Umsetzung eines Internetauftritts unter Nutzung des Content-Management-Systems „TYPOlight“

D i p l o m a r b e i t

an der Hochschule Hof
Fachbereich Informatik/Technik
Studiengang Medieninformatik

Vorgelegt bei
Prof. Dr. Heym
Alfons-Goppel-Platz 1
95028 Hof

Vorgelegt von
Marcus Kliche
Ortsstraße 25
07924 Crispendorf

Hof, 10.06.2008

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
BSI	Bundesvereinigung Spitzenverbände der Immobilienwirtschaft
CSS	Cascading Style Sheets
DAM	Digital Asset Management
DCA	Data Container Array
DDIV	Dachverband Deutscher Immobilienverwalter
e.V.	eingetragener Verein
EFG	Extended Form Generator
FTP	File Transfer Protocol
GPL	General Public License
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identification
IP	Internet Protocol
LGPL	Lesser General Public License
MVC	Model View Controller
PDF	Portable Document Format
PHP	PHP: Hypertext Preprocessor
RSS	Really Simple Syndication
SQL	Structured Query Language
TAR	Tape Archive
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
VdIVM	Verband der Immobilienverwalter Mitteldeutschland
VdIVS	Verband der Immobilienverwalter Sachsen
VdIVT	Verband der Immobilienverwalter Thüringen
WYSIWYG	What You See Is What You Get
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
ZIP	Zigzag Inline Package

Allgemeine Begriffsabgrenzung

Backend

Als „Backend“ wird der Bereich einer Website bezeichnet, der zur Verwaltung der Struktur und der Inhalte dieser Website dient. Dieser Bereich ist in der Regel nur ausgewählten Personen über die Eingabe eines Benutzernamens und eines Passworts zugänglich und wird durch die Verwendung eines Content-Management-Systems bereit gestellt.

Backend-Benutzer

Der Begriff „Backend-Benutzer“ bezeichnet die Rolle einer Person, welche sich durch Eingabe eines Benutzernamens und eines Passworts am Backend einer Website angemeldet hat und Zugang zu verschiedenen Bereichen des zu Grunde liegenden Content-Management-Systems hat. Backend-Benutzer können sich in der Regel nicht selbst registrieren, um Zugang zum Backend zu erhalten, sondern müssen stattdessen von einem Administrator des Systems angelegt werden.

Frontend

Der Begriff „Frontend“ wird in der Regel nur bei Websites verwendet, denen ein Content-Management-System zu Grunde liegt. Er bezeichnet den Bereich einer Website, welcher öffentlich zugänglich ist und zur Präsentation von Inhalten oder zur Bereitstellung bestimmter Funktionen für Besucher dieser Website dient. Im Frontend können jedoch auch geschützte Bereiche existieren, die nur durch Eingabe eines Benutzernamens und eines Passworts erreichbar sind.

Frontend-Mitglied

Als „Frontend-Mitglied“ wird die Rolle eines Besuchers einer Website bezeichnet, welcher sich durch Eingabe eines Benutzernamens und eines Passworts am Frontend dieser Website angemeldet hat und dadurch Zugang zu geschützten Bereichen erhält. Frontend-Mitglieder können sich in der Regel selbstständig auf einer Website registrieren, wobei die Registrierung teilweise auch erst durch einen Backend-Benutzer bestätigt werden muss.

Webseite

Als „Webseite“ wird ein einzelnes HTML-Dokument bezeichnet, welches unter einer spezifischen Adresse im Internet erreichbar ist und entweder dynamisch generiert wird oder als statische Datei vorliegt.

Website

Der Begriff „Website“ bezeichnet die Gesamtheit aller Webseiten, die zu einem Internetauftritt gehören und durch Hyperlinks miteinander verknüpft sind.

1 Einleitung

Mitte der 1990er Jahre begann das Internet sich als Standard zur Verbreitung von Informationen jeglicher Art zu etablieren. Im Zuge dieser Entwicklung fanden auch viele Unternehmen und Privatpersonen den Weg zu diesem neuen Medium, um sich selbst öffentlich zu präsentieren.

Zur Realisierung der Selbstdarstellung der eigenen Person oder der eigenen Firma boten sich zur damaligen Zeit, und bieten sich auch heute noch, im Wesentlichen zwei Möglichkeiten an: der Internetauftritt konnte in Eigenarbeit umgesetzt werden oder einer professionellen Firma in Auftrag gegeben werden.

