Menü eines *Teilnehmers* von dem eines *Administrators*, bzw. *Akademie-Managers* oder *Instruktors*. Da in diesem Kapitel nur die Screens des Frontends diskutiert werden, zeigen Abbildung 4-8 bis einschließlich 4-10 die Funktionen des persönlichen Profils eines *Teilnehmers*. Sobald ein Besucher eingeloggt ist, erscheint sein persönlicher Bereich unterhalb des Webseitenmenüs. So kann er auf **jeder** Unterseite des Weiterbildungsportals zwischen den Menüpunkten *Persönliche Daten bearbeiten*, *Zugangsdaten bearbeiten*, *Anmeldungen verwalten*, *Vormerker verwalten* und *Logout* wählen.



Abb. 4-8: Screen Startseite nach erfolgreichen Login als Teilnehmer

Darüber hinaus stehen ihm noch weitere Funktionen im Zusammenhang mit dem Veranstaltungs-Browser zur Verfügung. Wie in Abbildung 4-8 zu erkennen ist, erscheinen zusätzlich zu dem Button *Details* noch *Anmelden* und *Vormerken*. Es handelt sich dabei um die Funktionen *Anmelden* und *Vormerken*, die in allen Veranstaltungsansichten verfügbar sind. Klickt ein Besucher auf *Anmelden*, so meldet er sich zu dieser Veranstaltung an, klickt er auf *Vormerken*, so fügt er dadurch die Veranstaltung zu seinem Vormerker-Korb hinzu.

Die Abbildungen 4-9 und 4-10 zeigen die Ansichten „Anmeldungs-Korb“ und „Vormerker-Korb“. Diese werden dem Besucher nach Ausführen eines Anmelde- oder Vormerken-Vorgangs oder durch explizites Auswählen der Menüpunkte „Anmeldungen verwalten“ oder „Vormerker verwalten“ angezeigt.

Der in Abbildung 4-9 dargestellte „Anmeldungs-Korb“ zeigt eine Auflistung von Veranstaltungen, zu denen sich ein Benutzer angemeldet hat. Mit den Optionen *Details* und *Abmelden* kann er sich die Veranstaltungs-Details ansehen oder von der Veranstaltung abmelden.

Der „Vormerker-Korb“ verfügt über die Buttons *Details* und *Löschen*. *Details* verhält sich analog zum Button *Details* im „Anmeldungs-Korb“ und *Löschen* entfernt die Veranstaltung aus dem „Vormerker-Korb“.

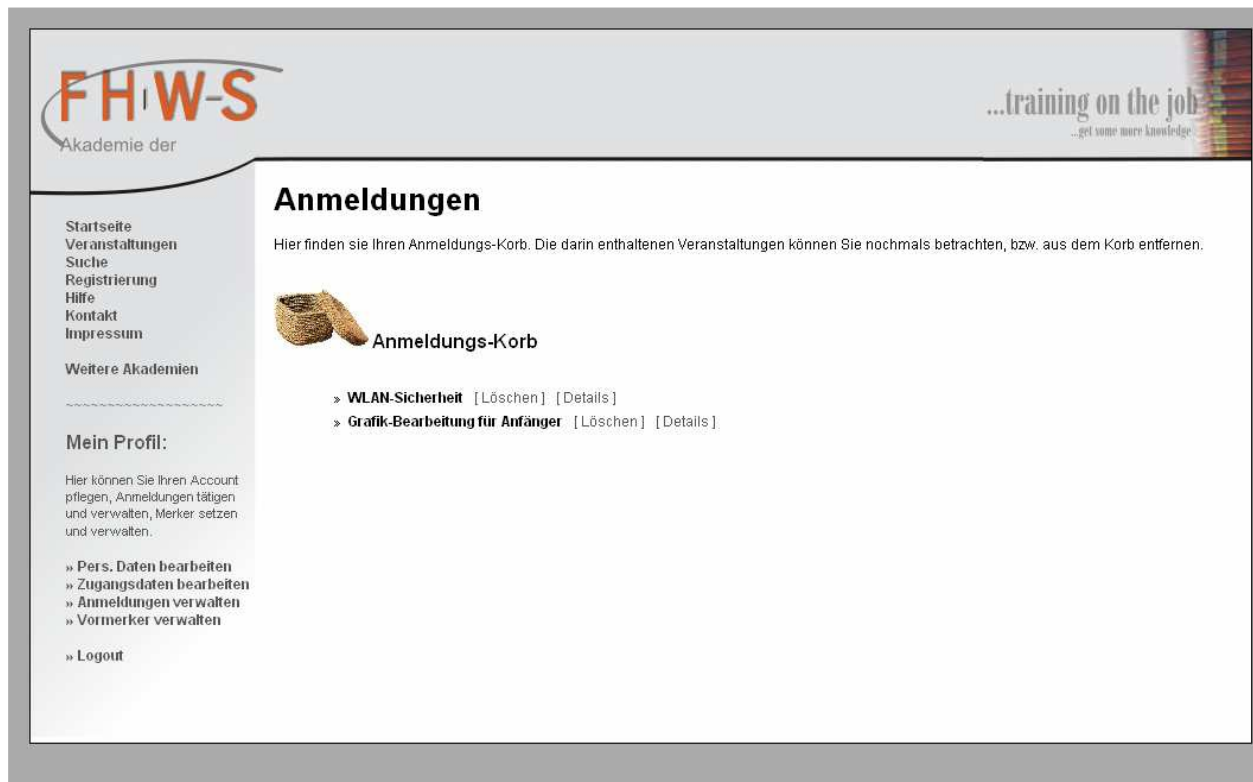


Abb. 4-9: Screen Anmeldungs-Korb bearbeiten

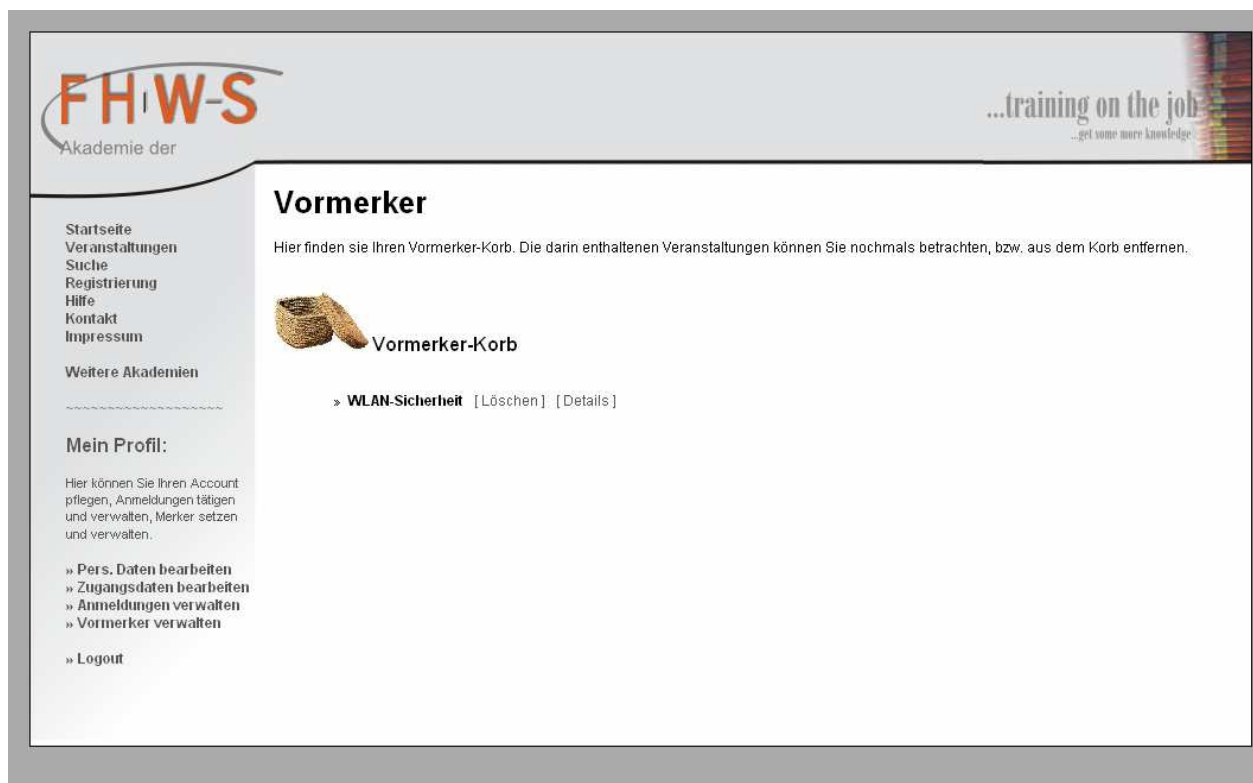


Abb. 4-10: Screen Merker-Korb bearbeiten

Der Screen in Abbildung 4-11 zeigt das Bearbeitungsformular, mit dem ein Besucher seine persönlichen Daten abändern kann. Es enthält alle Datenfelder, die auch bei der Registrierung angezeigt werden. Die Zugangsdaten kann der Besucher über einen in Abbildung 4-12 gezeigten Dialog unter dem Menüpunkt „Zugangsdaten bearbeiten“ geändert werden.

**FHWS**  
Akademie der

...training on the job  
...get some more knowledge

**Persönliches Profil**

Hier können Sie Ihr persönliches Profil bearbeiten.

Persönliche Daten:

Anrede: Herr  
 Titel:  
 Vorname: Christian  
 Name: Schäfer  
 Geburtsdatum: 12 / 07 / 1981

Memo:  
 Mein Account bei Akademie FH-SW

Firmen-Adresse:

Straße: Am Messesee  
 PLZ, Ort: 81823 München  
 Telefon: 089 / 949 21 985  
 Mobil: 0171/1414357

**Abb. 4-11: Screen Profildaten bearbeiten**

**FHWS**  
Akademie der

...training on the job  
...get some more knowledge

**Zugangsdaten**

Hier können Sie Ihre Zugangsdaten bearbeiten.

**Achtung:** Beachten Sie bitte, dass Sie sich nach der Änderung wieder neu einloggen müssen!

Benutzer: jschaefer  
 Passwort:

Ändern

**Abb. 4-12: Bearbeiten der Zugangsdaten**

## 4.2 Software-Design

Das folgende Kapitel befasst sich mit dem Design der Software. Unter Design versteht sich dabei die Art und Weise, wie die Applikation aufgebaut ist und welche Lösungsansätze bzw.

Vorgehensweisen Anwendung finden. Es wird auf die softwaretechnischen Rahmenbedingungen und die Architektur eingegangen. Dabei wird auf einige bereits in der einschlägigen Fachliteratur bekannte Pattern zurückgegriffen.

### 4.2.1 Allgemein

Um dem oben geäußerten Anspruch hinsichtlich moderner Implementierung und Design zu genügen, wird in diesem Projekt auf **objektorientierte Programmierung** in Verbindung mit **UML** (Unified Modeling Language) gesetzt. UML ermöglicht es ein System nahezu vollständig quellcodeunabhängig zu beschreiben, zudem erleichtert es die Kommunikation zwischen einzelnen Entwicklern, da diese dieselbe „Sprache“ sprechen. Gute UML-Tools – in diesem Projekt kommt der *Enterprise Architect 6.1* der Firma Sparx Systems zum Einsatz – bieten die Möglichkeit der direkten Code-Erzeugung aus den UML-Diagrammen. Umgekehrt ermöglicht der Enterprise Architect *Reverse Engineering*. Das bedeutet, dass bereits verfasster Code nachträglich in UML dokumentiert werden kann. Zusätzlich dazu findet das Software-Dokumentationswerkzeug *doxygen*<sup>1</sup> Verwendung. Mit diesem kann eine zusammenhängende Quellcode-Dokumentation im JAVADoc-Stil angefertigt werden. Dies ist vor allem bei der Implementierung hilfreich, falls diese auf bereits vorhandene APIs aufsetzt.

### 4.2.2 Drei-Schicht-Architektur

In der Literatur wird im Zusammenhang mit IT-Systemen und Software-Entwicklung häufig von Schicht-Modellen gesprochen. Es gibt sehr unterschiedliche Schichten-Modelle, die in Komplexität, Schichtenanzahl und Anwendungsbereich differieren. In dieser Applikation wird ein für die Software-Entwicklung oft verwendetes Modell eingesetzt – das Drei-Schicht-Modell. Dieses beschreibt die Trennung der Programm-Logik in die Bereiche Präsentation, Domäne und Datenquelle. Dabei definieren sich diese Schichten wie folgt:



Abb. 4-14: Drei-Schicht-Architektur

Die **Präsentation** stellt Dienste bereit, zeigt Informationen an und verarbeitet Benutzeranfragen wie Seitenaufrufe oder Mausklicks.

Die **Domäne**, bzw. die Business-Schicht, kapselt die eigentliche Programm- oder Geschäfts-Logik. Das umfasst Datenmanipulation, Bearbeitung von Benutzereingaben und Steuerung des Datenzugriffs. In einer Webapplikation kann dies beispielsweise das Versenden einer

<sup>1</sup> Weitere Informationen zu doxygen unter [4].

Bestätigungs-E-Mail, bzw. Koordination der Speicherung der aus der Präsentations-Schicht gewonnenen Daten sein.

Die **Datenquelle** übernimmt die Kommunikation mit Datenbanken, Messaging-Systemen und anderen Diensten, die sich um die Bereitstellung von Daten kümmern. Je nach Datenhaltungsmodell oder Typ des Datenhaltungssystems unterscheiden sie sich in Komplexität und Ausprägung.

Der Vorteil des Einsatzes eines Schichtenmodells liegt auf der Hand:

- Jede Schicht kann unabhängig von der anderen entwickelt und getestet werden. Somit lassen sich die Abhängigkeiten minimieren, mehr Standardisierung und Transparenz schaffen.
- Höhere Schichten können sich der Dienste niedrigerer Schichten bedienen und deren Funktionalität nach oben hin erweitern. Im Netzwerkumfeld bedient sich beispielsweise ein HTTP-Dienst der Implementierung des TCP/IP-Protokolls.
- Beim Austausch einer Schicht können die Übrigen ohne Veränderung weiter arbeiten, da jede eine wohl definierte gemeinsame Schnittstelle besitzt. Es kann beispielsweise zur Datenübertragung via TCP/IP-Pakete sowohl ein Kupfer-Kabel, als auch ein Lichtwellenleiter verwendet werden.

In diesem Projekt wird die Unterteilung in Präsentation, Business-Logik und Daten-Schicht noch zusätzlich unterstützt, indem die Code-Dateien der jeweiligen Schicht in eigene Ordner der allgemeinen Namespace-Struktur abgelegt werden.

### 4.2.3 Datenmodell und Abstraktion

Das Kapitel Datenmodell und Abstraktion beschäftigt sich mit der Datenschicht der Anwendung. Um die mit den Anforderungen definierten Daten und Beziehungen speichern zu können, bedarf es einiger Überlegungen. Im Zuge einer transparenten Datenhaltung ist es zudem erstrebenswert, die eigentliche physikalische Datenhaltung gegen die Anwendung soweit zu abstrahieren, dass es keine Rolle spielt, woher die Business-Schicht ihre Daten bezieht. In letzter Konsequenz kann das bedeuten, dass ein Gästebuch-Manager, der die Business-Logik kapselt, sowohl mit einer MySQL- als auch mit einer Flat-File-Datenbank kommunizieren können muss.

#### 4.2.3.1 Abstraktion der Datenhaltung

Obige Forderung setzt voraus, dass die bisher beschriebene Drei-Schicht-Architektur um weitere Abstraktionsschichten ergänzt werden muss. Damit ändert sich auch die Art der Implementierung der Datenschicht an einigen Punkten.

Wie in Abbildung 4-15 zu sehen ist, gliedert sich die Datenschicht nun in die Bereiche *Datenhaltung*, *Datenzugriffs-API* und *Daten-Mapper*.

- (1) *Datenhaltung* steht für die physikalische *Datenhaltung* in einer Datenbank. Diese Schicht „kennt“ die Datenbank, in der die Daten gespeichert sind, das Persistenz-Modell und die Zugriffsmethoden.
- (2) Die *Datenzugriffs-API* abstrahiert die *Datenhaltung* in soweit, als dass die darüber liegende *Mapper-Schicht* in standardisierter Weise auf diese zugreifen kann.
- (3) Der *Daten-Mapper* abstrahiert nun die in der *Datenhaltung* abgelegten Daten und mappt diese in das Objektmodell der Anwendung. Sollten mehrere Applikationen mit unterschiedlichen Datenmodellen auf der *Datenhaltung* arbeiten, so „kennt“ der Mapper das Datenmodell jeder Anwendung. Das ermöglicht beispielsweise, gleiche oder ähnliche Daten in verschiedene Anwendungen wieder zu verwenden. In diesem Fall spricht man vom so genannten Domain-Objektmodell<sup>1</sup>.

Diese Struktur hat den Vorteil, dass sowohl die physikalisch Datenhaltung als auch das Persistenz-Modell der Datenhaltung einer Applikation, beispielsweise im Fall einer Migration auf eine neu Datenbank-Version, ausgetauscht werden könnten, ohne dass die Applikation geändert werden muss. Lediglich das Anpassen des Mappers und der Datenhaltung, die sich auf Grund der Migration geändert hat, ist erforderlich.

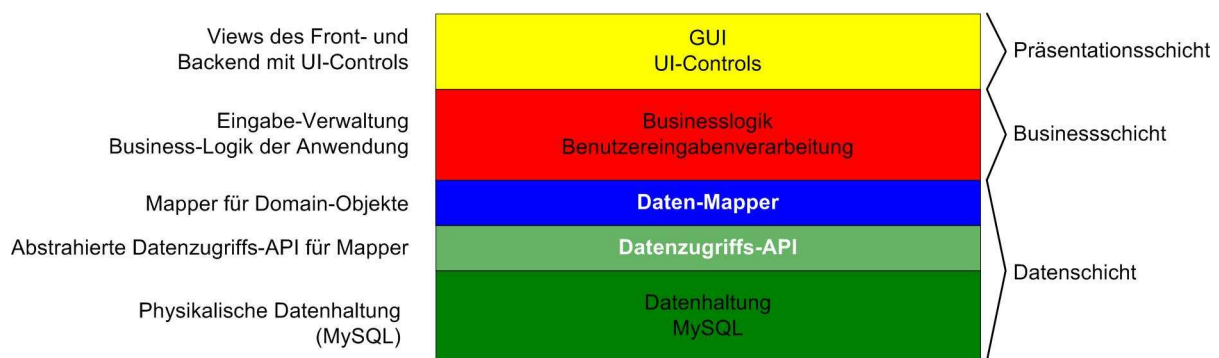


Abb. 4-15: Schichtenmodell der Anwendung

#### 4.2.3.2 Objekt- und Datenmodell

Das in Abbildung 4-16 skizzierte UML-Diagramm zeigt den Aufbau des Objektmodells der Applikation. Es beschreibt diejenigen Objekte und Beziehungen, die die Anwendung, insbesondere die Business-Schicht, benötigt, um die spezifizierten Daten und Funktionen abbilden zu können.

- (1) Das Objekt *oObjekt* ist ein abstraktes Objekt, das alle gemeinsamen Attribute und Methoden implementiert. Die übrigen Domain-Objekte erben von diesem.
- (2) *oAkademie* ist ein zentrales Objekt. Dieses beinhaltet alle für eine Akademie relevanten Informationen. Eine Akademie komponiert sowohl Kategorien (*oKategorie*) als auch Veranstaltungen (*oVeranstaltung*). Die Beziehung symbolisiert

<sup>1</sup> Objekte, die sich je nach Anwendungsfall aus einer Auswahl von Attributen und Objekten des gemeinsamen Objektmodells der Anwendung zusammensetzen und von dieser manipuliert werden. Das Domain-Objekt, oder die Domain-Objekte werden ebenfalls als Schnittstelle zwischen den unterschiedlichen Applikations-Schichten eingesetzt. Weitere Informationen siehe [1].



die direkte Zugehörigkeit einer Veranstaltung und einer Kategorie zu einer Akademie. Die Assoziation zu Person kennzeichnet die indirekte Zugehörigkeit einer Person zu einer Akademie. Die Beziehung ist jedoch von schwächerer Qualität (Assoziation), da eine natürliche Person keiner Akademie gehören kann.

- (3) *oVeranstaltung* trägt die Attribute einer Veranstaltung. Eine Veranstaltung kennt seine Kategorie und ist dieser dadurch assoziativ zugeordnet. Das komplexe Attribut Dozenten ist eine Liste von Personen (*oPerson*), die einer Veranstaltung Dozenten zuordnet.
- (4) *oKategorie* beinhaltet die Daten einer Kategorie.
- (5) *oPerson* repräsentiert eine natürliche Person und trägt alle personenbezogenen Attribute. Da eine Person mehrere Adressen, wie Privatadresse oder Firmenadresse besitzen kann, hat *oPerson* die komplexen Attribute *PrivatAdresse* und *FirmenAdresse*, die jeweils ein Objekt des Typs Adresse (*oAdresse*) beherbergen. Zudem sind einer Person die komplexen Attribute *VormerkerListe* und *AnmeldungsListe* zugeordnet. Jedes dieser Attribute beinhaltet eine Liste von Objekten des Typs *oVeranstaltung*. Durch diese Listen wird signalisiert, für welche Veranstaltungen eine Anmeldung bzw. ein Vormerker existiert.
- (6) *oAdresse* hält die Adressdaten einer Person und ist unterhalb von Person komponiert, da eine Adresse eine starke Bindung zu dieser besitzt.

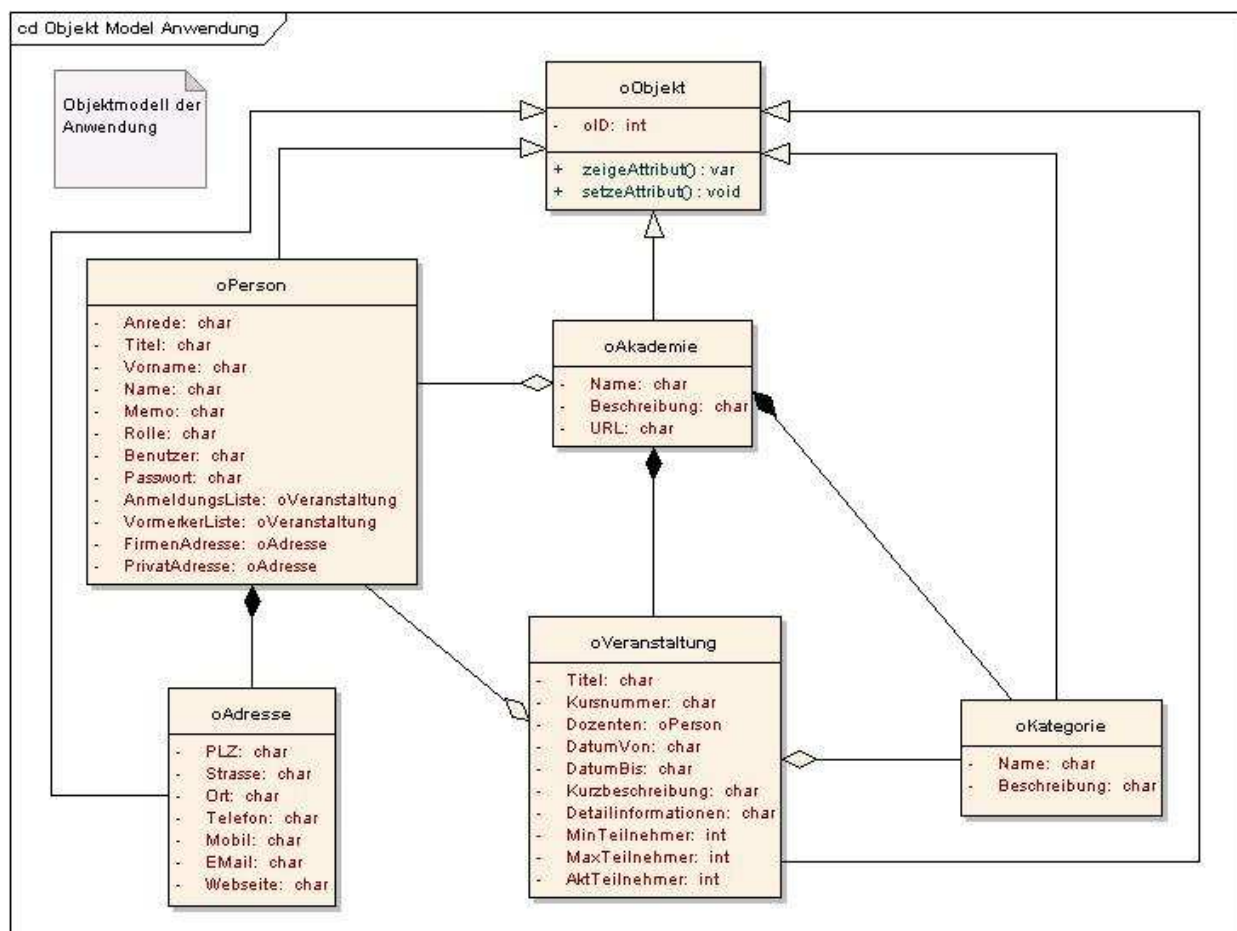


Abb. 4-16: Objektmodell der Applikation

Da die oben aufgeführten Objekte und deren Beziehungen nicht direkt und objektorientiert in einer MySQL-Datenbank gespeichert werden können, existiert ein weiteres Modell: das Modell der Datenhaltung. Das Entitäts-Modell beschreibt die Struktur der physikalischen



Datenspeicherung in der relationalen Datenbank. Abbildung 4-17 zeigt auf, wie die in der Anwendung generierten Objekte und deren Beziehungen in der relationalen Datenbank abgelegt werden.