Zur Umsetzung der ersten Variante waren Grundkenntnisse in HTML und später auch in CSS erforderlich. Für Benutzer ohne jegliche HTML- oder Programmierkenntnisse kamen 1996 erste clientseitige Systeme, wie z.B. Frontpage von Microsoft, zur Erstellung von Webseiten auf den Markt. Ein großer Nachteil dieser Systeme bestand darin, dass nur statische Inhalte erzeugt werden konnten und im Allgemeinen nur die visuelle Bearbeitung eines Internetauftritts möglich war. Weiterhin erzeugten diese Systeme in der Regel einen sehr unsauberen Quellcode, der keine Trennung von Inhalt und Layout berücksichtigte und meist auch nicht den standardisierten Regeln zur Umsetzung von Webseiten folgte, soweit solche schon vorhanden waren.

Wurde die zweite Möglichkeit in Betracht gezogen, also die Entwicklung und Pflege des Internetauftritts durch eine professionelle Firma oder Agentur, ergaben sich viele Vorteile im Vergleich zur Umsetzung in Eigenarbeit, allerdings auch einige Nachteile. Vorteilhaft waren unter anderem die besseren technischen Voraussetzungen wie z.B. sauberer Quellcode oder die Einhaltung von Web-Standards, welche durch einen professionellen Umsetzer geschaffen wurden. Nachteilig für den Auftraggeber, also in der Regel den Eigentümer des geplanten Internetauftritts, waren die höheren Entwicklungs- und Wartungskosten, da jede Änderung über die Agentur erfolgen musste und somit Arbeitskosten verursachte.

Im Laufe der fortschreitenden Entwicklung des Internets stieg auch die Komplexität der Internetpräsenzen exponentiell an. Als Beispiel sei nur der Internetauftritt der Hochschule Hof genannt, der unter der Domain „fh-hof.de“ derzeit mit ca. 11 400 Seiten[1] bei Google gelistet ist. Für Unternehmen oder öffentliche Einrichtungen wäre die Wartung einer Website dieser Größenordnung durch eine Agentur nicht mehr tragbar. So wurde parallel zum Wachstum der Internetpräsenzen die Forderung von deren Betreibern nach Systemen zur Wartung ihrer Website ohne HTML- oder Programmierkenntnisse laut.

[1] Stand: 17.03.2008

Eines der ersten serverseitigen Content-Management-Systeme, welches diese Anforderungen erfüllte, war das seit 1997 von Kasper Skårhøj entwickelte Typo3.[2] Diesem folgten kurze Zeit später zahlreiche weitere Systeme, die sich stark in Funktionsumfang und Qualität unterschieden. Unter diesen Nachfolgern befindet sich auch das System TYPOlight, welches seit 2004 von Leo Feyer entwickelt wird und aktuell in der Version 2.5.8[3] erschienen ist.

TYPOlight soll nun auch zentrales Thema dieser Diplomarbeit sein und zur Umsetzung eines konkreten Projektes verwendet werden. Bei dem Projekt handelt es sich um den geplanten Internetauftritt des Verbandes der Immobilienverwalter Mitteldeutschland e.V. Dieser im Jahr 2007 gegründete Verband ist ein Zusammenschluss der Verbände der Immobilienverwalter Thüringens und Sachsens. Die beiden ursprünglichen Verbände besitzen jeweils schon einen Internetauftritt, welche allerdings beide sehr trivial und längst nicht mehr zeitgemäß erscheinen. Deshalb soll nun ein gemeinsamer, neuer und moderner Internetauftritt für den neuen Verband entstehen.

Daher soll in Kapitel 2 zunächst der Zustand der aktuellen Internetauftritte der Verbände analysiert werden und die Anforderungen an den geplanten, neuen Internetauftritt festgehalten werden.

Anschließend soll in Kapitel 3 TYPOlight mit Typo3 als Referenzsystem verglichen werden, wobei dessen Vor- und Nachteile im direkten Vergleich mit dem Konkurrenten ermittelt werden sollen. In diesem Rahmen soll auch ein Benchmark-Test zum Vergleich der Geschwindigkeit beider Systeme beim Ausliefern von Inhalten durchgeführt werden. Aus diesem Vergleich heraus soll die Wahl von TYPOlight für dieses konkrete Projekt begründet werden.

Danach wird in Kapitel 4 die schrittweise Umsetzung eines Internetauftritts mit TYPOlight beginnend bei der Installation über das Anlegen von Modulen und Seitenlayouts bis hin zum Aufbau der Seiten- und Artikelstruktur beschrieben.

Weiterhin soll in Kapitel 5 der grundsätzliche Aufbau von TYPOlight beschrieben und analysiert werden, da dessen Verständnis Voraussetzung für die Umsetzung eines Internetauftritts mit diesem System und für die Entwicklung von Extensions ist.

Das Thema der Extension-Programmierung soll danach in Kapitel 6 behandelt werden, da für den geplanten Internetauftritt ebenfalls einige Erweiterungen benötigt werden. Deren Konzeption, Implementierung und Test soll anschließend in Kapitel 7 beschrieben werden.

Zum Schluss sollen in Kapitel 8 alle Ergebnisse zusammengefasst und in Kapitel 9 ein Ausblick auf mögliche, weitere Funktionen für die Zukunft gegeben werden.

---

[2] Vgl. O.V.: Typo3 CMS: Geschichte, 2006, Internet: http://typo3.com/Geschichte.1268.0.html?&L=2
[3] Stand: 30.05.2008

2 Projektvorstellung

In diesem Kapitel wird der momentane Zustand der vorhandenen Internetauftritte der Verbände der Immobilienverwalter Thüringens und Sachsens analysiert und deren Schwächen aufgezeigt. Weiterhin werden die Anforderungen an den geplanten, neuen Internetauftritt dargestellt.

2.1 Momentaner Zustand (Ist)

Die beiden unabhängigen Verbände der Immobilienverwalter Thüringens und Sachsens schlossen sich im Jahr 2007 zum „Verband der Immobilienverwalter Mitteldeutschland e.V." zusammen. Beide Verbände besitzen bereits jeweils einen Internetauftritt, welcher allerdings längst nicht mehr zeitgemäß ist. Dieses Kapitel stellt gestalterische und technische Mängel in einem kurzen Überblick dar.

2.1.1 Gestalterische Mängel

Zunächst entspricht das Design beider Auftritte nicht mehr dem momentan verbreiteten Stil im Webbereich. Die Seiten wirken optisch wenig ansprechend und sehen teilweise sehr trivial gestaltet aus. Es ist keine klare Linie im Design erkennbar, es wurden keine einheitlichen Schriften und Farben verwendet. Auch die verwendeten Grafiken wirken eher willkürlich erstellt und ausgewählt, als mit Bedacht entworfen.