- (1) Die aufgeführten Blöcke repräsentieren physikalische Datenbank-Tabellen mit unterschiedlichen Spalten und Spalteneigenschaften. Die Spalten sind der Übersichtlichkeit halber nicht vollständig aufgeführt, da es in dieser Darstellung lediglich um die Konzeption der physikalischen Datenspeicherung geht. Die bidirektionalen Pfeile deuten an, dass zwischen den angegebenen Blöcken datenhaltungstechnische Beziehungen über Fremdschlüssel existieren. Diese Art der Referenzierung wird in der Literatur auch oft unter dem Begriff „foreign key constraints“ gelistet. Eine Beziehung zwischen Person und Akademie wird dadurch aufgebaut, dass in der Tabelle *ass\_akademie\_person* ein Datensatz mit den beiden Primärschlüsseln der Tabellen *ent\_akademie* und *ent\_person* gespeichert wird. Die Beziehung ist somit bidirektional auflösbar.
- (2) Das Präfix „ent\_“ kennzeichnet, dass in der vorliegenden Tabelle die Attribute eines Objekts wie Name und Vorname gespeichert werden. Tabellen, die mit „ass\_“ beginnen, beinhalten Beziehungsdaten des Typs Assoziation. Kompositionen werden in Tabellen mit dem Präfix „comp\_“ gespeichert.
- (3) Es handelt sich bei dem in Abbildung 4-17 gezeigten Modell, um ein teilweise denormalisiertes<sup>1</sup> Datenbank-Design, da nicht alle Speicherstrukturen generisch angelegt sind. Dies bedeutet auch, dass die Struktur eines Objekts in vielen Punkten der des Entitäts-Modells ähnelt. So fällt bei den Objekten *Akademie* und *Veranstaltung* ins Auge, dass die Spalten der Tabellen mit den Attributen der Objekte in sehr hohem Maß übereinstimmen. Die Denormalisierung greift hingegen bei der Speicherung von Beziehungsinformationen. Jede Assoziation und Komposition wird, wie oben bereits beschrieben, in separaten Tabellen abgelegt, kann somit bidirektional aufgelöst und bei Bedarf hinzugeladen werden. Eine derartige Vorgehensweise vereinfacht das Mapping der in der relationalen Datenbank gespeicherten Daten in die Objekte der Anwendung (siehe 4.2.3.3.) und ist performancetechnisch besser zu handhaben.
- (4) Die aufgeführten Tabellen beinhalten folgende Daten bzw. Beziehungen:
  - a. *ent\_adresse*: Datensätze von Adressen (Privat- und Firmenadressen).
  - b. *comp\_person\_privat\_adr*: Komposition einer Privat-Adresse unter einer Person.
  - c. *ent\_akademie*: Datensätze von Akademien.
  - d. *comp\_akademie\_veranstaltung*: Komposition von Veranstaltungen unter einer Akademie.
  - e. *comp\_person\_firmen\_adr*: Komposition einer Firmen-Adresse unter einer Person.
  - f. *ent\_person*: Datensätze von Personen.
  - g. *ass\_akademie\_person*: Assoziation einer Akademie mit einer Person.
  - h. *comp\_akademie\_kategorie*: Komposition einer Kategorie unterhalb einer Akademie.
  - i. *ass\_person\_merker*: Assoziation einer Veranstaltung zu einer Person (Person hat Veranstaltung in seiner Vormerker-Liste).
  - j. *ass\_person\_veranstaltung*: Assoziation einer Veranstaltung zu einer Person (Person ist Dozent einer Veranstaltung).

---

<sup>1</sup> „denormalisieren“: Vollständiges zerlegen eines Objekts in seine Attribute. Dies hat eine hochgenerische Struktur zur Folge, die keine einfache Abfragbarkeit der Struktur zulässt.

- k. *ass\_person\_anmeldung*: Assoziation einer Veranstaltung zu einer Person (Person ist zu einer Veranstaltung angemeldet).
- l. *ent\_veranstaltung*: Datensätze von Veranstaltungen.
- m. *ass\_kategorie\_veranstaltung*: Assoziation einer Veranstaltung zu einer Kategorie.
- n. *ent\_kategorie*: Datensätze von Kategorien.

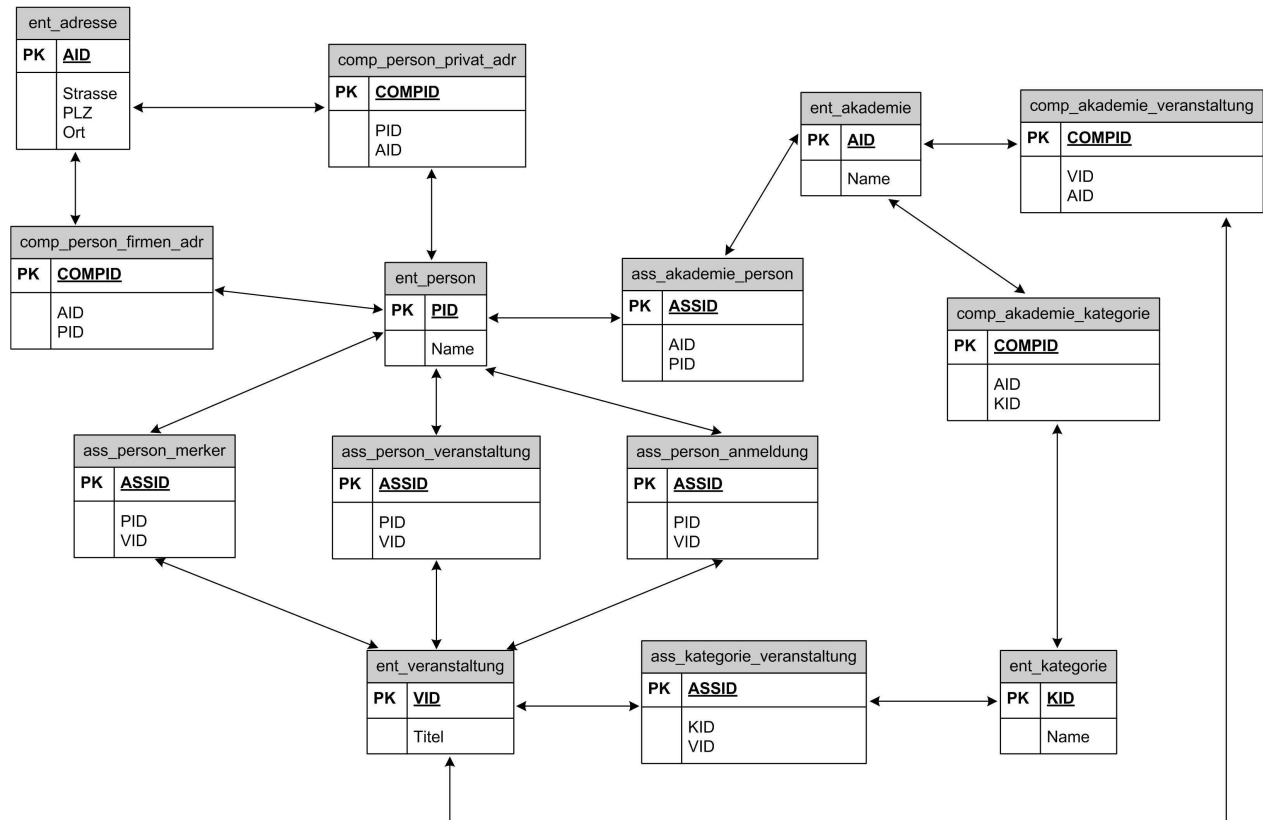


Abb. 4-17: Physikalisches Datenmodell

### 4.2.3.3 Data-Mapper

Durch objektorientierte Implementierung der Anwendung und Speicherung in einer relationalen Datenbank, ergibt sich die Notwendigkeit einer Instanz, die zwischen diesen beiden Welten vermittelt: der Data-Mapper. Dieser erfüllt in diesem Projekt zwei Aufgaben: Zum einen „übersetzt“ und „rückübersetzt“ er relationale Datensätze in Objekte, zum anderen abstrahiert er die Anwendung von der Datenhaltung.

Eine Recherche im Internet wird zeigen, dass es bereits eine Reihe von OR-Mappern gibt, die für diesen Anwendungsfall nur noch konfiguriert werden müssten. Als Beispiele seien hier Hibernate<sup>1</sup> und NDO<sup>2</sup> aufgeführt. Da es sich bei dem hier zu implementierenden Data-Mapper um einen, im Vergleich zu Hibernate, sehr einfachen Data-Mapper handelt, wird für diese Applikation ein eigener spezieller Data-Mapper verfasst. Dieser stellt der Business-Schicht standardisierte und gekapselte Methoden zur Verfügung, um Objekte und Strukturen zu laden, zu manipulieren oder zu löschen.

<sup>1</sup> Nähere Informationen zu Hibernate unter [www.hibernate.org](http://www.hibernate.org)

<sup>2</sup> Nähere Informationen zu NDO unter [www.netdataobjects.de](http://www.netdataobjects.de)

#### 4.2.4 Business-Logik

Die Business-Logik der Anwendung umfasst hauptsächlich die Kommunikation mit dem Data-Mapper, um der Präsentations-Schicht Daten zur Darstellung zu liefern. Viel Bedeutung kommt der Business-Logik in den Use-Cases *Teilnehmer benachrichtigen* und *Suche* zu. Bei *Teilnehmer benachrichtigen* werden die unterschiedlichen Teilnehmer einer Veranstaltung und Personen, die die Veranstaltung vorgemerkt haben, per Mail benachrichtigt. Bei der *Suche* müssen die Suchergebnisse der unterschiedlichen Teile konsolidiert und ausgewertet werden.

#### 4.2.5 Präsentations-Schicht

Um die objektorientierte Implementierung auch in der Präsentations-Schicht fortsetzen zu können, wird ein allgemeingültiger Page-Controller eingesetzt. Dieser ist nach dem Vorbild des Model-View-Controller- und Page-Controller-Pattern implementiert. Er abstrahiert eine Webseite in einen Objektbaum mit verschiedenen Knoten, wobei ein Knoten jeweils durch ein Model, einen View und einen Document-Controller repräsentiert wird. Der View wird nach dem Composite-Pattern aufgebaut und behandelt. Den Root-Knoten bildet eine *Page*, die ein initiales *Document* komponiert. Letzteres komponiert seinerseits eine unterschiedliche Hierarchie von „Documents“. Die Hierarchie-Tiefe und Komplexität bestimmt der Programmierer dadurch, dass er in den Templates weitere *Documents* importiert. Somit ist es möglich, dynamisch Module und damit Funktionalitäten hinzuzufügen. Knoten des Typs *Document* repräsentieren und kapseln Inhalte, die beim Aufrufen einer Seite in den Objektbaum eingehängt werden. Bei der Ausgabe (Transformation) der Seite wird die Baum-Struktur rekursiv durchlaufen, die in den „Documents“ angegebenen Document-Controller ausgeführt und die dadurch erzeugten Inhalte an die nächst höhere Instanz zurückgegeben.

Der in Abbildung 4-18 aufgeführte Objekt-Baum ist ein Beispiel für den Baum, der bei Aufruf der Startseite aufgebaut und bearbeitet wird. Die hier präsentierte Grundstruktur findet sich in vielen Bereichen der Webseite wieder, da nur die unter dem Knoten *Content:Document* befindliche Struktur je nach Inhalt oder eingebundenem PlugIn variieren.

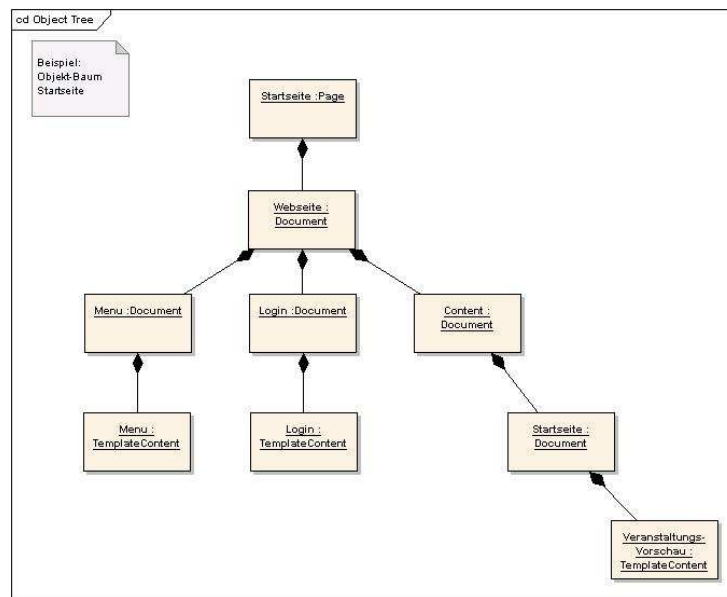


Abb. 4-18.: Objektbaum Page-Controller

## 4.2.6 PlugIn-Mechanismus

Unter einem PlugIn versteht der Software-Entwickler ein Ergänzungs- oder Zusatz-Modul, das in ein Softwareprodukt ohne Änderung des Kerns nach festgelegten Schnittstellen „eingeklinkt“ werden kann. Der im Folgenden skizzierte PlugIn-Mechanismus geht davon aus, dass alle in der hier beschriebenen Software enthaltenen Programmteile PlugIns sind und es einen globalen Rahmen gibt, der diese einbinden kann.

Zu diesem Zweck ruft der DocumentController, der zum Aufbau des Content-Bereiches der Seite zuständig ist, dasjenige Modul auf, welches mit dem Parameter „Seite“ in der URL benannt wird. Im konkreten Fall wird das Modul *Registrierung* dann aufgerufen, wenn die URL die Form <http://www.akademie1.tld/?Seite=Registrierung> besitzt. Technisch gesehen wird das Modul *Registrierung* in den Objekt-Baum der Seite eingebunden und die im Document definierten DocumentController beim Ausgeben der Seite ausgeführt. Die Implementierung und der Test des PlugIns können somit unabhängig des Rahmens erfolgen, da dieses wiederum durch eine eigene Objekt-Struktur definiert ist. Da der PlugIn-Handler auf Präsentations-Ebene greift, nutzt jedes PlugIn eine gemeinsame Business- und Daten-Schicht die allgemein gültige Funktionalitäten bereitstellen.

## 5 System-Architektur

Das vorliegende Kapitel beschreibt den schematischen Aufbau der Netzwerk- und Servertopologie des Webhosting-Anbieters „../\.. Bytecamp GmbH“, die mit der Auslieferung der Applikation betraut ist. Die Übersicht beruht jedoch weitestgehend auf der Erfahrung des Autors mit Webserver-Infrastrukturen und einigen Angaben über die Server-Landschaft, die aus den offiziellen Angaben (FAQs) des Hosters entnommen wurden. Die Abbildung ist daher unter qualitativen Gesichtspunkten zu bewerten. Nähere Details wurden von Bytecamp aus Sicherheitsgründen nicht herausgegeben.

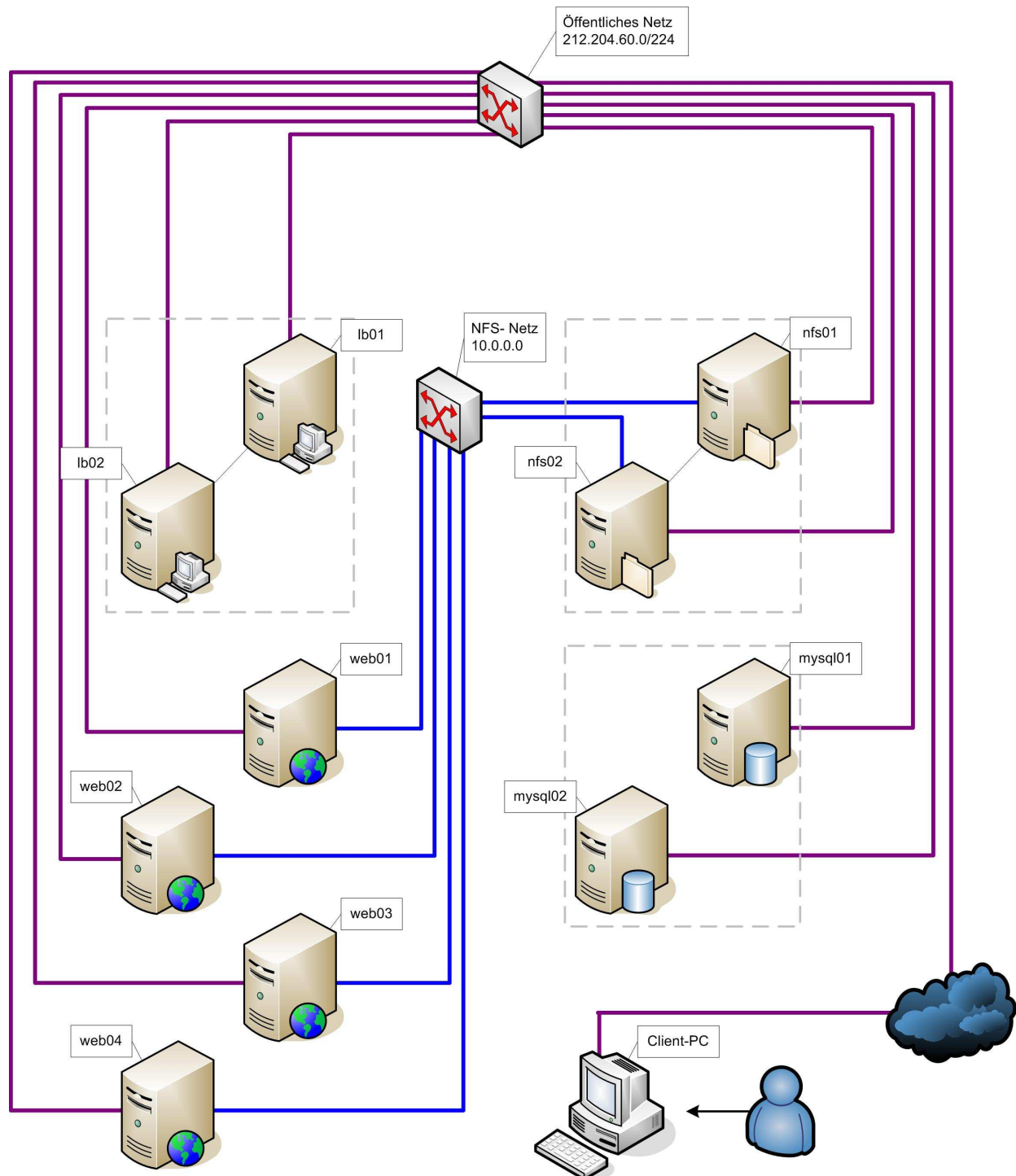


Abb. 5-1: Netzwerk-Topologie

## 5.1 Webserver

Die mit web01 bis web04 bezeichneten Server erfüllen die Aufgabe eines Webserver. Sie nehmen die Anfragen der Besucher entgegen, generieren die gewünschten Seiten und liefern diese wieder an den Kunden aus.

Die Webserver sind auf Standard-i386-Hardware aufgesetzt, da diese Art von Hardware im Markt etabliert ist und dadurch die Verfügbarkeit von Ersatz- oder Zusatz-Hardware zu jeder Zeit und zu einem günstigen Preis sichergestellt werden kann.

Die i386-Plattform wird zudem von sehr vielen Betriebssystem-Typen mit Treibern und Tools unterstützt. Auf den bei Bytecamp eingesetzten Webservern wurde das auf dem UNIX-Derivat BSD fußende freie Betriebssystem FreeBSD verwendet. Es vereint die Stabilität eines UNIX-Betriebssystems mit der Kostenneutralität eines LINUX-Betriebssystems.

Als Webserver-Software findet der Apache-httpd-Server Verwendung. Dies ist ein von der Apache Software Foundation (siehe [7]) entwickelter freier Webserver, der auf allen UNIX/LINUX- und Windows-Betriebssystemen verfügbar ist. Durch die modulare Struktur können auf sehr einfache Weise Erweiterungs-Module eingebunden werden um die Funktionalität des Webserver zu erweitern.

Die Möglichkeit, dynamische Webseiten und Webapplikationen zu betreiben, erhält der Apache Webserver durch die Installation des Skriptsprachen-Interpreters PHP. Die für PHP installierten Zusatzmodule und deren Konfiguration kann unter

<http://web01.bytecamp.net/phpinfo.php>

für PHP4 und unter

<http://web01.bytecamp.net/phpinfo.php5>

für PHP5 eingesehen werden. In der verfassten Software werden vor allem Funktionen der PHP-Standard-API, wie beispielsweise String-, Session- und Mail-Funktionen, darüber hinaus Teile der gd-Library zur Bildbearbeitung und die mysql-Library für die Kommunikation mit der MySQL-Datenbank eingesetzt.

Eine Besonderheit der vorliegenden PHP-Konfiguration ist zudem, dass der PHP-Skript-Interpreter nicht als ISAPI-Modul des Apache-Webserver, sondern als CGI-Programm eingebunden ist. Das bedeutet, dass das Parsen und Ausführen des PHP-Codes nicht direkt im Apache-Prozess, sondern in einem eigenen PHP-Prozess stattfindet. Diese Art der Installation wird in einigen HOWTO's und Foreneinträgen als unsicherer und langsamer gegenüber dem ISAPI-Modul bezeichnet, birgt jedoch den Vorteil, dass die reinen PHP-Prozesse sauberer von anderen getrennt werden können, wodurch das Monitoring und die Überwachung eines Webserver vereinfacht wird.

## 5.2 Loadbalancer

Um die Verfügbarkeit der Webserver zu maximieren, werden die eingehenden Anfragen von einem Loadbalancer nach bestimmten Algorithmen auf die verschiedenen Webserver verteilt. Somit ist sichergestellt, dass bei Ausfall eines Webserver der http-Service nach Außen weiter

verfügbar ist und dass bei steigender Zugriffslast weitere Webserver transparent hinzugefügt werden können.

Erfahrungsgemäß wird der *weighted-round-robin-Algorithmus* mit zusätzlichen *health-checks* gegen verschiedene Leistungs-Indikatoren der Webserver eingesetzt. Aussagekräftige Leistungs-Werte sind netzwerktechnische Erreichbarkeit, Erreichbarkeit des http-Servers, CPU, Load und RAM. Mit dem genannten Verteilungsprinzip können die eingesetzten Serverressourcen gleichmäßig ausgelastet und ausgefallene Knoten aus dem Verteilungsverbund automatisiert herausgelöst werden.

### 5.3 NFS-Server

Im bei Bytecamp aufgebauten Szenario wird auf das *scale-out*-Prinzip gesetzt. Das bedeutet, dass einer steigenden Zugriffszahl mit weiteren Webserver-Ressourcen entgegnet wird. Um diese Art der Skalierung nutzen zu können muss eine Virtualisierung des Plattenspeichers vorhanden sein. Diese wird mit Hilfe eines NFS-Servers bewerkstelligt, der zentrale Filesysteme zur Verfügung stellt.

Der NFS-Server hält dazu die für alle Webserver gemeinsam genutzten Verzeichnisse vor und exportiert diese über ein privates Netzwerk. Auf den Webservern sind diese Filesysteme als *home/* und *var/* im LINUX-Verzeichnisbaum eingebunden. Änderungen, die über einen Server, beispielsweise einem FTP-Server, in einem Bereich der Ordner-Struktur vollzogen werden, sind für alle Server, die diesen Export nutzen, verfügbar.

Erst durch diese Tatsache ist das unter 5.2 oben beschriebene Loadbalancing sinnvoll möglich, da alle statischen Informationen, wie Applikations- und HTML-Dateien, sowie dynamische Inhalte, wie Session-Daten, allen Webservern zur gleichen Zeit bekannt sind. In der Literatur und in einschlägigen Informationsquellen wird dieses Szenario auch als *stateless loadbalancing* und *transparent failover* beschreiben.

Der NFS-Fileserver wird durch den Einsatz als zentrale Komponente in der skizzierten Infrastruktur auch zum *single point of failure* (SPOF) und muss daher redundant ausgelegt werden.

### 5.4 MySQL-Server

Zur Speicherung der Daten der Anwendung wird ein MySQL-Server auf einem dedizierten Server betrieben. Dieser kann von den Webservern über den DNS-Namen [mysql.bytecamp.net](http://mysql.bytecamp.net) angesprochen werden.

Um die bereits mehrfach erwähnte Ausfallsicherheit und Verfügbarkeit zu gewährleisten, muss auch in diesem Bereich auf Clustering gesetzt werden. MySQL bietet dazu eigene Cluster-Lösungen wie NDB. Es besteht jedoch auch die Möglichkeit, den MySQL-Server durch externe Cluster-Manager, wie die *Red Hat Cluster Suite*, clusterfähig zu machen. Das zuletzt genannte Paket ermöglicht den Betrieb eines 2- oder Mehr-Knoten Clusters im *active/passive*-Modus unter der Voraussetzung, dass ein für die Maschinen gemeinsam nutzbarer Datenpool, wie ein NFS-Share, existiert. Die Virtualisierung des Dienstes wird durch den Cluster-Manager übernommen, der zu diesem Zweck eine virtuelle IP-Adresse zur Verfügung stellt. Diese dient den Applikationen als virtueller Service-Name. Fällt einer der Server aus, so übernimmt der andere den Service und die Anwendungen können ohne Unterbrechung weiterarbeiten.