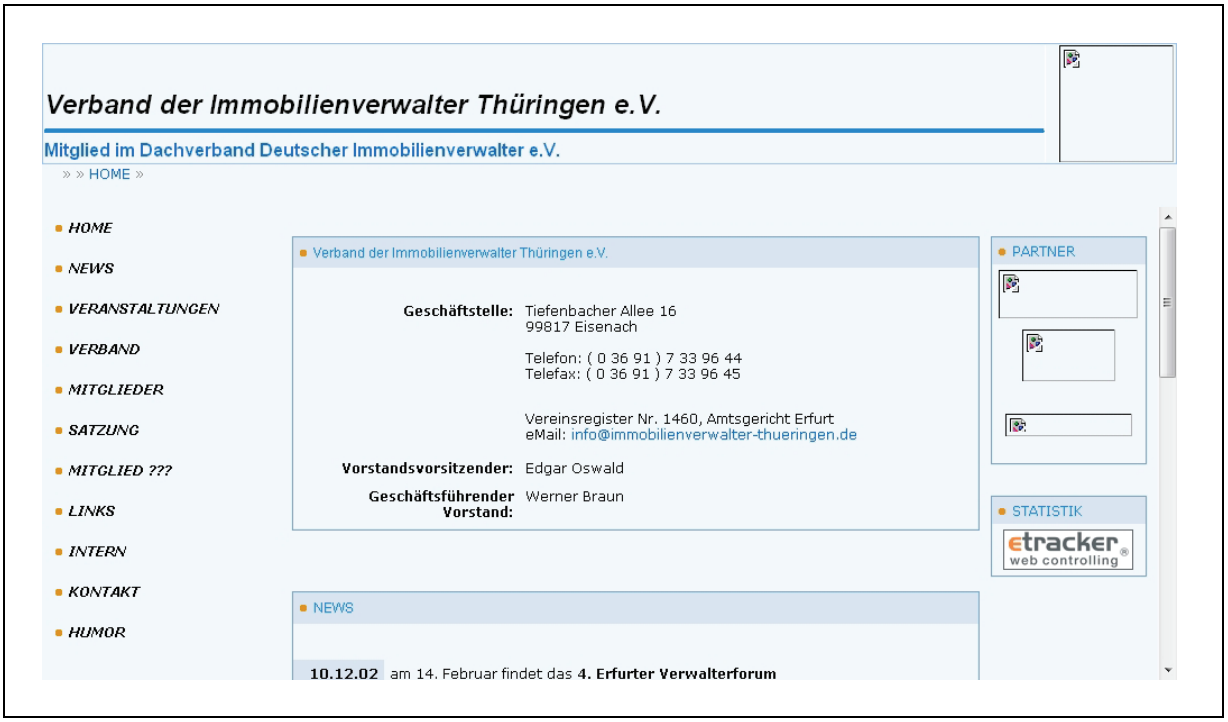

Abbildung 2-1: Ausschnitt der Startseite des VdIVT e.V. (fehlerhaft, siehe Text)

Abbildung 2-2: Ausschnitt der Startseite des VdIVS e.V.

Der Screenshot in Abbildung 2-1 wurde von der beim Internetarchiv „Waybackmachine“ archivierten Website des Verbandes der Immobilienverwalter Thüringen e.V.[4] erstellt, da die eigentliche Website zum Zeitpunkt der Erstellung dieser Diplomarbeit nicht mehr verfügbar war. Daher fehlen im Seitenabbild auch einige Grafiken, da diese im Archiv nicht enthalten waren.

Abbildung 2-2 zeigt im Gegensatz dazu ein Abbild der tatsächlichen Website des Verbandes der Immobilienverwalter Sachsen e.V., wie sie bis zur Erstellung des neuen Auftritts im Internet zu finden war.

2.1.2 Technische Mängel

In Ergänzung zu den gestalterischen Mängeln sind beide Internetauftritte nicht mehr auf einem aktuellen technischen Stand. Es steht kein Content-Management-System hinter den Websites, stattdessen liegen alle Unterseiten als statische HTML-Dateien vor. Eine regelmäßige Wartung ist somit nur sehr umständlich möglich.

[4] Zu finden unter http://web.archive.org/web/*/http://www.immobilienverwalter-thueringen.de

Als nächster Schwachpunkt ist zu nennen, dass beide Verbände ein Frameset zur Navigation durch ihre Seiten verwenden. Dieses bringt zusätzliche technische Nachteile und Barrieren mit sich. Zum einen wirken sich Frameseiten äußerst negativ auf die Suchmaschinenoptimierung einer Website aus, da auf der Startseite meist keine aussagekräftigen Inhalte im eigentlichen Sinne hinterlegt sind, sondern nur die Definition der verschiedenen Frames und die Verweise auf deren Unterseiten. Zum anderen verhindern Framesets das Setzen von Lesezeichen auf bestimmte Unterseiten, da vom Browser stets der Verweis auf die Startseite, welche das Frameset beinhaltet, gespeichert wird. Auch das Drucken einer Seite oder die Darstellung auf einem kleinen Bildschirm (z.B. eines Handys) können durch Frames erschwert werden, da für diese meistens eine feste Größe definiert wurde. Außerdem widersprechen Frameseiten den Regeln der Barrierefreiheit, da sie beispielsweise für blinde Menschen unter Umständen schwer zu navigieren sind.[5]

Weiterhin fehlt in einigen HTML-Dokumenten die Dokumenttypdefinition, diese entsprechen also nicht den Spezifikationen des World-Wide-Web-Konsortiums. Eine Validierung einer zufällig ausgewählten Unterseite mit Hilfe des Markup-Validation-Service des World-Wide-Web -Konsortiums zeigte neben der fehlenden Dokumenttypdefinition weitere 109 Validierungsfehler an. Diese große Zahl an Fehlern ist vermutlich auf die Erstellung der Seiten mit Hilfe eines sehr alten HTML-Editors zurückzuführen, was auch an den Meta-Tags einiger Seiten zu erkennen war.