## 5.5 Netzwerk

In Abbildung 5-1 ist zu erkennen, dass das Netzwerk der Server-Infrastruktur aus unterschiedlichen IP-Netzwerken besteht. Weitere, wie Stromnetze, SCSI- oder FiberChannel-Speichernetzwerke, finden der Übersichtlichkeit wegen keine Beachtung.

Eines der beiden eingezeichneten Netze wird für die Auslieferung der NFS-Verzeichnisse an die Webserver verwendet. Grund hierfür sind Sicherheit und Performance. Sicherheit, um den bereits nach Außen abgesicherten NFS-Service physikalisch von Außen unerreichbar zu machen, Performance, weil ungünstige Latenzzeiten im NFS-Filesystem-Zugriff die Gesamtperformance verschlechtern. Die Verwendung eines dedizierten Netzwerks kann eine negative Latenzzeiten-Beeinflussung durch andere Services ausschließen und es kann eine gleich bleibend hohe und stabile Netzwerkkommunikation und somit NFS-Performance garantiert werden.

Das zweite Netzwerk ist ein öffentliches IP-Netz, das mit anderen Knotenpunkten des Internets über diverse Routen verbunden ist. Über diesen Weg erreichen die Anfragen der Besucher die Webserver und Antworten werden von diesen über den gleichen Weg an den Kunden zurückgesendet. Die Anbindung der Server innerhalb des Rechenzentrums ist redundant ausgelegt, was bedeutet, dass mehrere Netzwerkstrecken zur Backbone des Providers existieren. Dadurch ist die uneingeschränkte Verfügbarkeit und Erreichbarkeit der Infrastruktur auf Netzwerkebene sichergestellt.

## 5.6 Internet

Die Wolke ist ein allgemeingültiges Symbol für das Internet. Netzwerkarchitektonisch handelt es sich um einen Zusammenschluss vieler kleiner und größerer Subnetze über Router, wobei die Subnetze wiederum unterschiedliche Ausprägungen haben können.

## 5.7 Client

Der Client-PC symbolisiert stellvertretend alle Besucher der Webseite. Wie in der Darstellung zu erkennen ist, wird vorausgesetzt, dass der potentielle Kunde über einen ISP mit dem Internet verbunden ist.



## 5.8 Weitere Komponenten

In der Skizze unbeachtet waren bisher die FTP-, Mail- und Auswertungsdienste. Der FTP-Dienst wird im Fall des Hosters Bytecamp dazu verwendet, Dateien und Ordner von der lokalen Entwicklungs-Maschine auf die Webserver, bzw. den zentralen NFS-Server, zu transferieren. Um innerhalb der Anwendung E-Mails versenden zu können, ist ein SMTP-Server installiert. Dieser kann mit den in PHP vorhandenen Funktionen bedient werden. Des Weiteren werden zur Auswertung von Webseitenzugriffen die Apache-Log-Dateien ausgewertet und Zugriffs-Statistiken generiert. Es ist in der Regel üblich für die Ausführung derartiger Dienste einen eigenen Server zur Verfügung zu stellen.

## 6 Implementierung

Das vorliegende Kapitel beschreibt Teile der Entwicklungsphase, geht auf Problemstellungen und architektonische Elemente der Software ein. Die Implementierung selbst orientiert sich stark am evolutionären Prototyping-Verfahren. Dieser in [1], [2] und [3] beschriebene Entwicklungs-Workflow sieht vor, eine funktionslose Oberfläche mit der Software-Basis zu erstellen, die bereits die GUI der fertigen Software abbildet, und die Programmteile iterativ hinzuzufügen. Dabei wurde darauf geachtet, dass die durch Hinzufügen eines Moduls entstandene Software stets in ihrer Gesamtheit lauffähig war. Die neu hinzugekommene Funktionalität kann somit auch in Abhängigkeit der Bisherigen sehr effektiv getestet werden. Die Reihenfolge der Implementierung der PlugIns kann dem Zeitplan in Kapitel 3.1 entnommen werden.

Die Vorteile des beschriebenen Verfahrens kommen jedoch innerhalb der Implementierungs-Phase der vorliegenden Arbeit nicht vollständig zu Geltung, da die Codierung in PHP-Code von nur einem Entwickler vorgenommen wurde. Üblicherweise werden die Erstellung, die Tests und die Einbindung unterschiedlicher Module oder auch verschiedener Schichten, auf mehrere Entwickler verteilt um eine höhere Effizienz zu generieren.

### 6.1 Ausführungsplan der Software

Abbildung 6-1 zeigt den systematischen Schichtenaufbau der Software-Architektur. Die Skizze beinhaltet sowohl Systemprogramme als auch Top-Level-Services wie Apache, MySQL oder PHP als auch „eigene“ Software-Komponenten, die im weiteren Text näher erläutert werden. Die aufgeführten Ziffern und Buchstaben bezeichnen wichtige Stationen oder Kommunikationswege bei der Ausführung der Software.

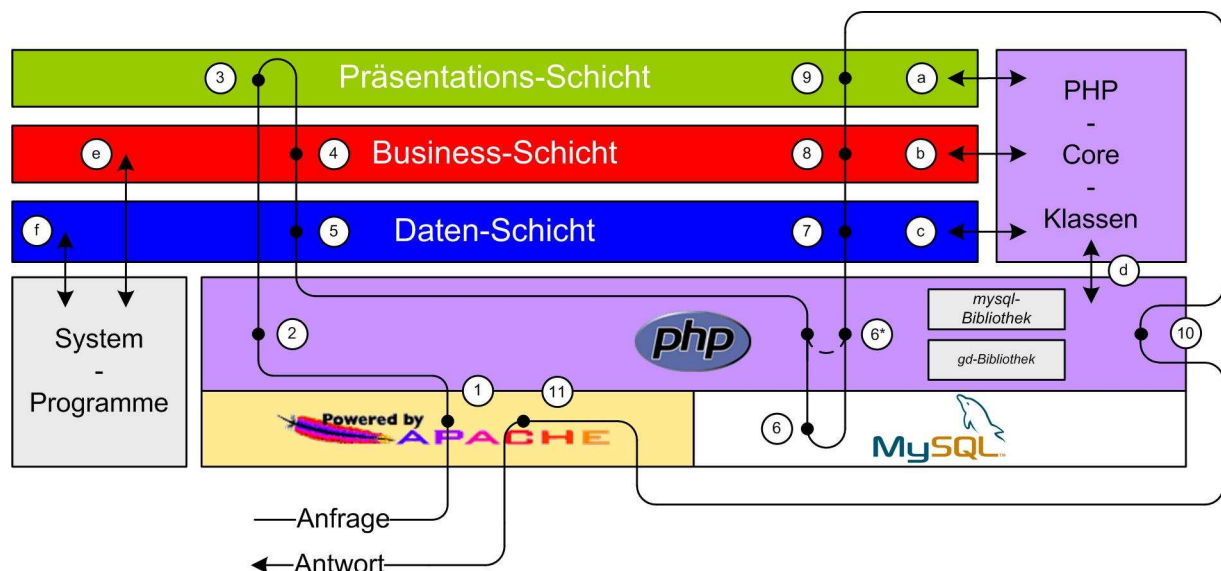


Abb. 6-1: Software-Architektur

Der oben eingezeichnete Kommunikationspfad zeigt den logischen Ablaufplan, der bei Aufruf der Software durchlaufen wird. Um den Verlauf beispielhaft erläutern zu können wird im Folgenden die Startseite der Akademie zur Verdeutlichung hinzugezogen.

- (1) Im ersten Schritt wird der Aufruf der Startseite (<http://www.w3service.net/akademie1/?Seite=Startseite>) vom Webserver entgegen genommen und die Datei *index.php* im Verzeichnis *akademie1/* geöffnet. Auf Grund der im letzten Kapitel erläuterten Webserver-Konfiguration wird diese Datei an den PHP-Prozess zur Verarbeitung weitergegeben.
- (2) Der PHP-Interpreter analysiert die Quelltext-Datei, bindet die notwendigen Dateien ein, compiliert den Quellcode und beginnt das Programm auszuführen.
- (3) Im Code, bzw. genauer der *index.php*, wird zuerst die Präsentations-Schicht initialisiert und das Template der Webseite geladen. Da dieses wiederum weitere Elemente wie Menü, Login- und Content-Bereich enthält, werden die im Haupt-Template angegebenen additiven Templates in den, vom PageController aufgebauten Objektbaum eingebunden. Um die Ansicht komplett generieren zu können, wird in vielen Bereichen der Seite die Business-Schicht benötigt. Dies ist auf der Startseite für die Darstellung der Veranstaltungsvorschau der Fall. Dazu initialisiert die Präsentations-Schicht die Business-Schicht und fordert diese auf, die benötigten Daten zu laden. Mit Hilfe des Singleton-Patterns wird die Business-Schicht für alle Teile der Präsentations-Schicht nur einmal pro Ausführung geladen um Speicher-Ressourcen zu schonen.
- (4) Im unter (3) beschriebenen Fall initialisiert die Business-Schicht wiederum die Datenschicht und fragt die von der Präsentations-Schicht gewünschten Daten an.
- (5) Um die Daten von der Datenbank zu beziehen lädt die Datenschicht die Datenbank-Treiber-Schicht und fragt bei dieser die benötigten Daten an.
- (6) Die Datenbank liefert die von der PHP-Core-Schicht angefragten Daten an die selbige zurück und die Treiber-Schicht reicht die relationalen Ergebnis-Datensätze an die Daten-Schicht der Anwendung zurück.
- (7) Die Datenschicht mappt die aus der Datenbank-Core-Schicht gewonnenen relationalen Veranstaltungsdaten in Domain-Objekte der Anwendung und gibt die Liste der Veranstaltungs-Objekte an die Business-Schicht zurück.
- (8) Die Business-Schicht loggt den Business-Prozess über eine dedizierte Logging-Methode, falls die gerade ausgeführte Aktion innerhalb des für die Anwendung konfigurierten Log-Levels liegt und reicht die Daten an die Präsentations-Schicht zurück. Üblicherweise übernimmt die Business-Schicht jedoch weitere Aufgaben wie die Sortierung oder Filterung von Daten und Koordination von Workflows.
- (9) Die Präsentations-Schicht stellt die von der Business-Schicht erhaltene Veranstaltungs-Liste in HTML dar.
- (10) Die nach der vollständigen Ausführung der Präsentations-Schicht generierte HTML-Seite wird vom PHP-Interpreter an den Apache-Webserver-Prozess zurückgegeben.
- (11) Der Webserver sendet die vom PHP-Prozess generierte HTML-Seite an den Besucher zurück.

Die mit den Buchstaben a bis f bezeichneten Verbindungen und Kommunikationswege werden im nächsten Kapitel näher behandelt.

## 6.2 Software-Komponenten

Unter 6.1 wurden bereits viele der in Abbildung 6-1 dargestellten Schichten und Komponenten angerissen. Dieses Kapitel geht auf jede der Komponenten näher ein, beschreibt deren Funktion und Bedeutung innerhalb der Applikation.

### 6.2.1 Apache-HTTP-Server

Der Funktionsblock „*Powered by Apache*“ steht für den Apache-HTTP-Server. Dieser Server-Dienst stellt die Basis für die HTTP-Kommunikation zwischen Client und Server bereit, ist also für die Entgegennahme von Client-Anfragen, deren Abarbeitung und Rücksendung der Ausgabe verantwortlich. Im Falle der im Rahmen der vorliegenden Arbeit zu verfassenden Software fungiert dieser zusätzlich als Schnittstelle zum Erweiterungs-Modul PHP. Weitere Informationen zur Architektur, zu Features, Konfigurations-Möglichkeiten und Anwendungs-Beispielen finden sich unter [7].

### 6.2.2 PHP

„*PHP*“ repräsentiert die Funktion des PHP-Interpreters. Dieser führt vom Webserver übergebene PHP-Programme aus und liefert die Ergebnisse – zumeist HTML-Dokumente – an den Webserver zurück. Dabei wird der aus einer Datei gelesene PHP-Code in Bytecode kompiliert und anschließend ausgeführt.

PHP beinhaltet einen Satz von Standard-Bibliotheken, es sind jedoch weitere so genannte PCRL-Bibliotheken verfügbar, die den Funktionsumfang von PHP erweitern. In der obigen Abbildung sind die beiden für die Weiterbildungs-Akademie wesentlichen Bibliotheken, die MySQL- und die Image-Erweiterung, symbolisch aufgeführt. Es ist ebenso möglich eigene Funktionalitäten als Module für PHP in C zu implementieren und diese zur Laufzeit zu laden und zu verwenden, erhöht jedoch den Aufwand der Programmierung erheblich.

Weitere Informationen zur Architektur, zu Features und Anwendungs-Beispielen können unter [8] nachgelesen werden. Die genannte Webseite beinhaltet eine vollständige Funktions- und Konfigurations-Referenz, sowie zahlreiche Anwendungs-Beispiele und Tipps anderer Programmierer sowie ein Forum.

### 6.2.3 MySQL

Der Block „*MySQL*“ symbolisiert die freie Datenbank MySQL. Sie bietet auf sehr einfache Art und Weise die Möglichkeit Daten zu speichern und in Applikationen abzugreifen. Die Einbindung der MySQL-Datenbank gestaltet sich sehr einfach, da diese durch den hohen

Verbreitungsgrad als Web-Datenbank von einer Vielzahl an Skript- und Programmiersprachen unterstützt wird. Der Entwicklungsgrad der Datenbank-Engine ist jedoch bereits soweit fortgeschritten, dass diese auch für komplexere Projekte eingesetzt werden kann. In der vorliegenden Software wird die Datenbank zur zentralen Speicherung aller programmrelevanten Daten verwendet. Weitere Informationen zur Architektur, zu Features und Anwendungs-Beispielen können unter [9] nachgeschlagen werden.

## 6.2.4 System-Programme

Unter System-Programmen versteht der Autor betriebssystemnahe Programme, die in der Software von höher liegenden Schichten der Anwendung genutzt werden. Dazu werden System-Kommandos ausgeführt und die Ergebnisse in der PHP-Anwendung weiter verarbeitet, wie beispielsweise das Auslesen des Host-Namens eines Servers über das System-Kommando *hostname*.

## 6.2.5 PHP-Core-Klassen

Es sei an dieser Stelle nochmals erwähnt, dass die Software, wie in den Anforderungen definiert, komplett objektorientiert implementiert ist. Aus diesem Grund sind auch die Grundfunktionalitäten in Klassen abgefasst. Die PHP-Core-Klassen bilden neben dem Apache-Webserver, dem PHP-Interpreter, der MySQL-Datenbank und den System-Programmen die Grundlage vieler Funktionen der Software. Alle hier erwähnten Klassen zusammen bilden ein Framework, das Lösungen zu Standard-Problemen einer Webapplikation bieten. Die PHP-Core-Klassen erzeugen zudem eine weitere Abstraktions-Schicht innerhalb der Software, bzw. den einzelnen Software-Schichten, wie in Abbildung 6-3 zu erkennen ist. Dort wird beschrieben, dass jede Schicht bei Bedarf auf Funktionen einer jeden darunter liegende Schicht zurückgreifen kann. Die Instanz der Datenschicht greift beispielsweise auf die die Instanz der Klasse *MySQLHandler* zurück um gekapselt auf die Datenbank zuzugreifen.

Beispiele für den Einsatz von unterschiedlichen PHP-Core-Klassen in den Schichten der Anwendung lassen sich folgender Tabelle entnehmen:

Schicht	Klasse der Applikation	Core-Klassen-Namespace	Core-Klassen-Name
Präsentations-Schicht	<i>content_v1_controller</i>	core::pagecontroller	<i>Page, Document, Content, Template</i>
Präsentations-Schicht	<i>veranstaltungsbrowser_v1_controller</i>	tools::datetime	<i>dateTimeManager</i>
Business-Schicht	<i>wvManager</i>	core::inihandler	<i>iniHandler</i>
Business-Schicht	<i>wvManager</i>	tools::image	<i>imageManager</i>
Daten-Schicht	<i>wvDataMapper</i>	core::database	<i>MySQLHandler</i>
Daten-Schicht	<i>basicDataMapper</i>	tools::string	<i>stringEncryptor</i>

**Tab. 6-a: Einsatz von Core-Klassen in der Applikation**

Eine detaillierte Beschreibung der genannten Klassen kann der Quellcode-Dokumentation der Begleit-CD entnommen werden. Tabelle 6-b führt die vorhandenen Dokumentationen auf:

Namespace	Name der Hilfedatei	Beschreibung
core	<i>apps_core.chm</i>	Dokumentation der PHP-Core-Klassen im core-Namespace
tools	<i>apps_tools.chm</i>	Dokumentation der PHP-Core-Klassen im tools-Namespace
modules	<i>apps_modules_weiterbildungsveranstaltung.chm</i>	Dokumentation der zusätzlichen Module
sites::weiterbildungsveranstaltung	<i>apps_sites_weiterbildungsveranstaltung.chm</i>	Dokumentation der Programmdateien für das Weiterbildungs-Portal

Tab. 6-b: Quellcode-Dokumentation der Software

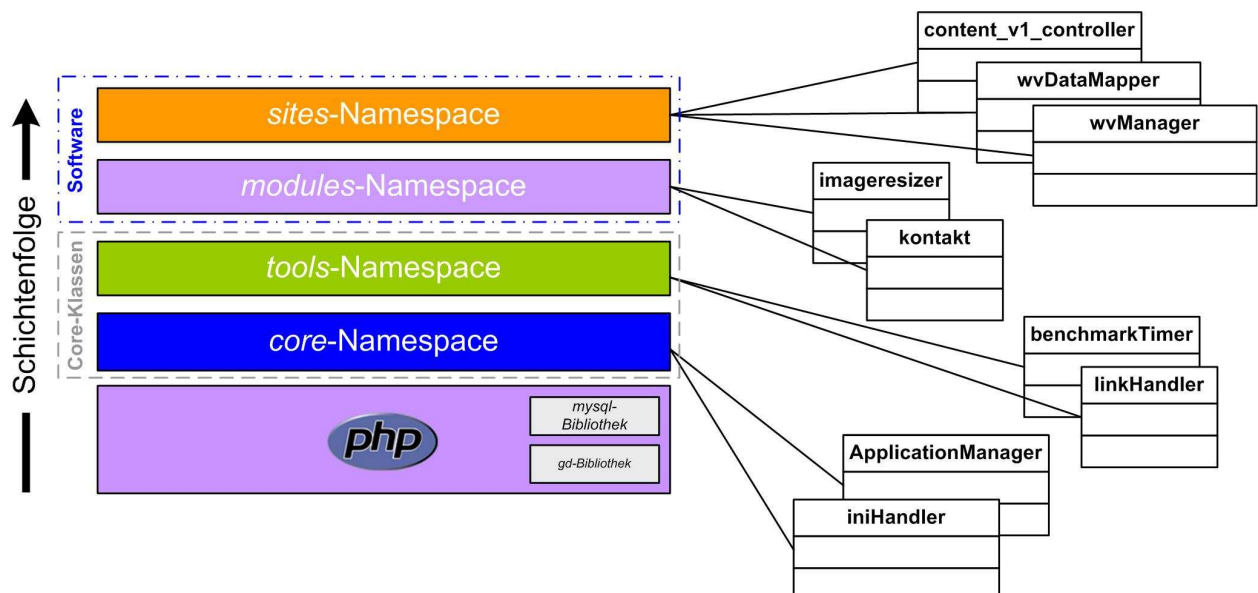


Abb. 6-2: Schichten-Architektur der Klassen

Abbildung 6-2 zeigt außerdem, dass der Architekturgedanke des Schichten-Designs auch in den Core-Klassen weitergeführt ist. Der *core*-Namespace beinhaltet die grundlegenden Funktionalitäten, die aus Sicht des Frameworks zum Betrieb einer Webapplikation unabdingbar sind. Die erweiterten Funktionalitäten bzw. Hilfsmittel zur Erstellung von Funktionen und Modulen, die nicht zwingend für den Betrieb benötigt werden, sind im *tools*-Namespace strukturiert. Alle weiteren Programmteile können nun die Funktionalitäten aus den beiden Namespaces einbinden und für erweiterte Funktionen nutzen.

Module, die in mehreren Programmen oder Programmteilen verwendet werden und auf den Core-Klassen aufsetzen sind im *modules*-Namespace abgelegt. Noch höher angesiedelte Applikationen befinden sich im *sites*-Namespace. Dort werden alle projektrelevanten Programmteile in einer Ordnerstruktur abgelegt, die nach Daten-, Business- und Präsentations-Schicht unterscheidet. Die zum Betrieb der PHP-Core-Schicht und der Applikation benötigten Konfigurationsdateien sind unter dem *config*-Namespace abgelegt.

## 6.2.6 Daten-Schicht

Die Datenschicht der Anwendung befasst sich mit der Akquise bzw. der Persistierung von Daten der Anwendung. Sie stellt der Business-Schicht Methoden zur Verfügung, mit denen Daten aus der Datenquelle geladen werden können. Das Laden von Daten beinhaltet, wie bereits unter 6.1 beschreiben, das Lesen der relational abgelegten Daten aus den dafür vorgesehenen Tabellen und das anschließende Überführen der Daten in Objekte und Beziehungen. Beim Speichern der Objekte, oder Objektbäume, werden diese in ihre Bestandteile – Daten und Beziehungen – zerlegt und relational in der Datenbank gespeichert.

Zum Zweck der Strukturierung und Vereinfachung ist die Daten-Schicht in weitere Unterschichten unterteilt, wie das folgende Schaubild verdeutlicht. Der Vollständigkeit wegen und um in der Teilarchitektur auch die Verwendung der Core-Klassen für die Abstraktion zu unterstreichen, sind auch die Schichten *Datenhaltung* und *Datenzugriffs-API* erläutert:

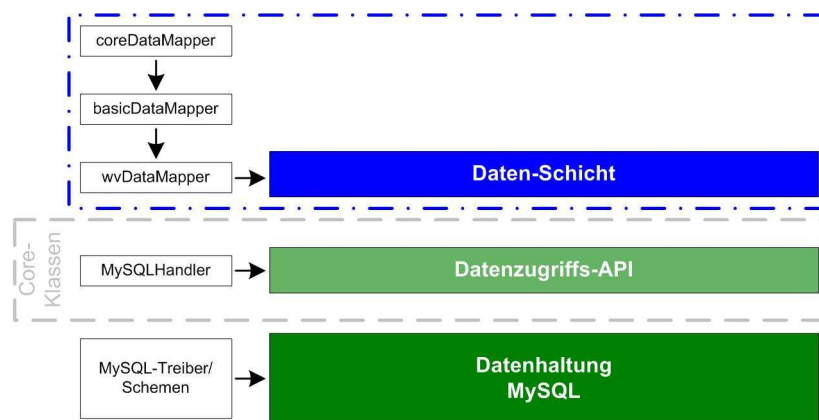


Abb. 6-3: Aufbau der Datenschicht

Die unterste Schicht, der in Abbildung 6-3 dargestellten Schichten, bildet die physikalische *Datenhaltung*. Diese setzt sich aus Sicht der Applikation aus den Schemen, die in die Datenbank eingespielt wurden und den MySQL-Treibern zusammen. Unter einem Schema versteht der Autor die Gesamtheit der Tabellen und Konfigurations-Einstellungen der verwendeten Datenbank.

Die als *Datenzugriffs-API* definierte Schicht bildet eine standardisierte Zugriffs-Schicht auf die Tabellen und Inhalte der *Datenhaltung*. Um diese Funktion bereitzustellen wurde der *MySQLHandler* im Namespace `core::database` implementiert. Diese Core-Klasse stellt Standard-Zugriffsmethoden bereit um eine SQL-Abfrage an die Datenbank zu senden. Der *MySQLHandler* abstrahiert die Datenbank damit soweit gegen die *Daten-Schicht*, das letztere keine Kenntnis der Datenbank-Zugriffs-Spezifika besitzen muss. Details zu den zur Verfügung stehenden Methoden können in der Quellcode-Dokumentation oder im Quellcode nachgelesen werden.

Der Bereich der *Daten-Schicht* ist, wie im folgenden UML-Diagramm beschreiben, in drei weitere logische Schichten unterteilt:

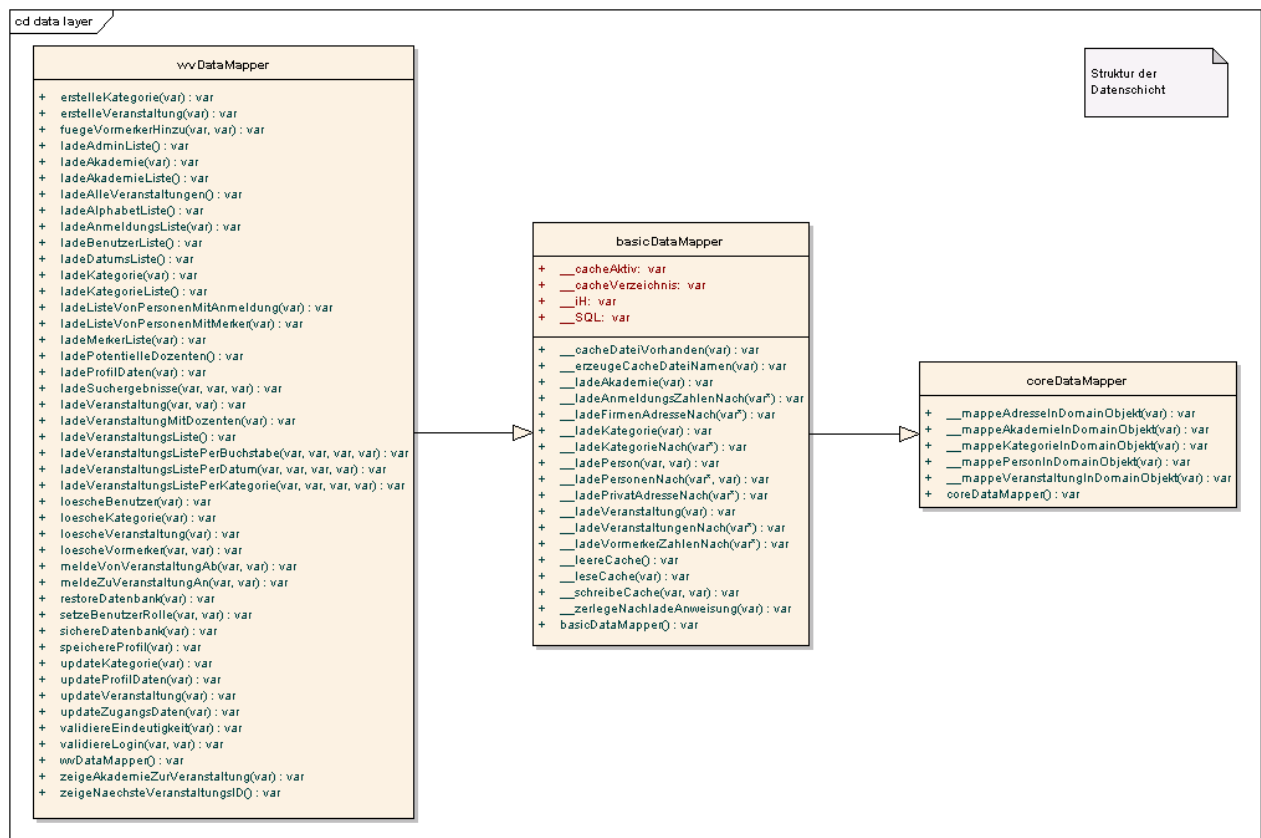


Abb. 6-4: UML Klassen-Diagramm der Daten-Schicht

Die Klasse *coreDataMapper* stellt grundlegende Methoden des „Mappens“ von SQL-Ergebnismengen in (Domain-)Objekte der Anwendung bereit. In den dort verfassten Methoden werden aus der Datenbank gewonnene Ergebnis-Datensätze in Objekte eingelesen.

Da der *basicDataMapper* von *coreDataMapper* abgeleitet ist, erbt er dessen Methoden. Er stellt Funktionen bereit, die das Laden von Objekten auf einer höheren Ebene abstrahieren. Die Klasse bietet beispielsweise die Möglichkeit, die Ladezeit der Objekt-Bäume mittels eines Loading-Caches zu verringern. Der genannte Mechanismus legt bereits geladene Objekte in einem dafür angelegten Cache-Verzeichnis auf der lokalen Festplatte des Webserver ab und bedient sich bei Bedarf daraus. Jedes Objekt ist dabei mit einem „Verfallsdatum“ gekennzeichnet und wird nach Ablauf dieses Zeitpunktes erneut aus der Datenbank geladen, um die Aktualität der angezeigten Daten garantieren zu können.

Der *vvDataMapper* erbt wiederum von *basicDataMapper* und stellt für die Business-Schicht abstrakte Datenzugriffs-Methoden zur Verfügung. Die im *vvDataMapper* vorhandenen Funktionen können bei Bedarf parametrisiert werden, was den Umfang der geladenen Objekt-Liste bzw. die Ausprägung der Einzel-Objekte der Liste beeinflusst. Im Detail bedeutet das, dass durch den angegebenen Konfigurationsabschnitt das Nachlade-Verhalten von komplexen Attributen bzw. Beziehungen und sich daraus ergebende Objektbäume geregelt wird. Im konkreten Beispiel kann beim Laden einer Veranstaltungsliste, bei der die Veranstaltungen mit ihren Dozenten geladen werden sollen, die zuständige Mapper-Methode mit einem Konfigurations-Parameter aufgerufen werden. Zum komplexen Attribut *Dozent* werden dann die als Dozenten assoziierten Personen hinzu geladen, jedoch keine weiteren Beziehungen, die diesen betreffen. Weitere Implementierungsdetails können den in der Tabelle 6-b aufgezeigten Quellcode-Dokumentationen entnommen werden.



## 6.2.7 Business-Schicht

Die Business-Schicht ist für die Abwicklung aller Business-Prozesse innerhalb der Anwendung zuständig, steuert Workflows oder lädt Daten. Zudem beinhaltet sie die Domain-Objekte der Anwendung. Abbildung 6-5 zeigt die Struktur der Daten-Objekte und das Klassendiagramm des *wvManager*'s – der zentralen Business-Schicht-Komponente.

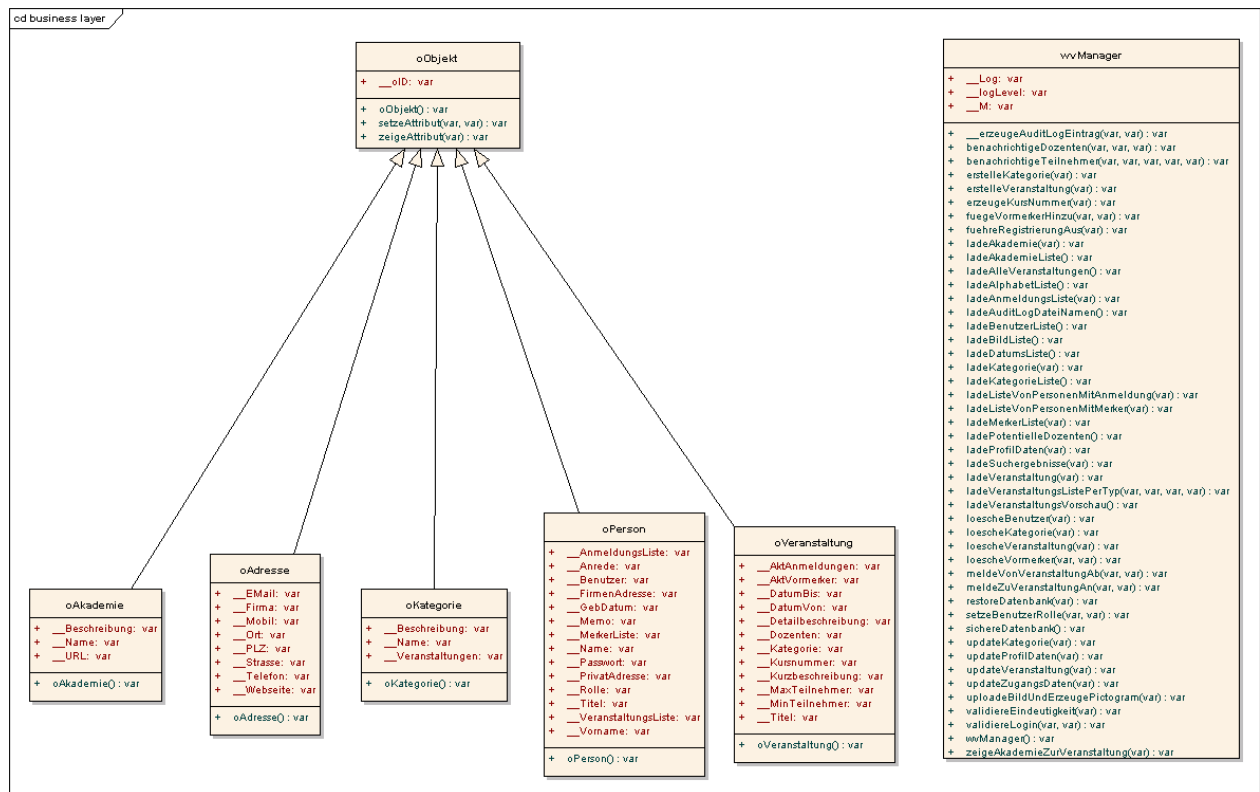


Abb. 6-5: UML Klassen-Diagramm der Business-Schicht

Die Klassen *oObjekt*, *oAkademie*, *oAdresse*, *oKategorie*, *oPerson* und *oVeranstaltung* repräsentieren die Domain-Objekte der Anwendung. Sie kapseln die Daten während der Ausführung der Software und fungieren als Schnittstelle zwischen den Schichten der Anwendung.

Der *wvManager* stellt den Modulen und PlugIns der Präsentations-Schicht Methoden zur Verfügung, mit denen die Präsentations-Schicht Daten laden, oder Business-Prozesse auf Grund von Benutzer-Interaktionen anstoßen kann. Ein plastisches Beispiel hierfür ist die Ausführung der Suche. Bei dieser fragt die Präsentations-Schicht eine zum Suchbegriff passende Ergebnis-Liste an. Daraufhin führt die Business-Schicht abhängig von der Gestaltung der Suchbegriffe die von der Datenschicht zur Verfügung gestellten Methoden zum Laden einer Ergebnis-Menge aus. Wurden die Ergebnismengen der unterschiedlichen Suchkriterien geladen, so werden die Suchergebnisse konsolidiert, die Objektinformationen zu den Ergebnis-IDs aus der Datenschicht gezogen und die Liste sortiert an die Präsentations-Schicht zurückgegeben.

Eine detaillierte Beschreibung zu weiteren Business-Prozessen und deren Implementierung kann der Dokumentation der Klasse *wvManager* entnommen werden.

Ein weiterer Bereich der Business-Schicht ist der Bereich der automatisierten Benachrichtigungen bzw. des Backups der Daten. Die Implementierungen dieser Aufgaben sind zum Teil in der Klasse *wvManager* enthalten, zum Teil werden diese Funktionen in mehreren Skripten abgebildet. Um diese cronjobfähig zu machen muss ein Workaround implementiert werden, da der Shared-Hosting-Account *w3service.net* keine direkte Konfiguration eines Cronjobs unterstützt.

Eine erste Möglichkeit ist ein so genannter „Fake-Cron“, bei dem die CronJob-Funktionalität in die aufgerufenen Seiten eingebettet wird. Dabei werden, wenn beispielsweise die Startseite aufgerufen wird, die automatischen Aufgaben wie das Bereinigen von Daten, Versenden von E-Mails oder Backup der Datenbank während des Abrufs abgearbeitet. Das birgt jedoch zwei gravierende Nachteile: der Aufbau der Webseite verzögert sich bei großen Jobs, bzw. die Jobs werden nur beim Besuch der Seite, d.h. nicht regelmäßig ausgeführt.

Eine weitaus bessere Alternative ist ein Cronjob-Skript (*cronjob.php*), das via http zugänglich ist und die Funktionalität kapselt. Dieses wird dann mit Hilfe des Services **cronjob.de** regelmäßig aufgerufen und ausgeführt. Die in den Anforderungen definierten automatischen Benachrichtigungen werden somit regelmäßig und pünktlich versendet und die Backups zu den definierten Zeiten angefertigt. Die Ausführungszeiten und Häufigkeiten können in einem Konfigurationsmenü unter <http://cronjob.de> problemlos konfiguriert werden. Sollten später weitere CronJobs notwendig werden, können diese auf dem bisherigen Software-Kern aufsetzend implementiert und dort eingetragen werden.

## 6.2.8 Präsentations-Schicht

Die Präsentations-Schicht ist für den Aufbau der GUI und die Reaktion auf Benutzer-Eingaben zuständig. Sie bezieht Daten und Verhaltens-Anweisungen von der Business-Schicht. Da die Präsentations-Schicht dieser Software nicht nach dem Front-Controller-Pattern implementiert ist, wird sie nicht direkt von der Business-Schicht gesteuert, sondern ruft eigeninitiativ die Business-Schicht auf und erwartet Daten bzw. Anweisungen, was für diesen Anwendungsfall jedoch keine Nachteile mit sich bringt.

Abbildung 6-6 beinhaltet die Dokumentation der Klassen der Präsentations-Schicht. Um die Erkennbarkeit der Struktur zu maximieren, wurde die Abbildung in Querformat abgedruckt. An dieser Stelle sei darauf hingewiesen, dass die in diesem Kapitel abgebildeten UML-Diagramme als *Enterprise Architect*-Projekte im Verzeichnis *Dokumentation/Abbildungen/* der Begleit-CD zu finden sind.

Die Klassen *Page*, *Document*, *Content*, *Template* und *baseController* sind Komponenten der PHP-Core-Schicht. Sie bilden die Funktionalität des Page-Controllers ab, organisieren bei der Initialisierung einer Webseite den internen Objektbaum und kümmern sich während der Transformation der Seite um den Aufbau von Views und die Ausführung der dabei beteiligten Controller.

Die Klasse *baseController* ist dabei das allgemeine Interface, das von einem Document-Controller gemäß MVC-Pattern implementiert werden muss. Er beinhaltet Standard-Methoden, die innerhalb eines Document-Controllers verwendet werden können.

Um die in den verschiedenen Controller-Klassen gemeinsam verwendete Komponenten und Darstellungs-Funktionalitäten zentral zur Verfügung stellen zu können, existieren mehrere abstrakte Document-Controller, die wiederum vom Interface *baseController* ableiten. Ein

Beispiel dafür ist die Klasse *formhelper\_v1\_controller*, die auf der Core-Klasse *formManager* aufbauend komplexere GUI-Elemente zur Verfügung stellt. Des Weiteren beschäftigt sich die Klasse *adminBaseController* mit der Rolle und den Berechtigungen des eingeloggten Benutzers. Ist ein Benutzer angemeldet, so prüft der vom *adminBaseController* ererbte Document-Controller, ob der aufgerufene View vom Benutzer eingesehen werden darf.

Die Instanz der Klasse *content\_v1\_controller* kümmert sich um die Einbindung der Seiten-Inhalte und stellt damit den zentralen PlugIns-Handler bereit. Zudem bildet sie zusammen mit der Klasse *loginmenu\_v1\_controller* die Funktionalität des Applikations-Rahmens ab. Klassen wie *merker\_v1\_controller*, *registrierung\_v1\_controller* und *akademieliste\_v1\_controller* sind Controller von diversen eingebetteten PlugIns. In gleicher Weise kapseln beispielsweise die Klassen *rechtezuweisen\_v1\_controller* oder *logdateieneinsehen\_v1\_controller* die Funktionen des Backends.

Weitere Implementierungsdetails können den Quellcode-Dateien im Installationsverzeichnis der Begleit-CD oder den Quellcode-Dokumentationen entnommen werden.



## 6.3 Sequenz-Diagramme

Das vorliegende Kapitel dokumentiert die Dynamik der oben beschriebenen Klassen. Es wird im weiteren Verlauf sowohl auf das Zusammenspiel der skizzierten Instanzen als auch auf deren Bedeutung näher eingegangen.

### 6.3.1 Übersicht

Abbildung 6-7 beschreibt den in 6.1 abstrakt beschriebenen Programm-Ablauf in Form eines Sequenz-Diagramms, welches das Zusammenspiel der Objekte der unterschiedlichen Schichten aufzeigt. Die dort skizzierten Instanzen sind farbig markiert um die Zugehörigkeit zu den im Design-Prozess definierten Schichten zu verdeutlichen. Grün eingefärbte Objekte sind Komponenten der Präsentations-Schicht, Objekte in violetter Farbe sind Teil der PHP-Core-Schicht, rote gehören zur Business-Schicht, blaue repräsentieren Instanzen der Datenschicht.

Der Großteil der in der Übersicht dargestellten Klassen bildet den allgemeinen Applikations-Rahmen, in den PlugIns eingebunden werden können. Dazu zählen die Objekte der Präsentations-Schicht, ausgenommen die Instanzen der PlugIn-Klassen, die Objekte der PHP-Core-Klassen, die zentrale Business-Komponente *wvManager* und die Datenschicht-Objekte. Der grau hinterlegte Bereich definiert den PlugIn-Bereich der Präsentations-Schicht in welchem diese vom zentralen Content-Controller *content\_vl\_controller* eingebunden werden. Je nach Modul werden unterschiedliche Core-Klassen und verschiedene Methoden der Business-Schicht verwendet, wie den folgenden Abbildungen entnommen werden kann.



### 6.3.2 Sequenz-Diagramme der PlugIns

Auf Grund dessen, dass die in einem Use Case beschriebenen Funktionalitäten programmiertechnisch in einem PlugIn gekapselt sind, sind die folgenden Diagramme mit den Titeln der unter Kapitel 2 beschriebenen Use Cases benannt. Dabei wurde die unter 6.3.1 gewählte Farblogik beibehalten. Implementierungsdetails und Beschreibungen zu den skizzierten Methoden können den Quellcode-Dokumentationen oder den PHP-Dateien entnommen werden. Die UML-Diagramme befinden sich ebenfalls als *Enterprise Architect* Projekt auf der Begleit-CD im Ordner *Dokumentation/Abbildung/*.