Zusätzlich zu den bereits genannten Mängeln wurde eine große Sicherheitslücke entdeckt: der interne Bereich der Seite des Verbandes der Immobilienverwalter Sachsen e.V., welcher nur über die Eingabe eines Passwortes erreichbar ist, wurde lediglich durch eine kleine Javascript-Anwendung geschützt. Dies birgt ein hohes Sicherheitsrisiko, da Javascript immer vom Browser ausgeführt wird und dessen Quellcode damit auch bei Bedarf von jedem Besucher der Seite eingesehen werden kann. So können alle nicht autorisierten Benutzer, die Grundkenntnisse in Javascript besitzen, den Quellcode analysieren und sich so Zugang zum geschützten Bereich verschaffen.

2.2 Anforderungen (Soll)

Der neue Internetauftritt des Verbandes der Immobilienverwalter Mitteldeutschland e.V. soll Informations- und Kommunikationsplattform für Mitglieder, Partner und Förderer des Verbandes sein sowie der gezielten Suche nach Immobilienverwaltern, Dienstleistern und Handwerkern dienen. Somit soll auch eine Vermittlung bzw. Kontaktaufnahme der genannten Gruppen untereinander vereinfacht werden. In diesem Kapitel sollen daher die Ansprüche an Layout und Gestaltung, die geplanten Inhalte und Funktionen sowie technische Anforderungen dargestellt werden.

[5] Vgl. O.V.: Frame (HTML), 2008, Internet: http://de.wikipedia.org/wiki/Frame_(HTML)

2.2.1 Layout und grafische Gestaltung

Abbildung 2-3: Ausschnitt der Startseite des DDIV

Das Layout und die Gestaltung der neuen Internetpräsentation sind weitestgehend durch die Website des Dachverbandes Deutscher Immobilienverwalter (DDIV) vorgegeben, da sich das optische Erscheinungsbild des Regionalverbands am Dachverband orientieren soll. Abbildung 2-3 zeigt einen Screenshot dieser Website vom 29. März 2008. Der Layout-Entwurf wurde daher auch bereits im Vorfeld dieser Diplomarbeit fertiggestellt.

Das neue Layout (Abbildung 2-4) wurde aus diesem Grund ebenfalls dreispaltig mit Kopf- und Fußbereich angelegt. Für einige Unterseiten ist auch ein vierspaltiges Layout vorgesehen, da teilweise zusätzliche Inhaltsbereiche benötigt werden.