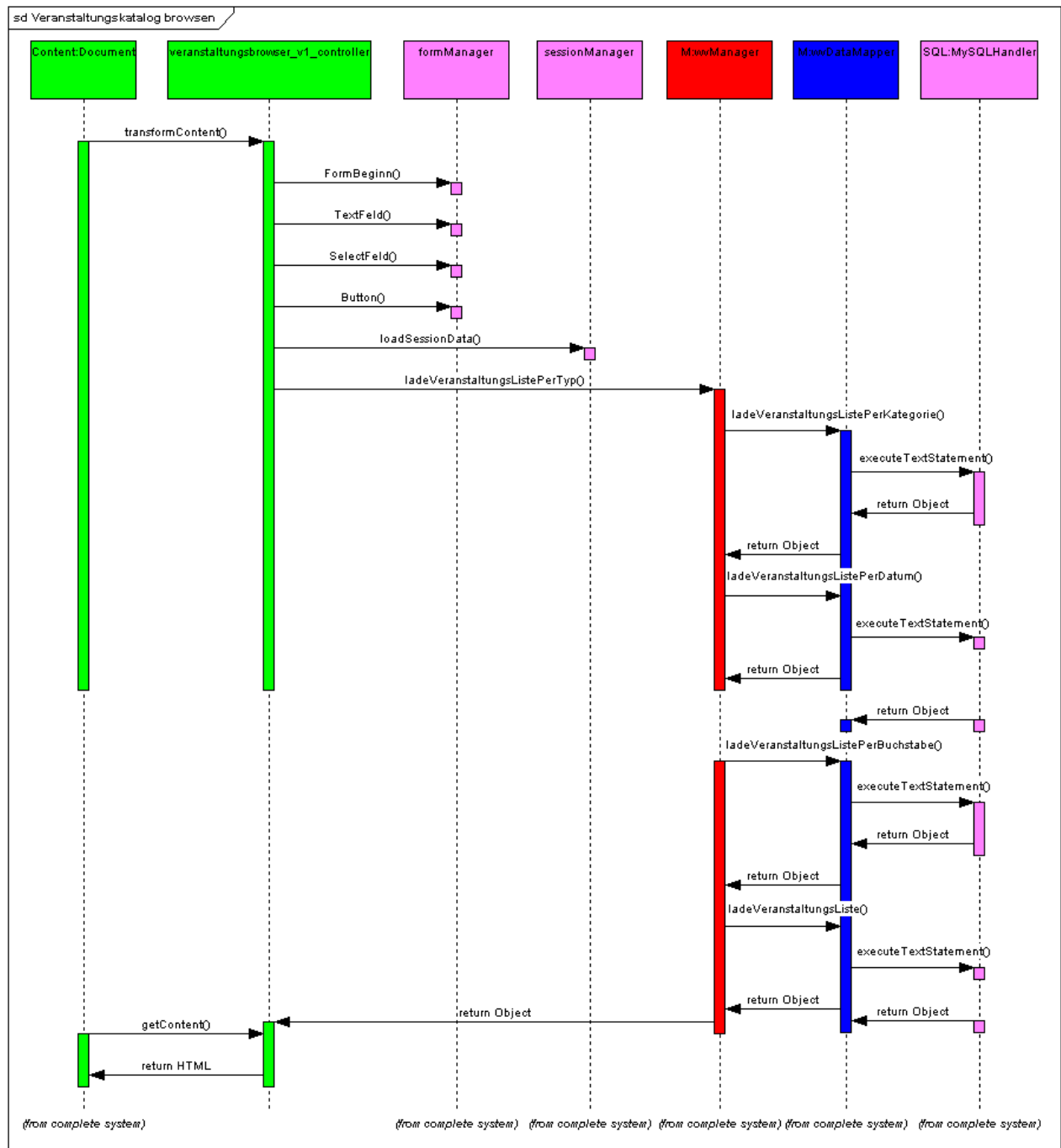


Abb. 6-8: Sequenz-Diagramm zum Use Case „Veranstaltungskatalog browsen“

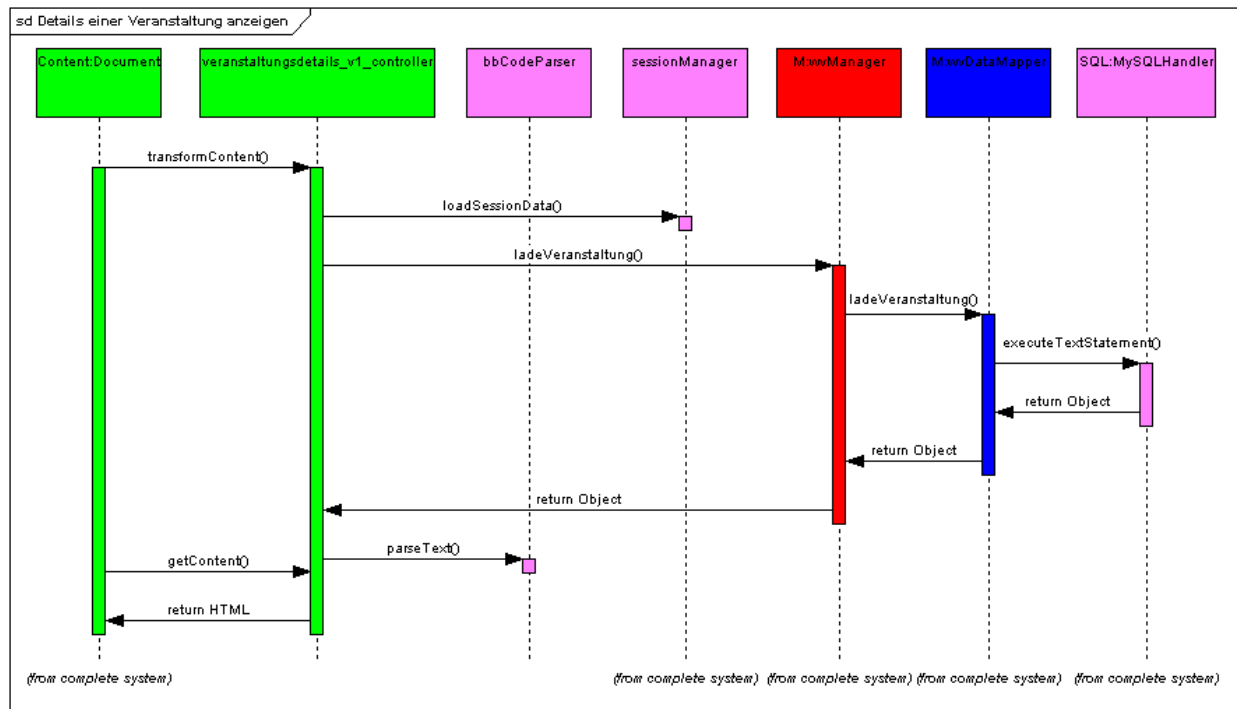


Abb. 6-9: Sequenz-Diagramm zum Use Case „Details einer Veranstaltung anzeigen“



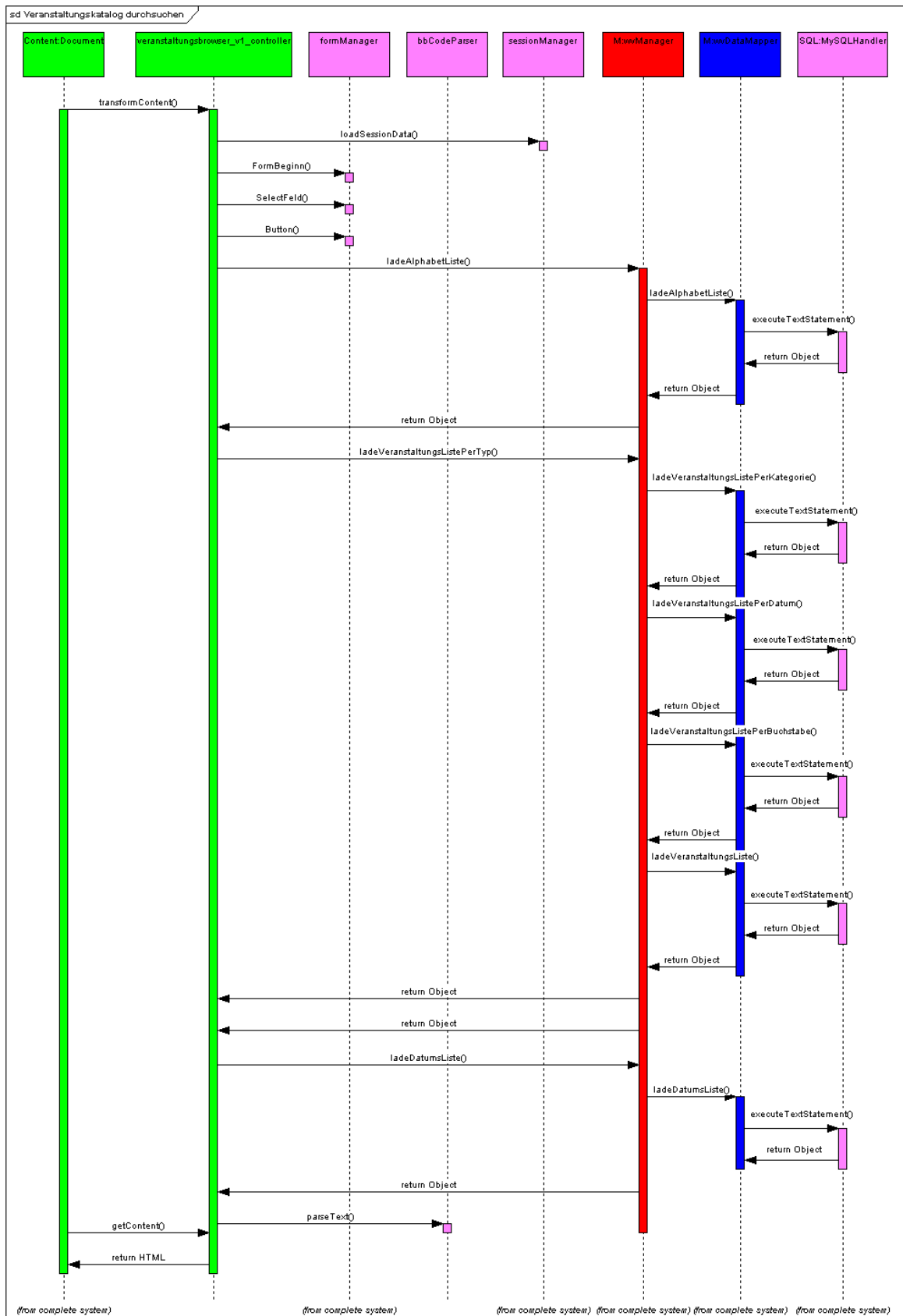


Abb. 6-10: Sequenz-Diagramm zum Use Case „Veranstaltungskatalog durchsuchen“

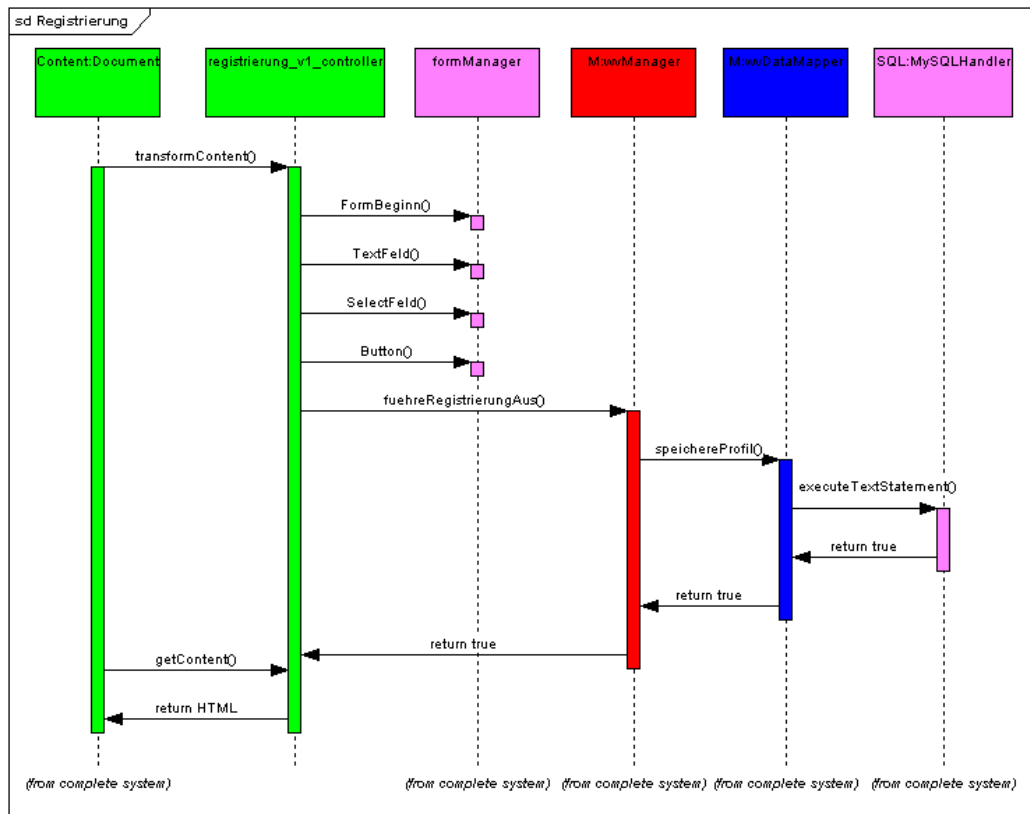


Abb. 6-11: Sequenz-Diagramm zum Use Case „Registrieren“

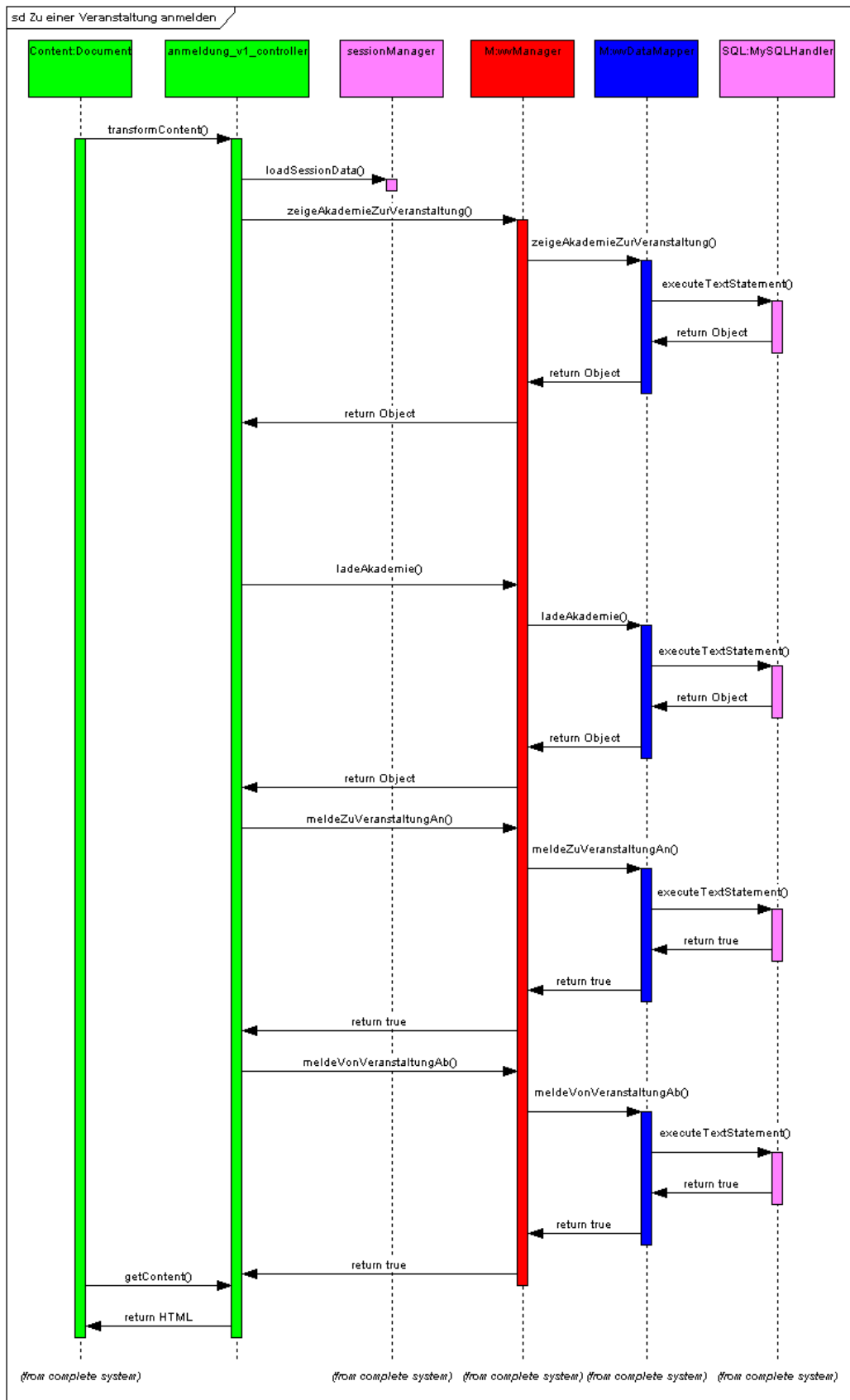


Abb. 6-12: Sequenz-Diagramm zum Use Case „Zu einer Veranstaltung anmelden“

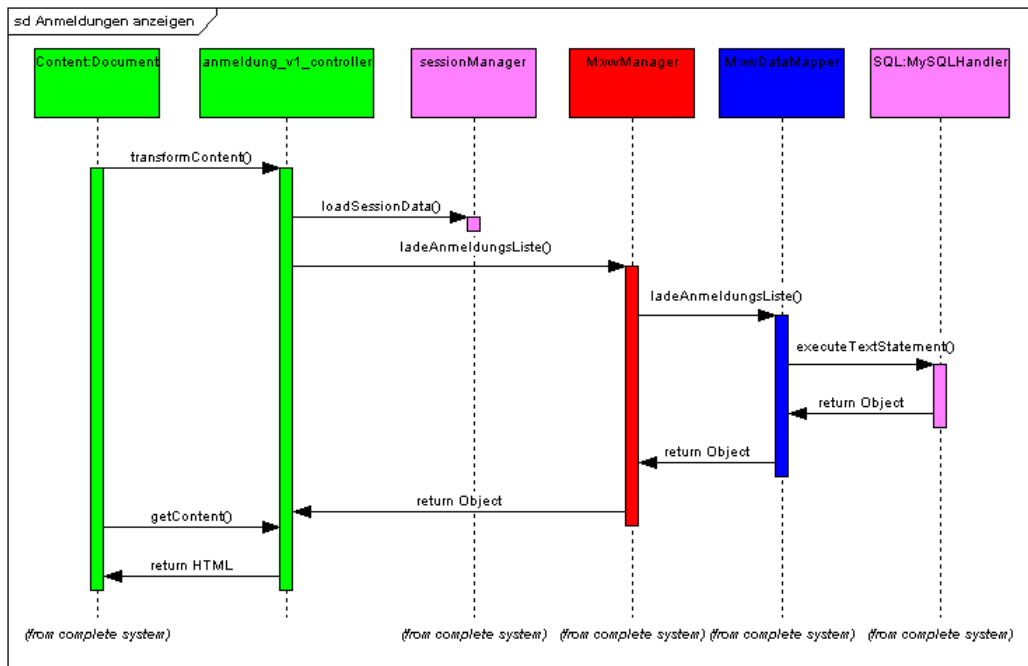


Abb. 6-13: Sequenz-Diagramm zum Use Case „Anmeldungen anzeigen“

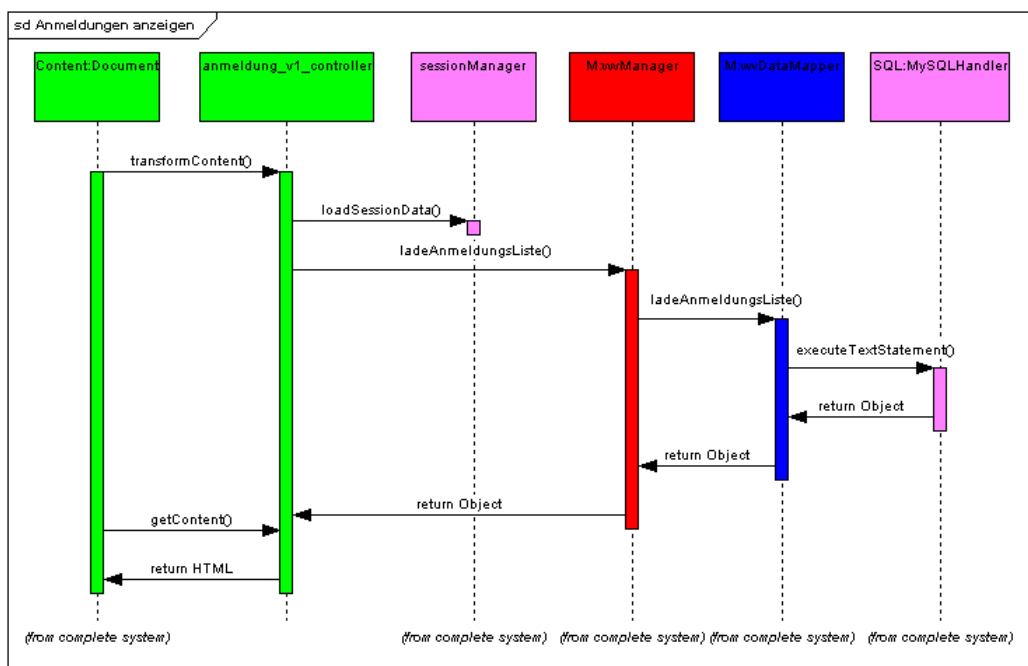


Abb. 6-14: Sequenz-Diagramm zum Use Case „Anmeldungen anzeigen“

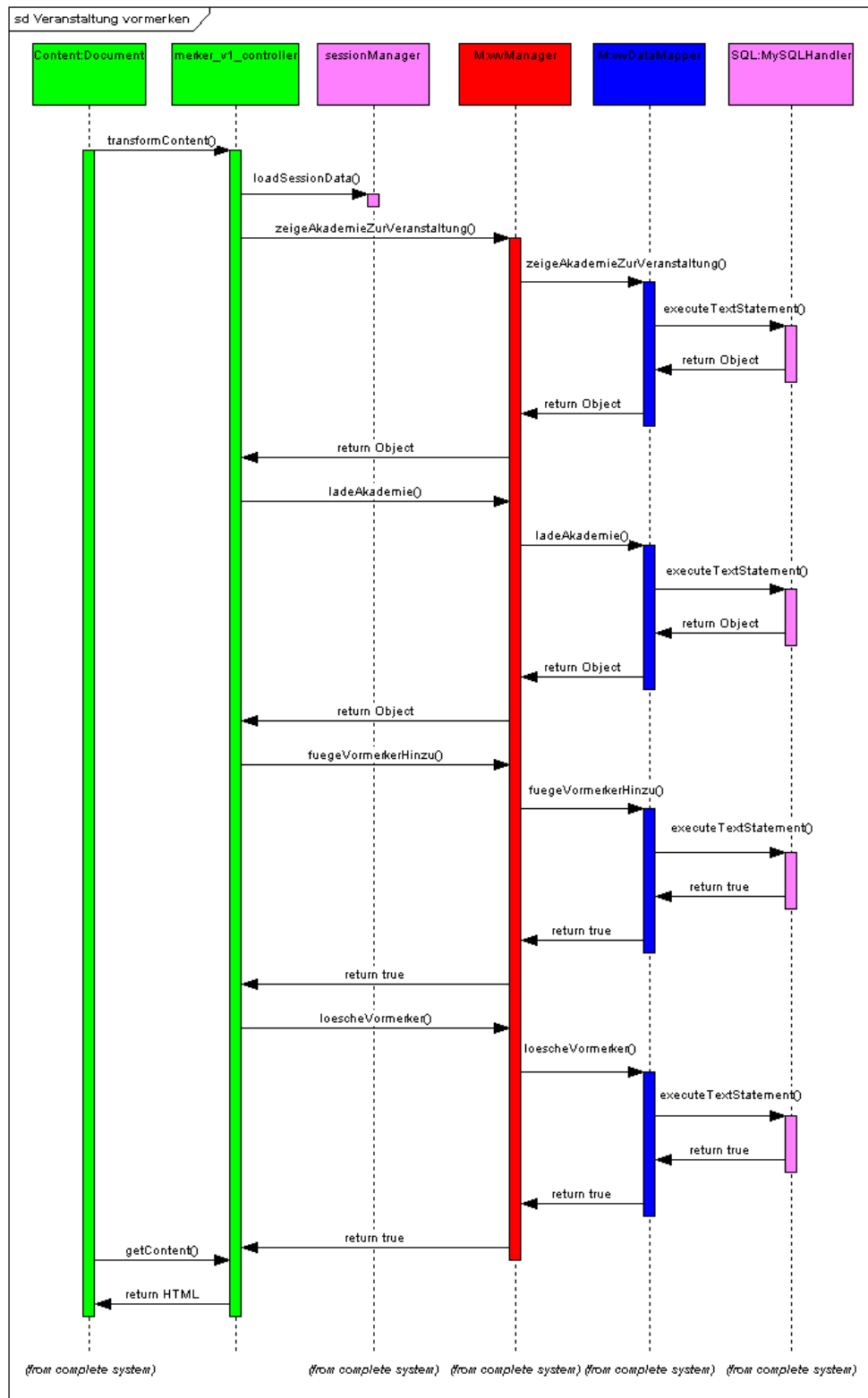


Abb. 6-15: Sequenz-Diagramm zum Use Case „Veranstaltung vormerken“

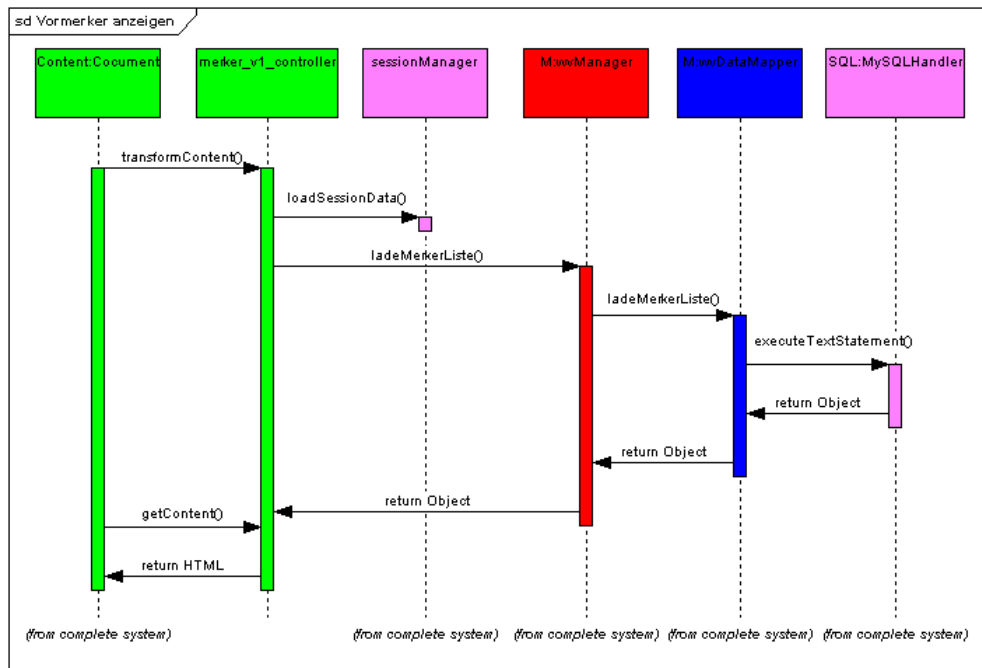


Abb. 6-16: Sequenz-Diagramm zum Use Case „Vormerker anzeigen“

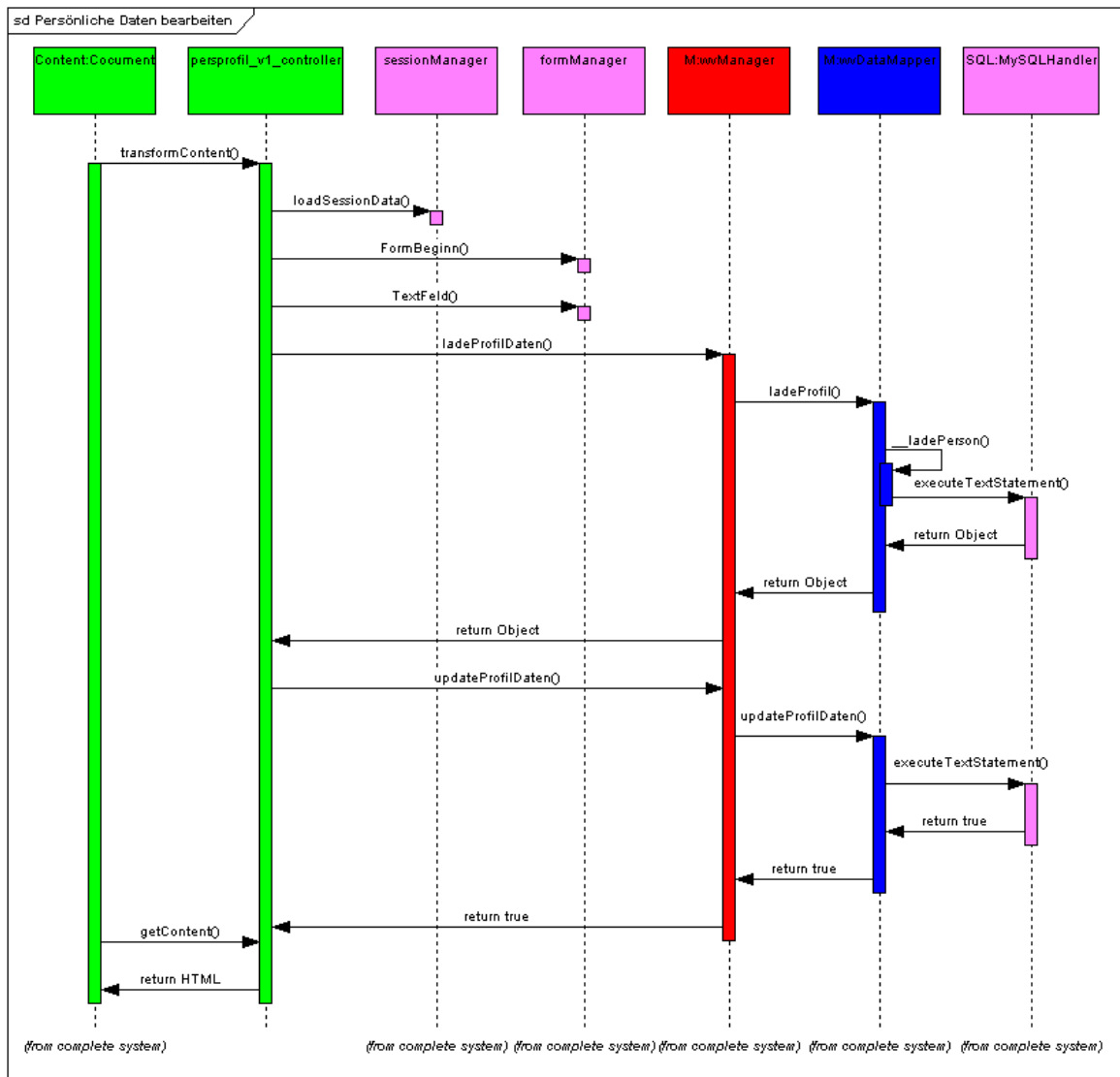


Abb. 6-17: Sequenz-Diagramm zum Use Case „Persönliche Daten bearbeiten“

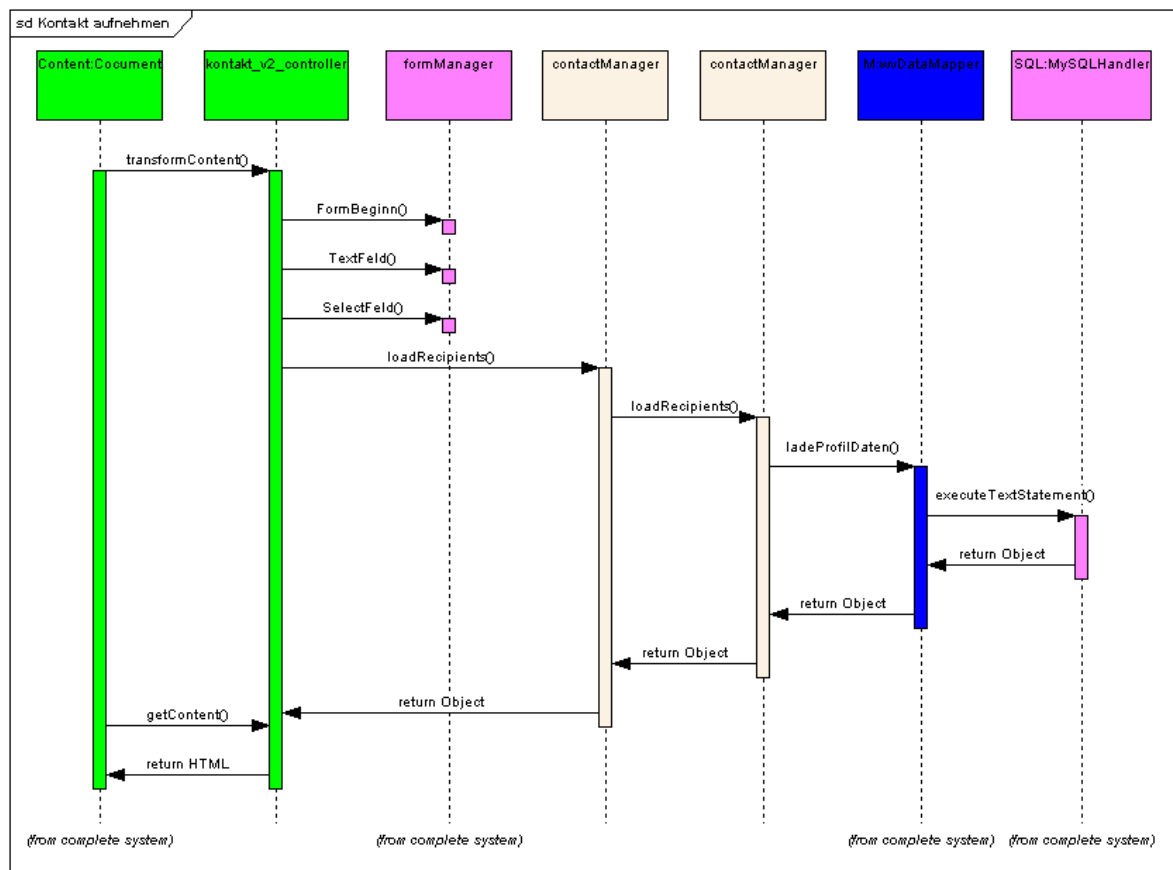


Abb. 6-18: Sequenz-Diagramm zum Use Case „Kontakt aufnehmen“

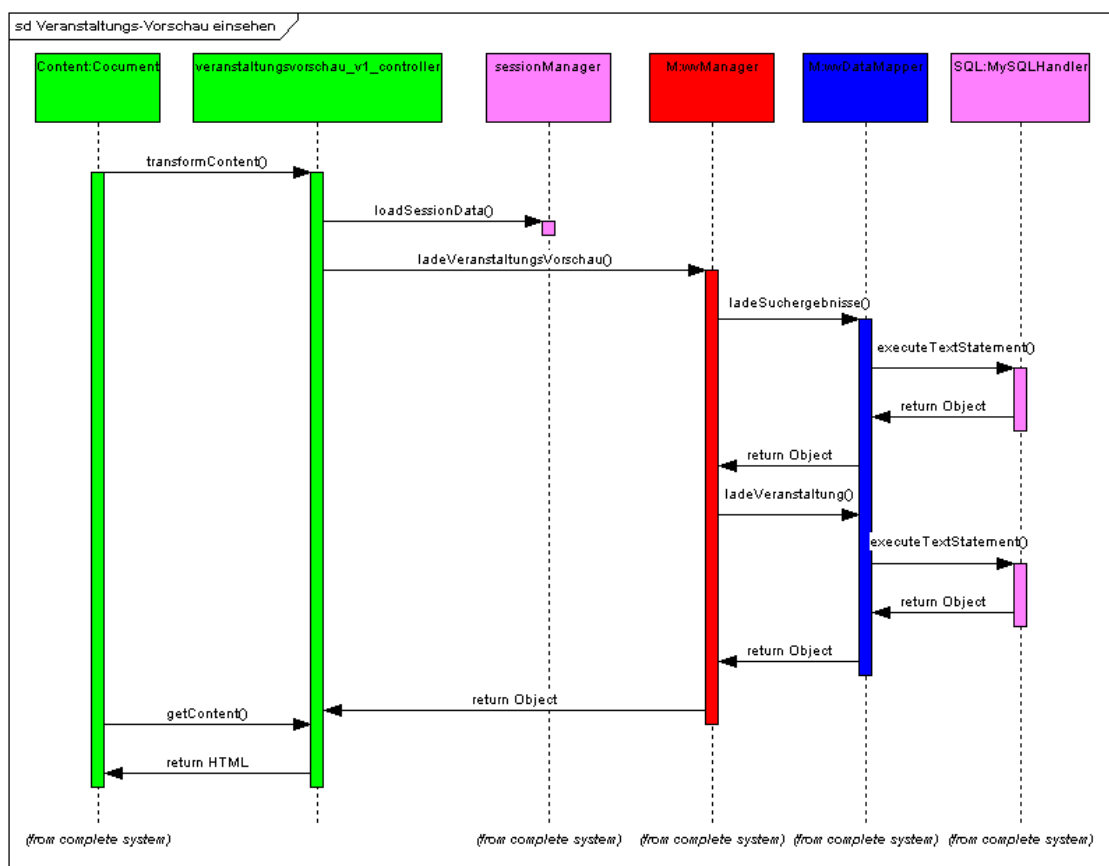


Abb. 6-19: Sequenz-Diagramm zum Use Case „Veranstaltungsvorschau einsehen“



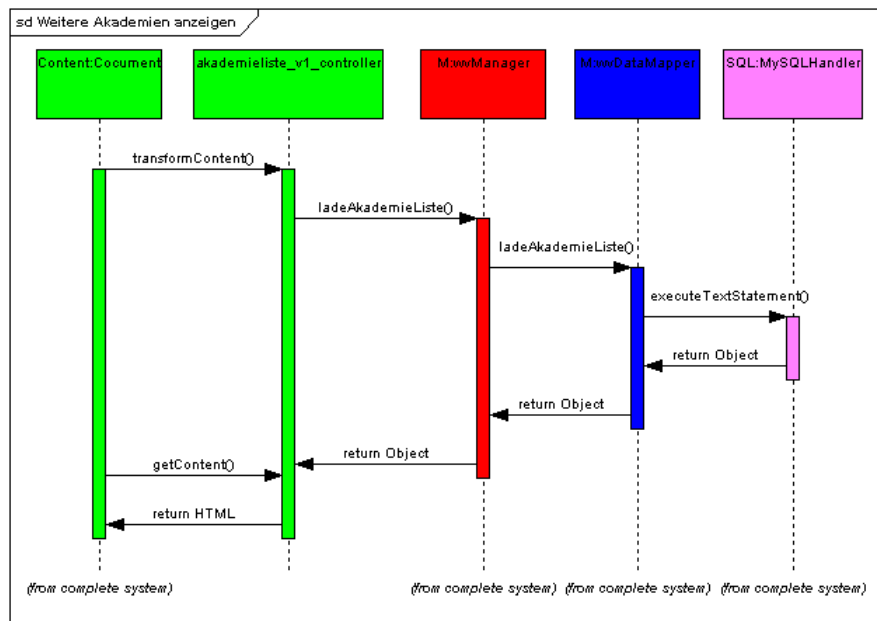


Abb. 6-20: Sequenz-Diagramm zum Use Case „Weitere Akademien anzeigen“

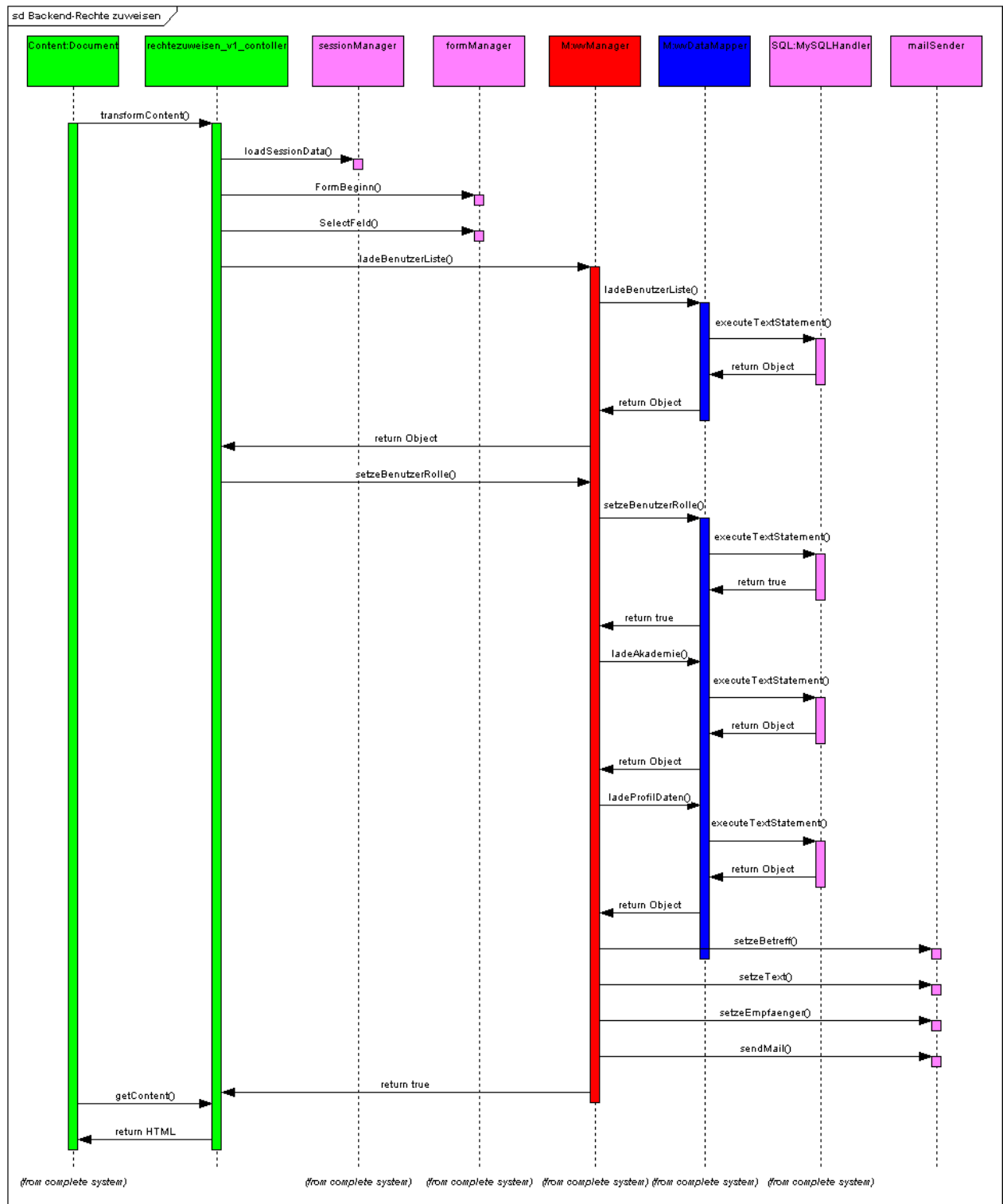


Abb. 6-21: Sequenz-Diagramm zum Use Case „Backend-Rechte zuweisen“

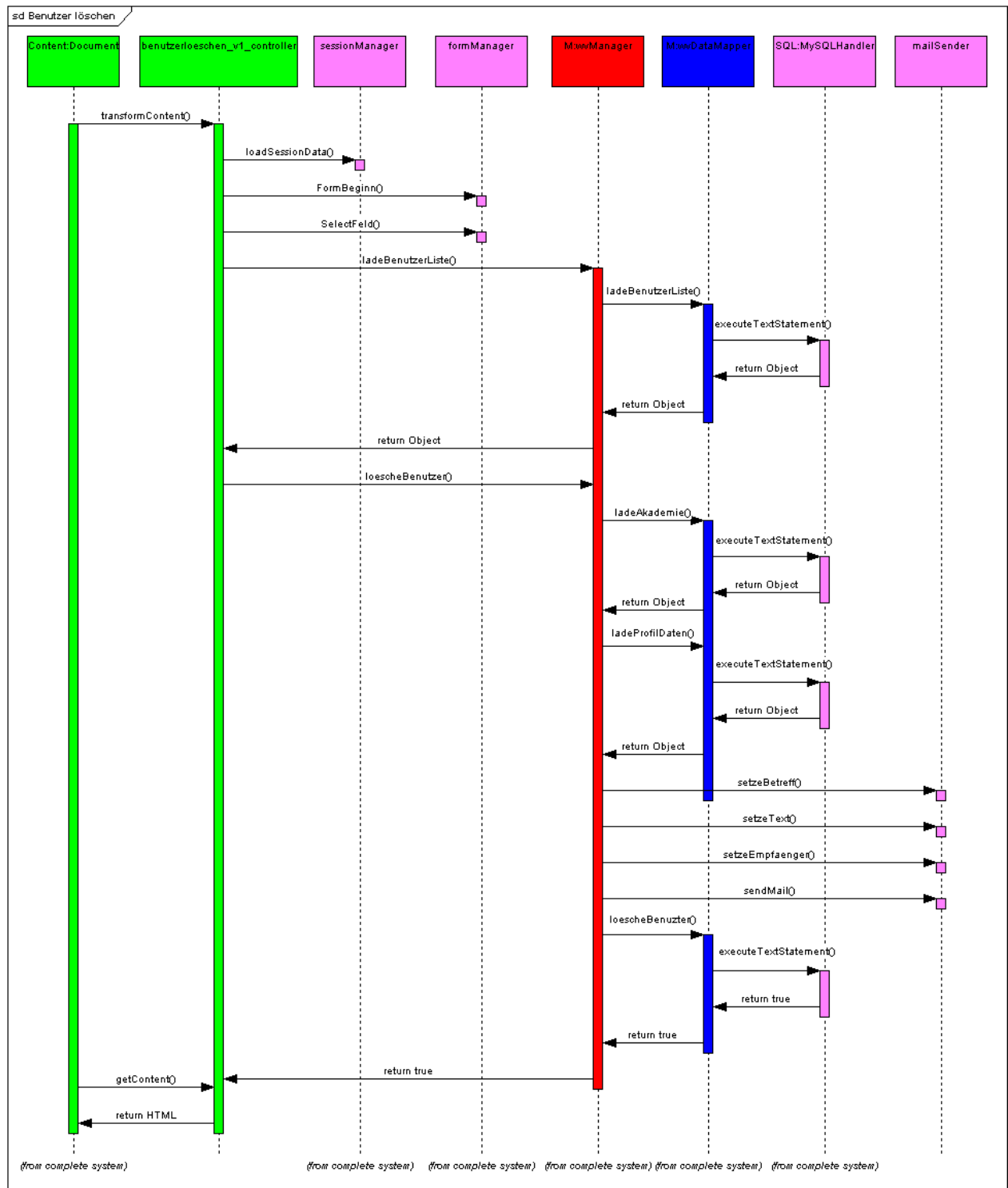


Abb. 6-22: Sequenz-Diagramm zum Use Case „Benutzer löschen“

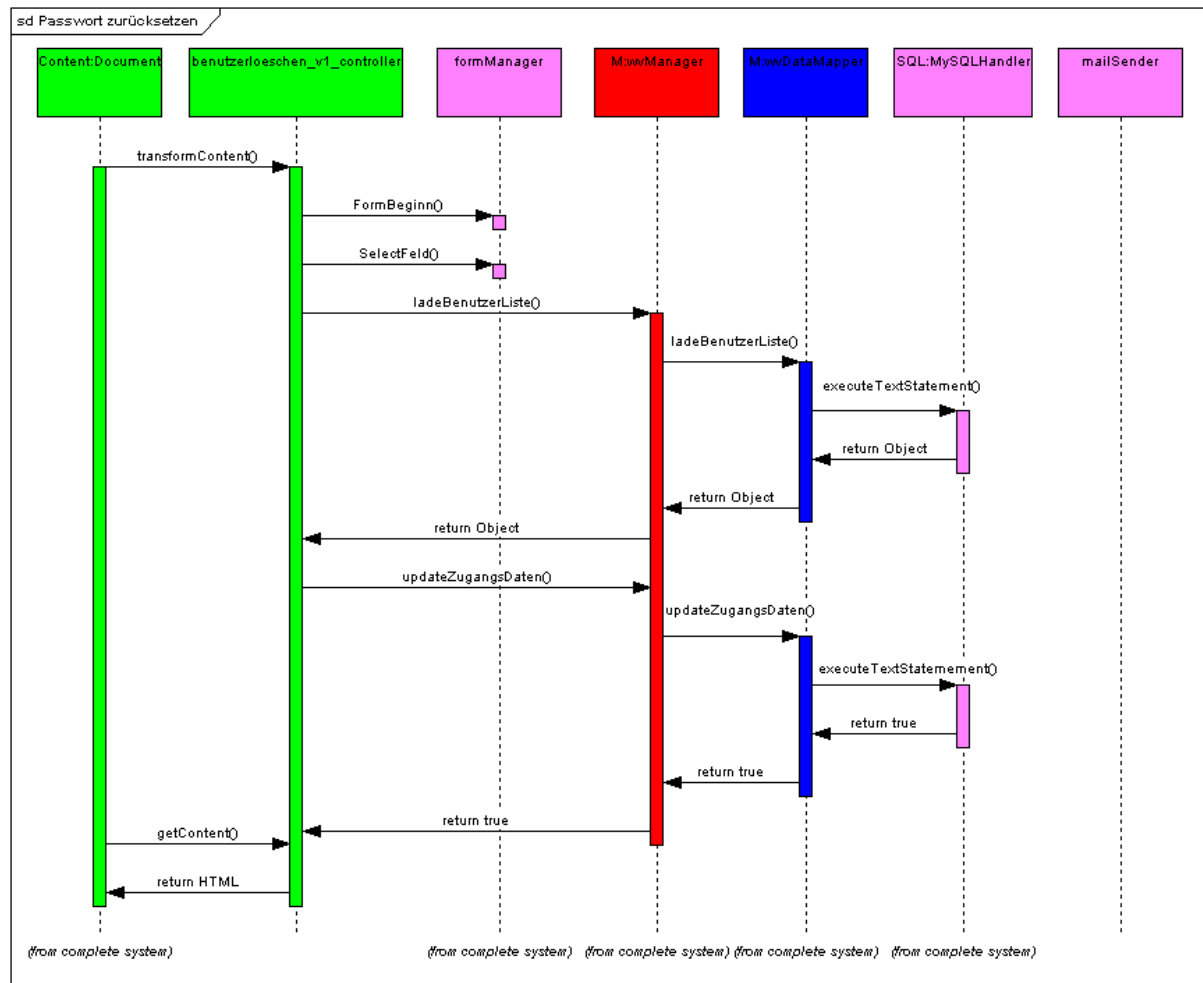


Abb. 6-23: Sequenz-Diagramm zum Use Case „Passwort rücksetzen“

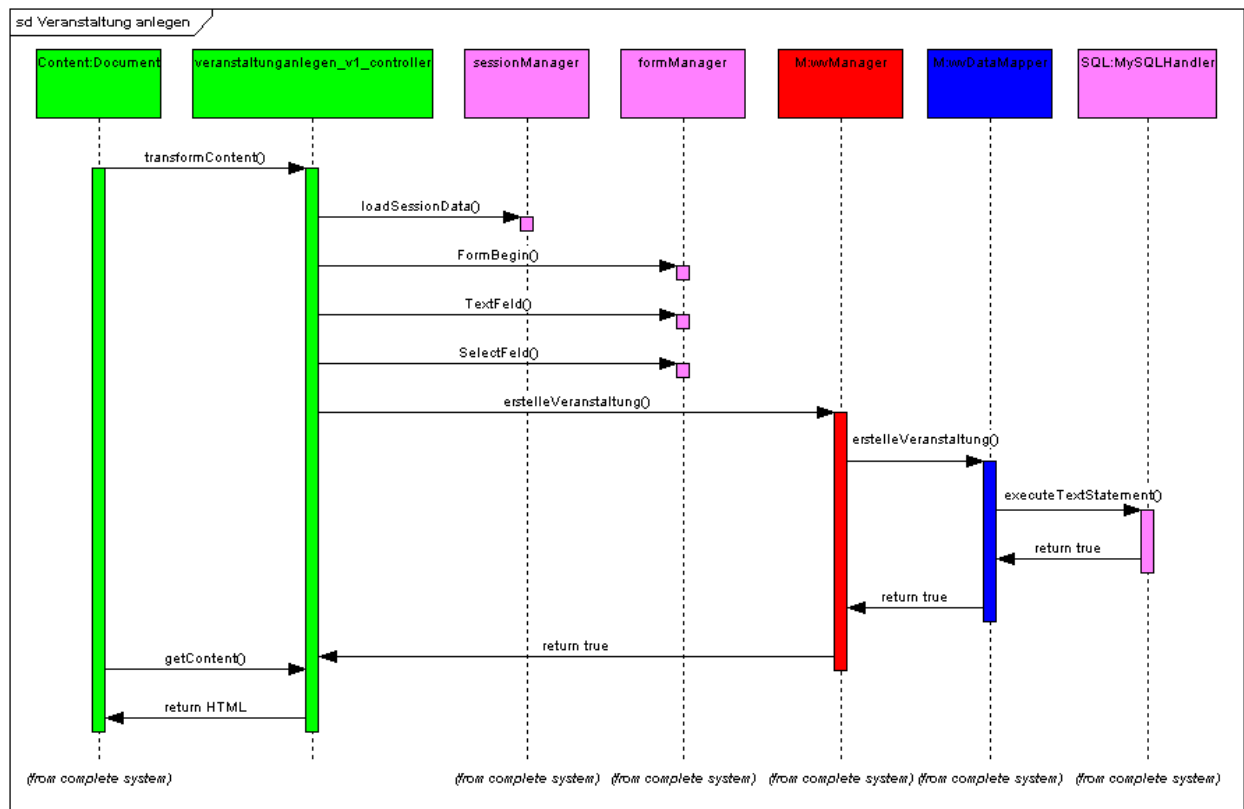


Abb. 6-24: Sequenz-Diagramm zum Use Case „Veranstaltung anlegen“

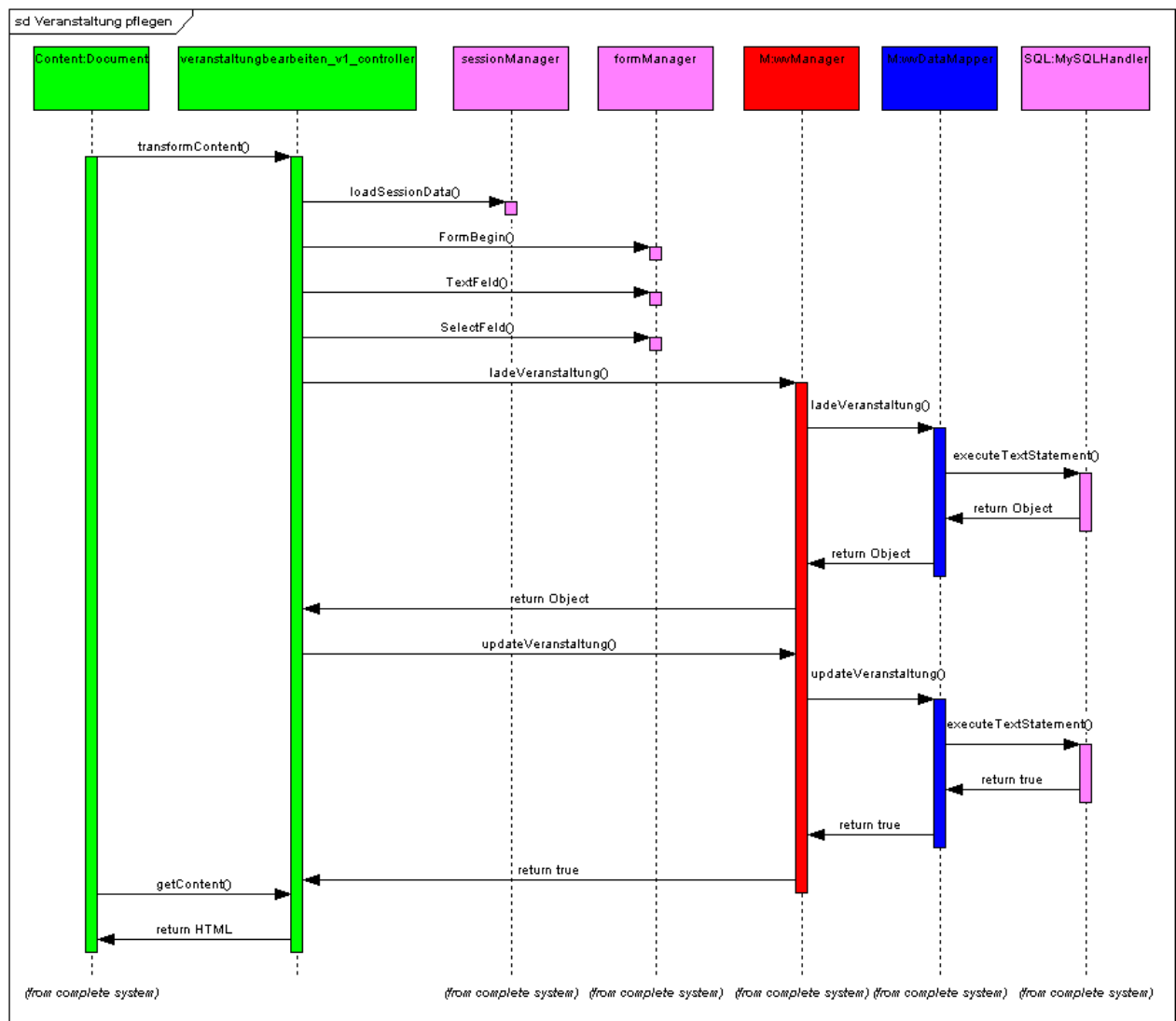


Abb. 6-25: Sequenz-Diagramm zum Use Case „Veranstaltung pflegen“

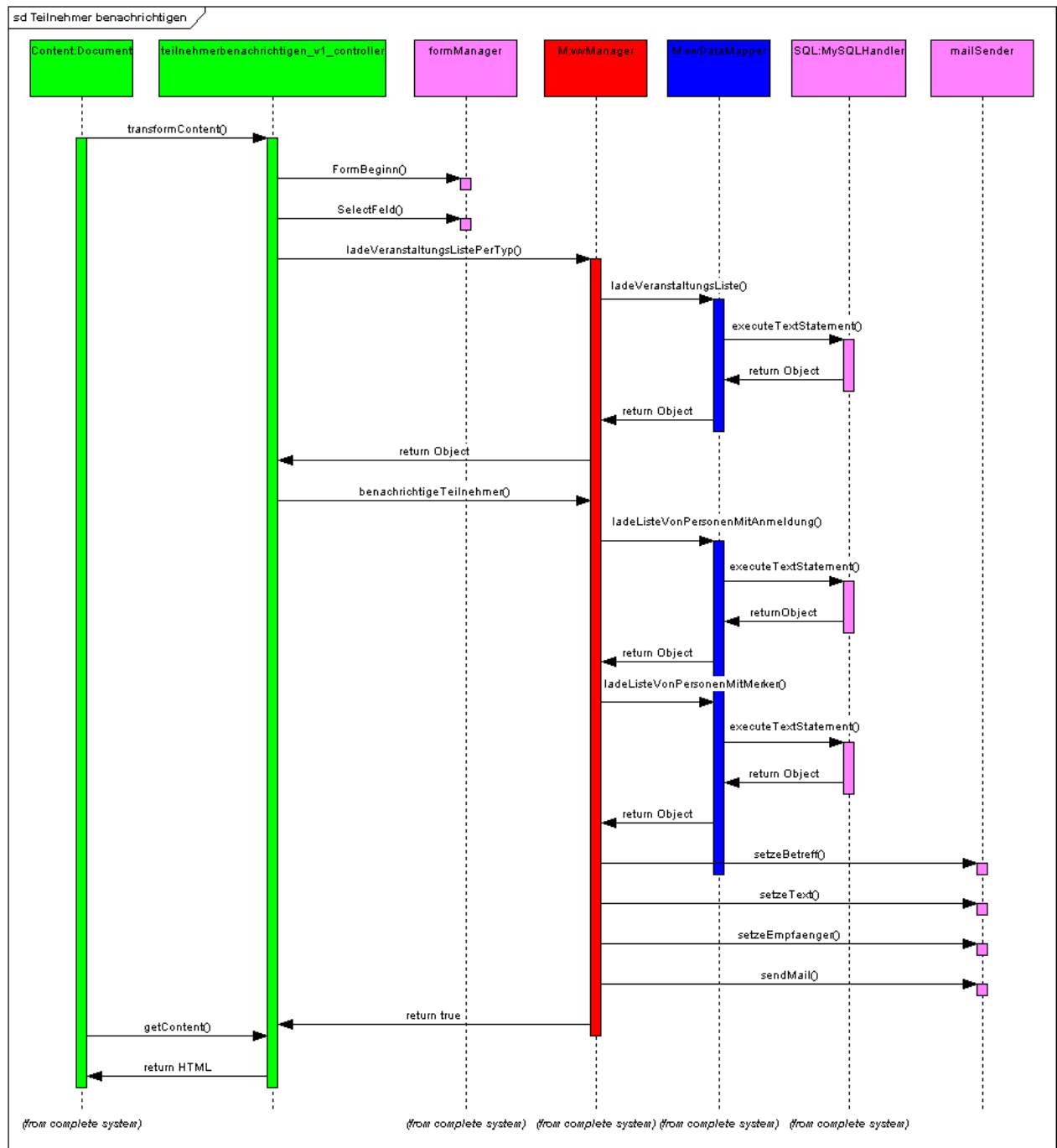


Abb. 6-26: Sequenz-Diagramm zum Use Case „Teilnehmer benachrichtigen“

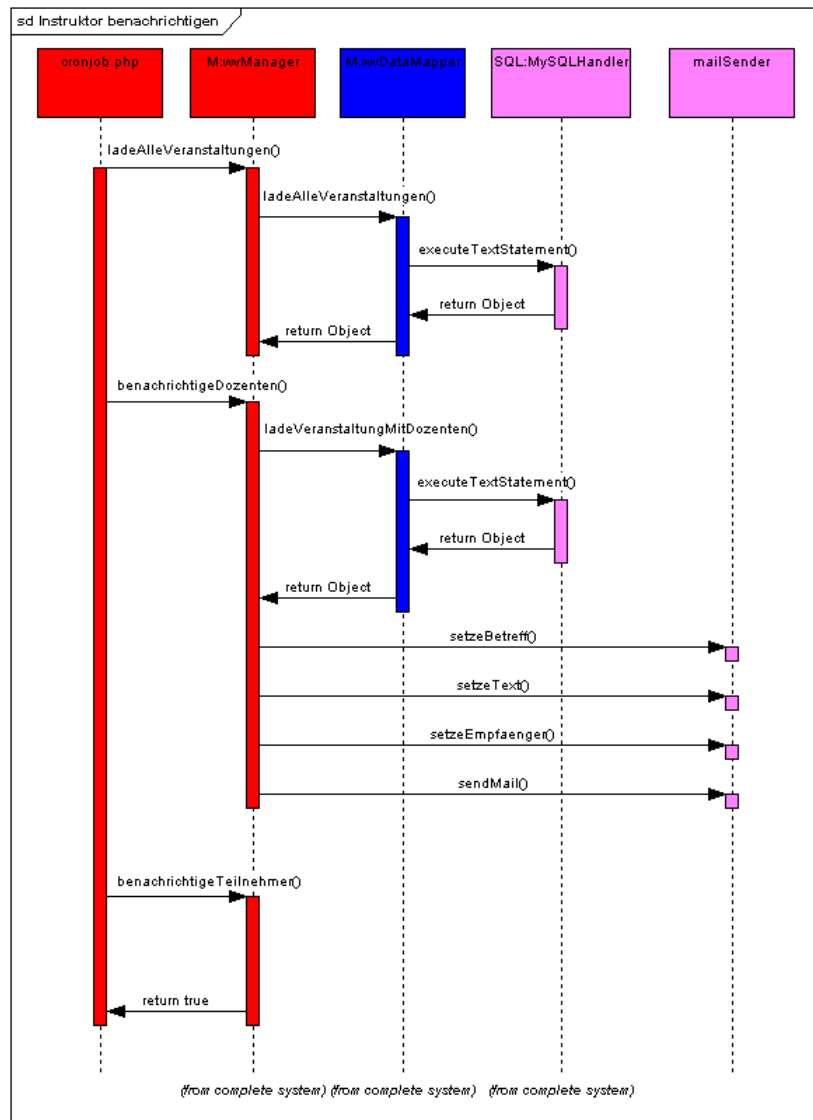


Abb. 6-27: Sequenz-Diagramm zum Use Case „Instruktor benachrichtigen“



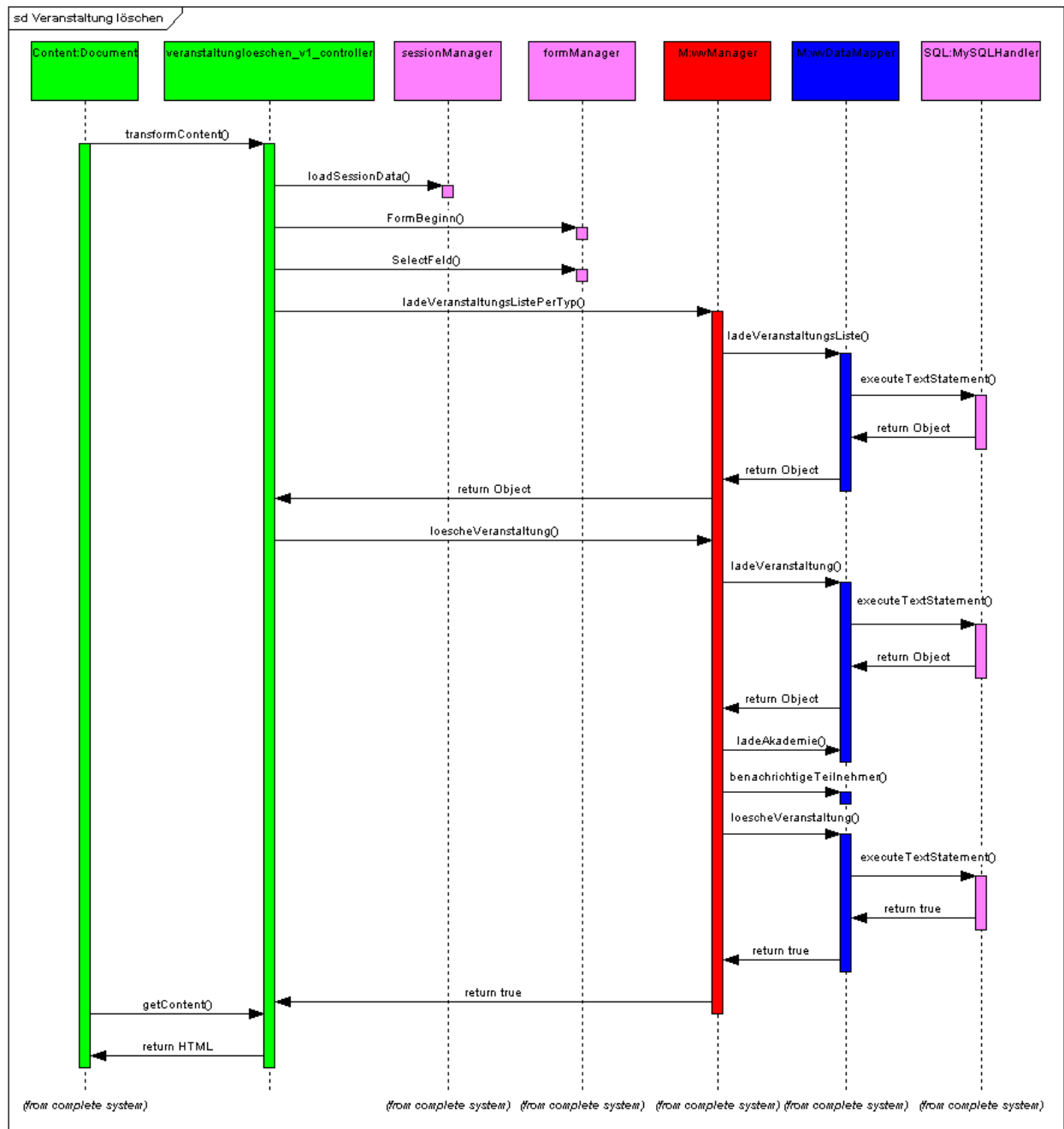


Abb. 6-28: Sequenz-Diagramm zum Use Case „Veranstaltung löschen“

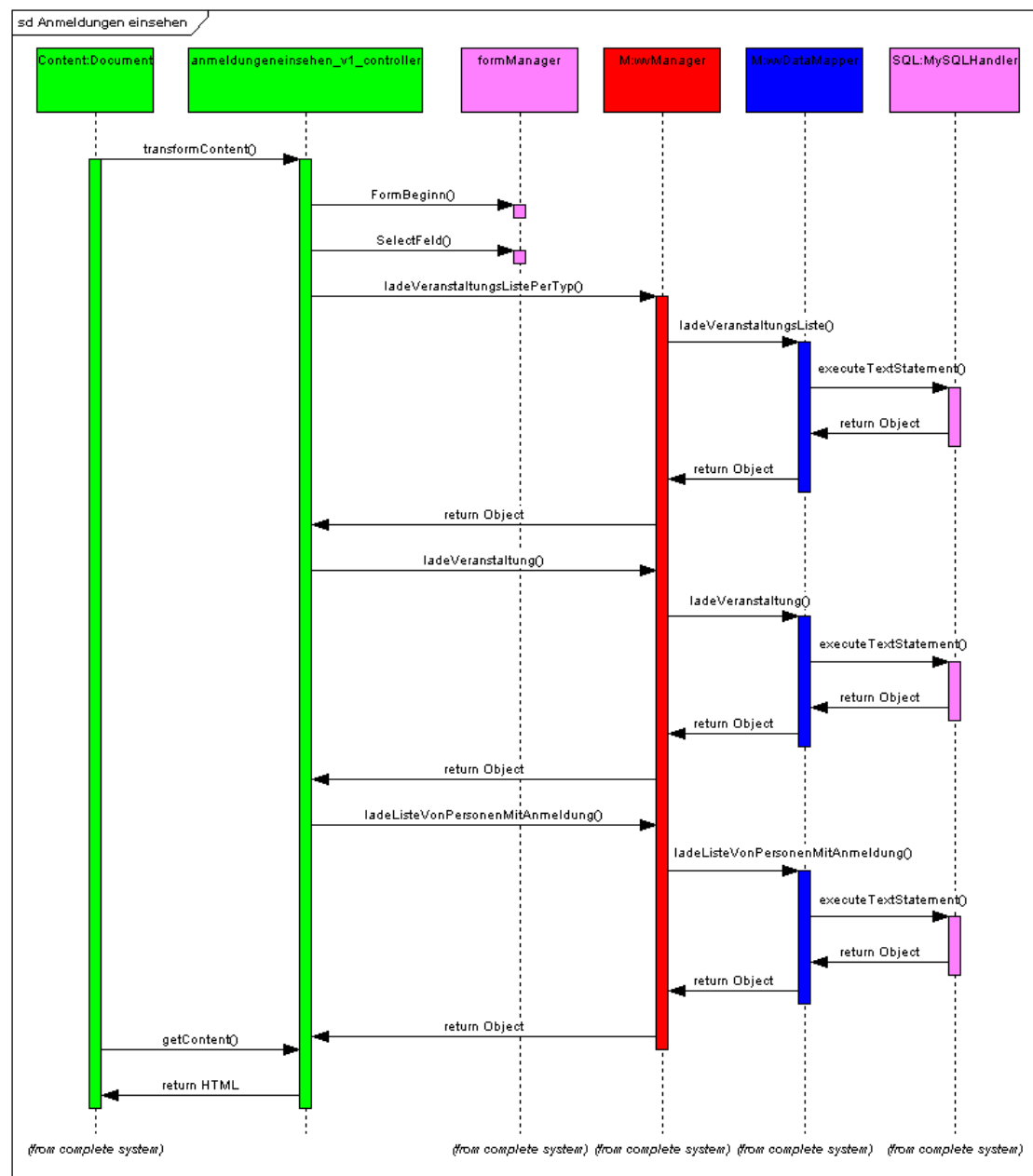


Abb. 6-29: Sequenz-Diagramm zum Use Case „Anmeldungen einsehen“

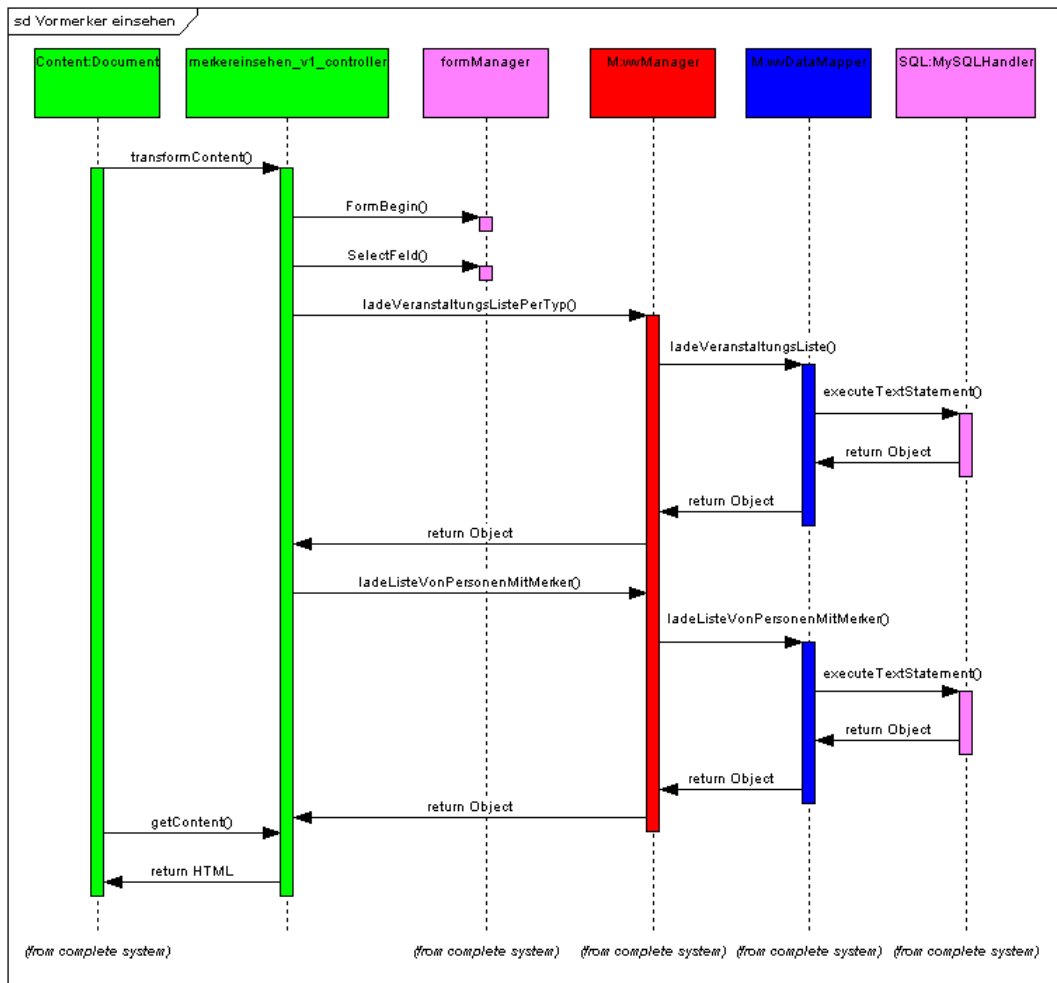


Abb. 6-30: Sequenz-Diagramm zum Use Case „Vormerker einsehen“

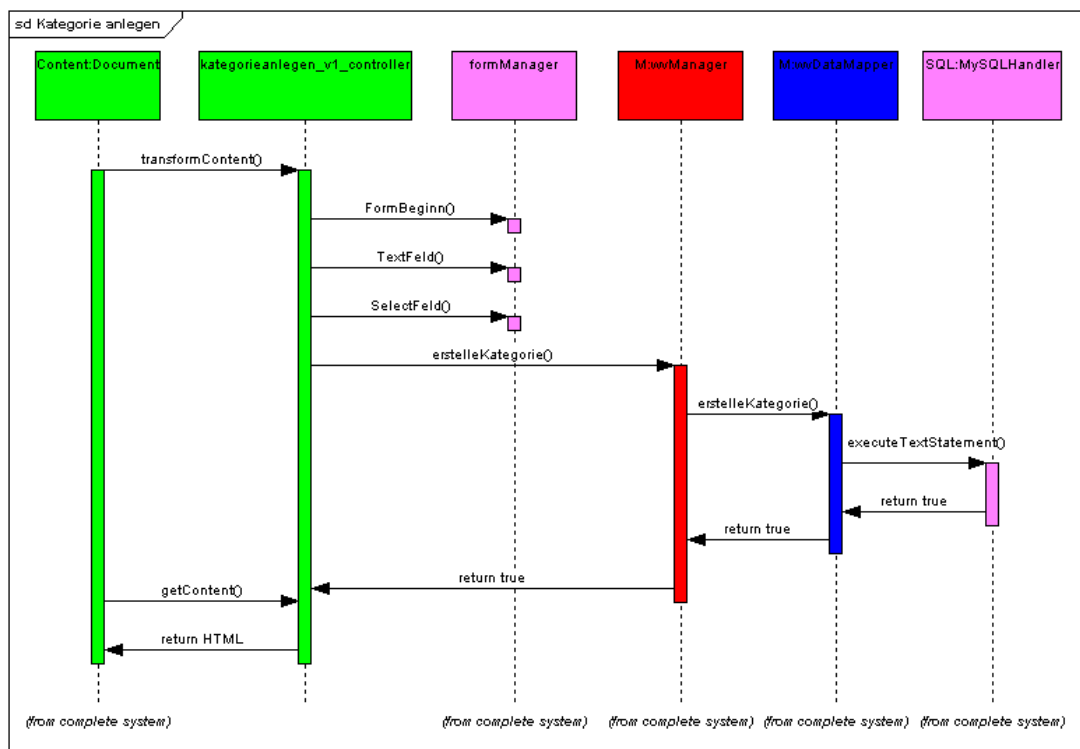


Abb. 6-31: Sequenz-Diagramm zum Use Case „Kategorie anlegen“

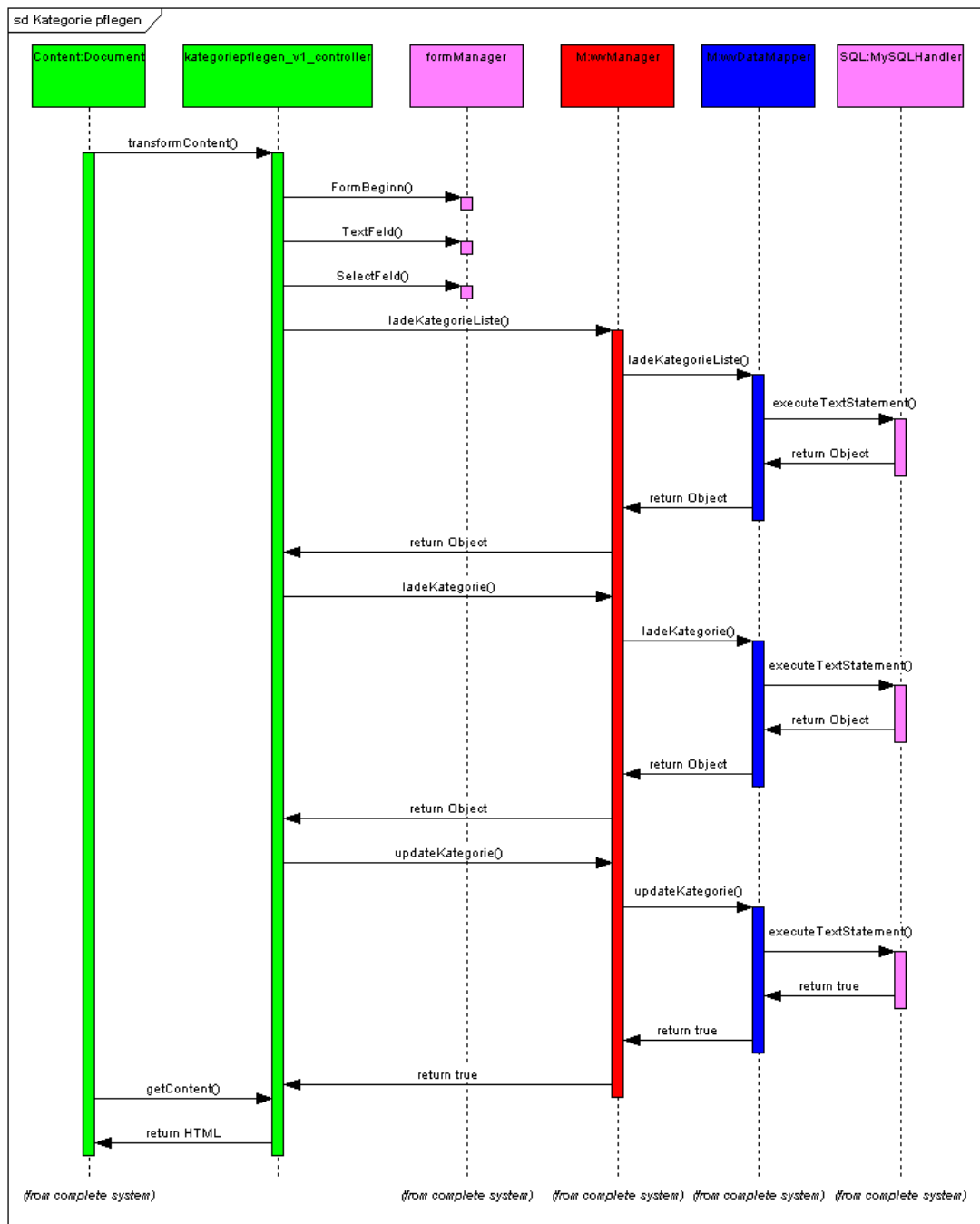


Abb. 6-32: Sequenz-Diagramm zum Use Case „Kategorie pflegen“

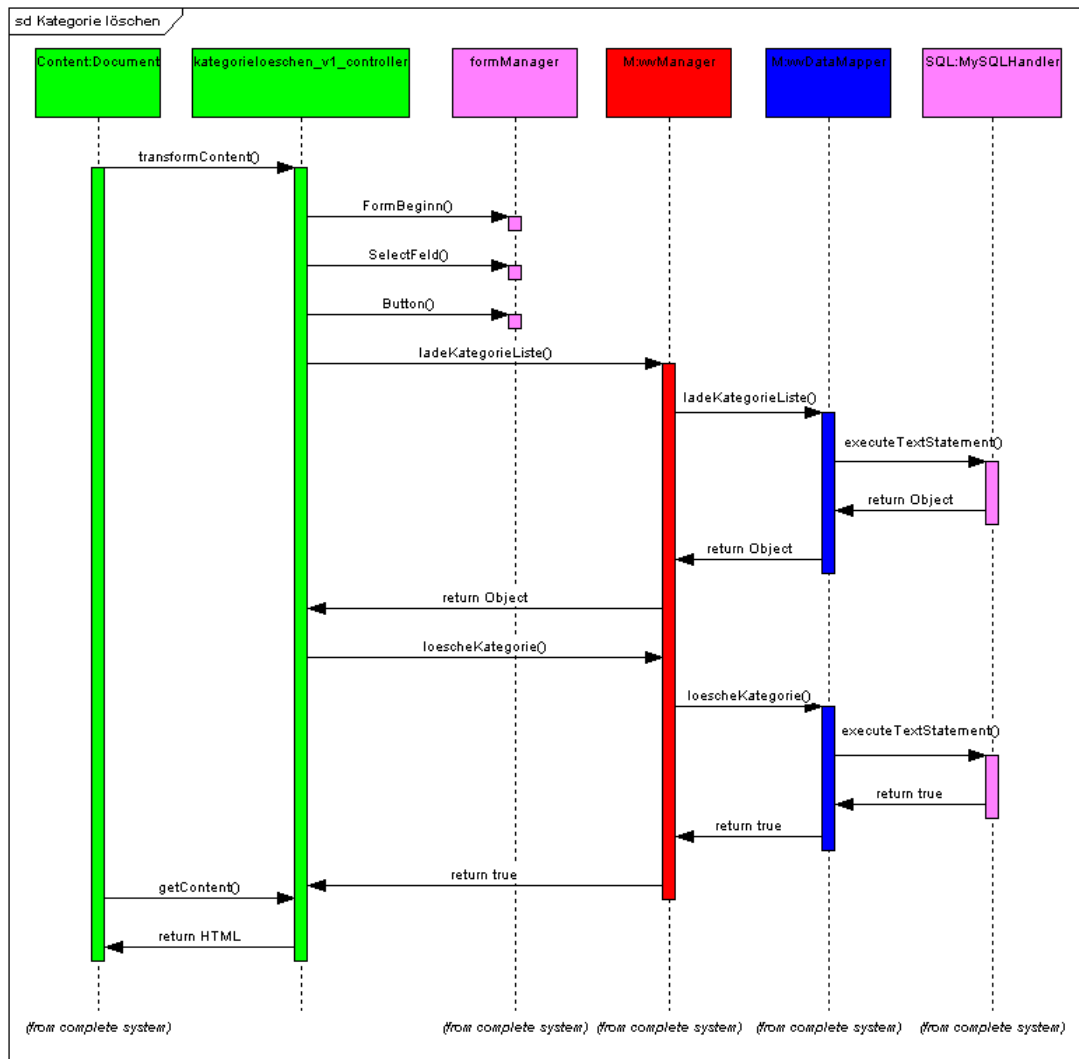


Abb. 6-33: Sequenz-Diagramm zum Use Case „Kategorie löschen“

## 6.4 Statistiken

Die folgende Tabelle zeigt einige wichtige Kennzahlen der Applikation. An Hand der hohen Zahl der Ordner und Quellcode- oder Konfigurations-Dateien zeigt sich der Strukturierungsgrad der Software.

Bereich	Anzahl
Anzahl der PHP-Dateien	81
Anzahl der Code-Zeilen (LOC)	~20000
Anzahl der Konfigurations-Dateien	20
Anzahl der HTML-Templates	39
Anzahl der Ordner	93

Tab. 6-c: Quellcode-Statistiken

## 7 Installationsanleitung

Kapitel 8 behandelt die Installation der Software. Mit Hilfe dieser Anleitung kann die Beispiel-Akademie auf dem Virtual-Host-Server [www.w3service.net](http://www.w3service.net) eingerichtet werden. Voraussetzung ist, dass die Person, die das Setup vornimmt, Kenntnisse in **HTML/XML**, **FTP** und **PHPMyAdmin** besitzt. Zudem ist es hilfreich die **PHP-Syntax** in Grundzügen zu verstehen. Hinweise zu diesen Themen befinden sich im Literaturverzeichnis.

### 7.1 Aufbau des Installationsverzeichnisses

Auf der Begleit-CD dieses Dokuments befindet sich ein Installations-Verzeichnis, das alle zur Installation benötigten Dateien in mehreren Unterordnern beinhaltet.

- (1) *akademie1/*: beinhaltet alle Ordner und Dateien, die zum Betrieb der Beispiel-Akademie notwendig sind.
- (2) *apps/*: beinhaltet alle Programm-Dateien und Konfigurationen, die für den Betrieb der Software notwendig sind.
- (3) *datenbank/*: beinhaltet das Datenbank-Setup- und Initialisierungs-Skript.

### 7.2 Installation

Die Installation ist in verschiedene Bereiche unterteilt, in denen unterschiedliche Teilaufgaben erledigt werden. Die in diesem Kapitel aufgeführten Schritte sind „best practices“, die aus Installations-Tests mit unterschiedlichen Probanden resultieren. Es wird empfohlen, die Installation in der unten aufgeführten Reihenfolge vorzunehmen.

#### 7.2.1 Datenbank-Initialisierung

In den folgenden Schritten wird die Datenbank für die spätere Verwendung initialisiert. Dabei werden die notwendigen Tabellen erstellt und ein initialer Datenbestand eingespielt.

- (1) Nehmen Sie die Zugangsdaten Ihrer MySQL-Datenbank zur Hand und loggen Sie sich auf dem PHPMyAdmin ihres Webpace-Anbieters ein. Das Datenbank-Verwaltungstool wird häufig standardmäßig unter einer nur für den Nutzer bekannten URL zur Verfügung gestellt. Sollten Sie diese Möglichkeit nicht haben, so setzen Sie sich mit Ihrem Provider in Verbindung.
- (2) Erstellen Sie entweder über die Konfigurations-Oberfläche ihres Providers oder über den PHPMyAdmin eine neue Datenbank. Sollten Sie keine Berechtigung haben eine Neue zu erstellen, so können Sie die Applikation auch in einer bestehenden Datenbank installieren. Es ist jedoch darauf zu achten, dass keine Tabellen mit dem Namen

*ass\_akademie\_person, ass\_kategorie\_veranstaltung, ass\_person\_anmeldung, ass\_person\_merker, ass\_person\_veranstaltung, comp\_akademie\_kategorie, comp\_akademie\_veranstaltung, comp\_person\_firmen\_adr, comp\_person\_privat\_adr, ent\_adresse, ent\_akademie, ent\_kategorie, ent\_person* oder *ent\_veranstaltung* existieren.

- (3) Spielen Sie das Installations-Skript *init\_db.sql* aus dem Verzeichnis *datenbank/* der Begleit-CD via PHPMyAdmin in die Datenbank ein und prüfen Sie, ob das Skript erfolgreich ausgeführt wurde. Falls Fehler vorliegen sollten Sie die Version der Datenbank sowie die bestehenden Tabellen prüfen. Die Software wurde für Version 4.0.x der MySQL-Datenbank geschrieben und ist aufwärtskompatibel.

## 7.2.2 Applikations-Initialisierung

Die Applikations-Initialisierung dient der Einrichtung einer Akademie nach Ihren inhaltlichen Vorgaben. Diese Angaben können Sie durch wiederholen dieser Schritte jederzeit verändern.

- (1) Navigieren Sie dazu im PHPMyAdmin zur Tabelle *ent\_akademie*.
- (2) Wählen sie den Reiter „Anzeigen“ und klicken Sie auf den vor dem Eintrag stehenden Stift um den Datensatz zu bearbeiten.
- (3) Tragen Sie in das Feld „Name“ den für die Akademie gewünschten Namen, in Beschreibung die korrekte Beschreibung der Akademie, in URL die URL ein, unter der die Akademie später zu erreichen sein wird.
- (4) Bestätigen Sie den Schritt mit „OK“.

## 7.2.3 Konfiguration der Programm-Dateien

Es wird bei der folgenden Anleitung davon ausgegangen, dass die Software unter der der URL <http://www.beispiel.tld/akademie1/> verfügbar sein wird. Dabei ist *beispiel.tld* ein Platzhalter für die von Ihnen betreute Domain. Der Pfad *akademie1/* impliziert, dass im Document-Root der Webseiten-Präsenz der Ordner *akademie1* erstellt werden muss.

- (1) Kopieren Sie die Dateien im Installations-Ordner *apps/* auf der Begleit-CD zur Bearbeitung in einen lokalen Ordner Ihres Systems.
- (2) Wechseln Sie in den Ordner *apps/config/core/database/weiterbildungsveranstaltung/iniconfig/* und öffnen Sie bitte die Datei *LIVESERVER\_dbconnectiondaten.txt*.
- (3) Tragen Sie die Zugangsdaten der Datenbank Ihres Providers in die dafür vorgesehenen Platzhalter ein. Relevant sind *Host* (Server-Adresse des MySQL-Servers), *Datenbank* (Name der Datenbank), *Benutzer* (Datenbank-Benutzer) und *Passwort* (Passwort des Datenbank-Benutzers). *DebugMode* sollte für den Installations-Test mit „1“ gefüllt

werden, da so zusätzliche Debug-Ausgaben angezeigt werden. Für den Live-Betrieb ist dieser Parameter sinnvollerweise auf „0“ zu setzen.

- (4) Speichern und schließen Sie die Datei und wechseln Sie in den Ordner *apps/config/core/mailexchanger/weiterbildungsveranstaltung/iniconfig/*. Dort öffnen Sie bitte die Datei *LIVESERVER\_mailexchanger.txt*.
- (5) Tragen Sie in den jeweiligen Sektionen die gewünschten Werte für *AbsenderName*, *AbsenderAdresse* und *ReplyAdresse* ein. Das Attribut *DebugMode* verhält sich in diesem Fall ähnlich und sollte daher im Live-Betrieb auf „0“ gesetzt bleiben. Bitte beachten Sie, dass die dort aufgeführten Sektionen für den fehlerfreien Betrieb der Anwendung erhalten bleiben müssen.
- (6) Speichern und schließen Sie die Datei und wechseln Sie in den Ordner *apps/config/core/applicationmanager/akademie1/iniconfig*. Dort öffnen Sie bitte die Datei *STANDARD\_applicationmanager.txt*.
- (7) Diese Datei ist die zentrale Konfigurations-Datei der Applikation. Tragen Sie in der Sektion „Standard“ unter URL diejenige URL ein, unter der die Akademie erreichbar sein soll. In diesem Beispiel <http://www.beispiel.tld/akademie1>.
- (8) Als weiterer Schritt müssen nun die in der Sektion *IncludeConfig* aufgeführten PHP-Konfigurations-Dateien angepasst werden. Derzeit sind dort die Konstanten
  - *AKADEMIE\_NO*: ID der aktuellen Akademie
  - *LOG\_PATH*: absoluter Pfad zum Log-Verzeichnis
  - *MEDIA\_PATH*: absoluter Pfad zu den Media-Dateien der Akademie
  - *BACKUP\_\_FS\_PATH*: absoluter Pfad zum Backup-Verzeichnis
  - *BACKUP\_\_URL\_PATH*: URL zum Backup-Verzeichnis
  - *STATEMENT\_CACHE\_DIR*: absoluter Pfad zum Statement-Cache-Verzeichnis
  - *CACHE\_DIR*: absoluter Pfad zum Cache-Verzeichnis

definiert. Setzen Sie diese bitte in den Dateien *LIVESERVER\_config.php* und *LIVESERVER\_cacheconfig.php* im Verzeichnis *apps/config/core/applicationmanager/akademie1/phpconfig* auf den für die Akademie korrekten Wert. Es kann dabei immer vom Basis-Verzeichnis des Web-Accounts ausgehend der Verzeichnis-Pfad eingetragen werden.

- (9) Speichern und schließen Sie diese Dateien und wechseln Sie in das Verzeichnis *apps/config/sites/weiterbildungsveranstaltung/wvmanager/weiterbildungsveranstaltung/iniconfig/*.
- (10) Öffnen Sie dort die Datei *LIVESERVER\_debugconfig.txt* und tragen Sie den gewünschten *DebugLevel* ein. Dieser Konfigurations-Parameter bestimmt, welche Aktionen der Oberfläche in die Log-Dateien geschrieben werden. Weitere Hinweise dazu finden Sie in der Konfigurations-Datei.
- (11) Speichern und schließen Sie diese Datei und wechseln Sie in das Verzeichnis *apps/config/sites/weiterbildungsveranstaltung/cronjob/weiterbildungsveranstaltung/in*



*iconfig*.

- (12) Dort öffnen Sie bitte die Datei *LIVESERVER\_backupcronjob.ini* und tragen in der Sektion *Backup* unter der Konfiguration Recipients diejenigen Personen ein, welche die Bestätigungs-E-Mail des Backup-Cronjobs erhalten sollen. Mehrfacheintragungen sind durch ein Semikolon, Name und E-Mail durch ein Pipe zu trennen.
- (13) Speichern und schließen Sie diese Datei. Die Basis-Konfiguration ist damit abgeschlossen.

## 7.2.4 Installation der Programm-Dateien

Nun werden die Programm-Dateien auf dem Webserver Ihres Providers installiert. Halten Sie dazu einen FTP-Client ihrer Wahl bereit und stellen Sie die Verbindung mit dem Internet her.

- (1) Verbinden Sie den FTP-Clienten ihrer Wahl mit dem Webserver Ihres Providers.
- (2) Übertragen Sie den zuvor entpackten Ordner *apps/* in das Root-Verzeichnis Ihres Accounts. Es ist darauf zu achten, dass der Ordner *apps/* eine Datei mit Namen *.htaccess* enthält. Diese verhindert, dass der Ordner via HTTP erreichbar ist und Angreifer Konfigurationen und Zugangsdaten auslesen können!
- (3) Übertragen Sie nun den Ordner *akademie1/* aus dem Installations-Verzeichnis der Begleit-CD ebenfalls in das Root-Verzeichnis.
- (4) Navigieren Sie in Ihrem FTP-Programm in den Ordner *akademie1/* und geben Sie dem Ordner *logs/* die Datei-Attribute **775**. Dies bezweckt, dass die Applikation in diesem Verzeichnis Log-Dateien ablegen kann. Verfahren Sie ebenso mit dem Ordner *akademie1/cache/statements/*, *akademie1/cache/imageresizer/* und *akademie1/backup/*. Der Ordner *statements/* beinhaltet später die Cache-Dateien der Anwendung, *imageresizer/* die Cache-Dateien des Bildanzeige-Moduls und *backup/* die Sicherungs-Dateien, die via Oberfläche angefertigt werden.
- (5) Die Installation ist mit diesem Schritt abgeschlossen und Sie können die Akademie unter <http://www.beispiel.tld/akademie1> aufrufen. Auf der dort gezeigten Oberfläche können Sie sich mit dem vordefinierten Benutzer „root“ und dem Passwort „root“ einloggen. Da dieses Passwort kein sicheres Passwort ist wird empfohlen, dieses nach dem Login auf ein nur Ihnen Bekanntes zu ändern. Hierzu verwenden Sie bitte den Menüpunkt „Zugangsdaten bearbeiten“.

## 7.2.5 Einrichten der Cronjobs

Um die Cronjob-Funktionalität nutzen zu können muss dafür Sorge getragen werden, dass die beiden Skripte in der gewünschten Häufigkeit via http aufgerufen werden. Eine einfache und bereits beschriebene Variante ist es die relevanten URLs in einem Cronjob-Anbieter wie cronjob.de einzutragen. Es wird folgende Konfiguration empfohlen:

URL	Aufrufhäufigkeit
<a href="http://www.beispiel.tld/akademie1/backupcronjob.php">http://www.beispiel.tld/akademie1/backupcronjob.php</a>	1x pro Tag gegen 01:00Uhr
<a href="http://www.beispiel.tld/akademie1/cronjob.php">http://www.beispiel.tld/akademie1/cronjob.php</a>	1x pro Woche zu den üblichen Bürozeiten

**Tab. 8-a: Aufrufhäufigkeit der Cronjob-Skripte**

## 7.2.6 Sicherung der Konfiguration

Um bei evtl. auftretenden Fehlern eine Sicherung der Konfiguration zu haben, wird empfohlen, die auf dem Webserver vorhandenen Dateien und Ordner, sowie die Datenbank zu sichern. Dies erledigen Sie dadurch, dass Sie die in den Ordnern *apps/* und *akademie1/* vorhandenen Dateien und Ordner auf Ihren Rechner herunterladen und ein Archiv erstellen. Die Datenbank-Sicherung können Sie bequem über die Oberfläche der Software vornehmen. Wählen Sie dazu den Menüpunkt *Datenbank sichern* und klicken Sie auf *Sicherung starten*. Die im Anschluss angezeigte Status-Meldung beinhaltet einen Link zur angefertigten Sicherungsdatei. Laden Sie diese entweder über die Oberfläche oder via FTP aus dem zuvor konfigurierten Backup-Verzeichnis herunter und speichern Sie diese zusammen mit den übrigen Dateien auf ein Drittmedium. Um regelmäßige Sicherungen zu erstellen, ist es lediglich notwendig den unter 6.2.5. beschriebenen Backup-Cronjob zu aktivieren und die Sicherungs-Dateien per FTP herunterzuladen. Weitere Sicherungsmaßnahmen sind nicht notwendig, da die Dateien auf dem Dateisystem nur Programm-Code und Konfigurationen enthalten.

## 7.3 Anlegen einer weiteren Akademie

Um eine weitere Akademie zu erstellen sind tiefere Eingriffe in das System notwendig. Es wird empfohlen, die Ersteinrichtung einer zusätzlichen Akademie mit dem Verfasser der Software vorzunehmen, da sowohl erweiterte Programmier-Kenntnisse als auch ein erweitertes Verständnis für das Design der Anwendung vorausgesetzt werden.

## 8 Ausblick

Im Verlauf der Software-Erstellung und des Reviews wurden von den beteiligten Personen weitere Ideen gesammelt, die teilweise in die Umsetzung mit eingeflossen sind oder für die Bereitstellung in weiteren Versionen der Software notiert wurden. Die wichtigsten Features sein an dieser Stelle nochmals hervorgehoben:

<b>Anlegen und Editieren von Benutzer-Accounts</b>	Ein Administrator oder Akademie-Manager sollte ein vorhandenes Benutzer-Profil direkt bearbeiten, bzw. ein neues außerhalb der aktuell implementierten Registrierung anlegen können. Es ist jedoch darauf zu achten, dass ein Benutzer mit administrativen Rechten aus Sicherheitsgründen nie ein Konto mit höheren Rechten als er selbst anlegen darf.
<b>Deaktivieren von Benutzer-Accounts</b>	Benutzer-Accounts sollten mit einem Status versehen werden. Dieses zusätzliche Merkmal kann dazu genutzt werden, Konten zeitweise zu deaktivieren und gelöschte Accounts noch einige Zeit zu Reaktivierungszwecken aufzubewahren. Sinnvollerweise wird das Bereinigen von länger inaktiven, bzw. logisch gelöschten Accounts automatisiert.
<b>Bestätigung von Registrierungen</b>	Registrierungen sollten erst nach Versand einer Validierungs-E-Mail und deren Bestätigung vom Benutzer als „valide“ signiert werden. Dadurch kann der Missbrauch von E-Mail-Adressen verhindert werden. So können nicht korrekt getätigte Registrierungen nach einer gewissen Zeitspanne automatisiert bereinigt werden.
<b>Validierung von E-Mail-Adressen</b>	Einhergehend mit der Validierung von Benutzernamen sollte auch eine E-Mail-Adresse auf Eindeutigkeit im System geprüft werden.
<b>Passwort-Rücksetzen</b>	Der bisherige Workflow soll durch ein automatisiertes Zusenden einer Mail mit einem zufällig gesetzten Passwort ausgetauscht werden, um dem Benutzer auch ohne das Zutun eines Administrators das Rücksetzen seines Passworts zu ermöglichen.
<b>Gültigkeitsstatus eines Accounts</b>	Ein Account sollte mit einem Gültigkeitsdatum versehen werden. Hat ein Benutzer innerhalb dieses Zeitraums keinen erfolgreichen Login, wird der Account auf inaktiv gesetzt und eine Mail zur Erinnerung verschickt. Zur Sicherheit muss der Benutzer gezwungen werden, sein Passwort alle x Monate zu ändern.
<b>Wiederverwendung von Passwörtern</b>	Passwörter sollten bei der Änderung derselben nicht wieder verwendet werden können. Es muss eine Prüfung implementiert werden, die die bisher eingesetzten Passwörter prüft und den Benutzer zwingt ein anderes einzugeben.
<b>RSS-Feed</b>	Um die auf den Akademien angebotenen Kurse zu Werbezwecken auf anderen Webseiten darstellen zu können, wurde eine RSS-Feed-Funktion gewünscht. Mit Hilfe dieses Austausch-Formates können auf Drittseiten News-Ticker, oder ähnliche Werbe-Ticker standardisiert gefüllt werden.

## 9 Literatur- und Abbildungs-Verzeichnis

### 9.1 Literaturverzeichnis

Dieses Kapitel beinhaltet die verwendeten Literaturstellen und zeigt die wichtigsten Bezugsquellen für Quellcodes und Programme auf:

**[1] Fowler, Martin: Patterns of Enterprise Application Architecture**

Buch über Software-Design, Nachschlagewerk für Design-Pattern und Implementierungshilfen.

**[2] Wikipedia**

Freie Enzyklopädie, gutes Nachschlagewerk für Begriffserklärungen. Weitere Informationen unter <http://de.wikipedia.org>

**[3] RUP**

Das bei der Erstellung der Software verwendete iterative Modell ist Teil, bzw. ein Ausschnitt des *Rational Unified Processes* von Rational Rose. Nähere Informationen dazu in einschlägiger Fachliteratur zu RUP, bzw. unter <http://www.rational.com>.

**[4] Doxygen**

Das Quellcode-Dokumentations-Werkzeug *doxygen* generiert eine HTML -Dokumentation aus einem dafür dokumentierten Quelltext. Quellcode und kompilierte Windows-Versionen sowie Hilfen finden sich unter <http://www.doxygen.org>.

**[5] SelfHTML**

SelfHTML ist eine der besten online verfügbaren HTML-Referenzen. Es dient als Nachschlagewerk für HTML-, XML- und CSS-Spezifikationen und ist hilfreich für Anwendungs-Beispiele. Die komplette Referenz, sowie ein Diskussionsforum findet man unter <http://de.selfhtml.org>.

**[6] PHPMyAdmin**

State-of-the-art Administrations-Tool für MySQL-Datenbanken. Es wird sowohl zur Entwicklung als auch zur Installation verwendet. Installations-Packages und Anleitungen sind unter <http://phpmyadmin.org> downloadbar.

**[7] Apache HTTP-Server**

Der am weitesten verbreitetste Webserver auf UNIX/Linux. Die Webseite dient als Referenz für die Installation und den Betrieb des Webserver. Dokumentation und Windows-Binaries sind ebenso unter <http://httpd.apache.org> erhältlich.

**[8] PHP**

Auf php.net befinden sich die komplette Dokumentation der Skriptsprache, sowie mehrere Foren. Sourcen und Binaries für PHP sind ebenso auf <http://www.php.net> downloadbar.

[9] **MySQL**

Auf der Webseite <http://www.mysql.com> findet sich eine komplette Referenz und Bezugsquellen für die MySQL-Datenbank.

[10] **Microsoft HH-Compiler**

Bezugsquelle für den HTML-Help-Compiler, mit dessen Hilfe chm-Dateien für die Quellcode-Dokumentation erstellt wurden. Weitere Informationen zu HTML-Help-Files befinden sich im MSDN-Bereich von <http://www.microsoft.com>.

[11] **Sparx Systems**

Das UML Entwicklungstool *Enterprise Architect* wurde für Design, Dokumentation und Code-Generation eingesetzt. Produktinformationen finden sich unter <http://www.sparxsystems.com.au>.

## 9.2 Abbildungsverzeichnis

Alle Abbildungen und Grafiken wurden während der Erstellung der Arbeit selbständig angefertigt. Auf die Einbindung von Fremdmaterial wurde bis auf die Produkt-Logos der eingesetzten Server-Software in Gänze verzichtet.

## 10 Lizenzbestimmungen

Die im Rahmen der vorliegenden Arbeit erstellte Software unterliegt der GPL-Lizenz. Sie darf frei kopiert, verteilt und verändert werden. Weiterentwicklungen bzw. Anpassungen, die nicht vom Autor vorgenommen wurden, müssen öffentlich zugänglich gemacht werden. Weitere Einzelheiten zur eingesetzten Lizenz können der Datei *gpl.txt* im Verzeichnis *Lizenzinformationen/* auf der Begleit-CD entnommen werden.

## Schluss-Erklärung

Gemäß den Gestaltungshinweise der Fachhochschule Würzburg-Schweinfurt, Abteilung Schweinfurt, Fachbereich Elektrotechnik, wird folgende Erklärung abgegeben:

*„Hiermit erkläre ich, dass ich die vorliegende Diplomarbeit selbständig verfasst und noch nicht anderweitig für Prüfungszwecke vorgelegt habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und habe wörtliche oder sinngemäße Zitate als solche gekennzeichnet“.*

Die Richtigkeit dieser Angaben wird durch die folgende Unterschrift bestätigt:

---

Christian Schäfer (Diplomand)