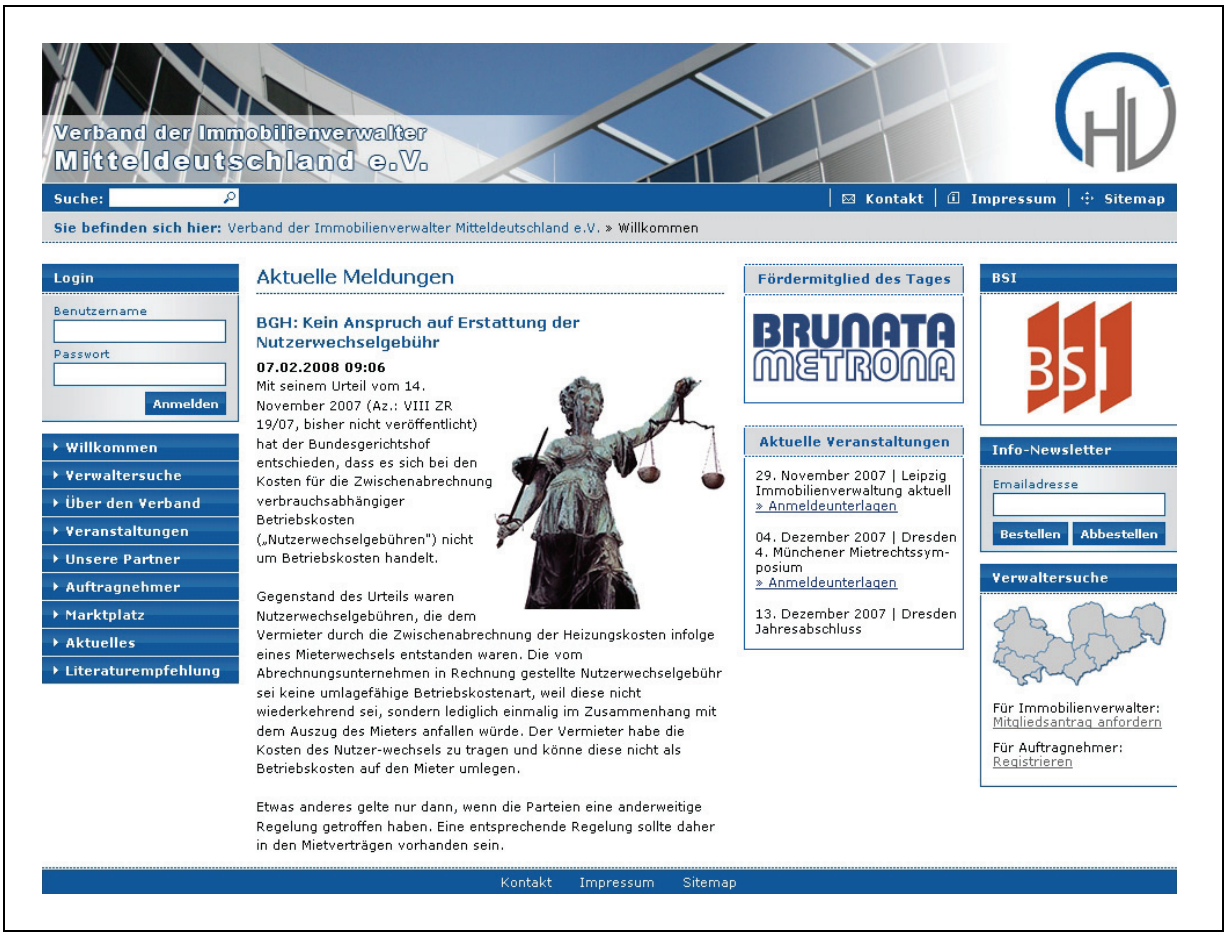

Abbildung 2-4: Layout-Entwurf für die Website des VdIVM e.V.

Im Kopfbereich der Seite befindet sich ein Themenfoto mit dem Namen des Verbandes als grafisches Element. Direkt darunter sind ein Textfeld zum Durchsuchen der Website sowie der Navigationspfad angeordnet. Außerdem befinden sich hier die Links zu den Service-Seiten Kontakt, Sitemap und Impressum.

Der Hauptbereich ist je nach Unterseite unterteilt in drei oder vier Spalten. Die Inhalte der linken und rechten Spalte sind auf allen Unterseiten gleich. In der linken Spalte befinden sich oben ein Login-

Formular und darunter die Hauptnavigation. In der rechten Spalte ist ganz oben das Logo der Bundesvereinigung Spitzenverbände der Immobilienwirtschaft (BSI) mit Hyperlink auf deren Website angeordnet. Darunter befindet sich ein Formularfeld zur Bestellung bzw. Abbestellung des Info-Newsletters. In der mittleren Spalte werden wechselnde Inhalte präsentiert. In diese kann bei Bedarf auch noch eine weitere Spalte eingefügt werden, um so das vierspaltige Layout zu realisieren und Platz für zusätzliche Inhalte zu schaffen.

Im Fußbereich befinden sich lediglich die gleichen Links zu den Service-Seiten, die auch im Kopfbereich vorhanden sind.

Um das optische Erscheinungsbild insgesamt etwas moderner wirken zu lassen, wurden bestimmten Elementen wie z.B. den Hauptnavigationspunkten bzw. den Überschriften in der linken und rechten Spalte noch dezente Glanzeffekte hinzugefügt. Andererseits wurden auch einige grafische Elemente der Website des Dachverbandes bewusst entfernt, wie z.B. die gepunkteten Pfeilspitzen vor den Hauptpunkten in der Navigation. Diese passten nicht mehr in das Gesamtbild des neuen Designs. Auch wurde auf eine farbliche Unterscheidung bestimmter Punkte wie „Verwaltersuche“ oder „Login“ verzichtet, um die Seite insgesamt geradliniger erscheinen zu lassen.

2.2.2 Inhalte und Funktionen

Startseite / Aktuelle Meldungen

Auf der Startseite des Internetportals sollen aktuelle Meldungen angezeigt werden. Zusätzlich sollen an jede Meldung optional Dateidownloads angefügt werden können.

Weiterhin wird hier im täglichen Wechsel ein neues Fördermitglied angezeigt, wobei die Auswahl allerdings mit gleicher Priorität erfolgen muss, so dass jedes Mitglied mit gleicher Häufigkeit angezeigt wird.

Auch ein Auszug aus kommenden Veranstaltungen soll auf der Startseite eingeblendet werden.

Newsletter

Auf jeder Seite soll die Bestellung oder Abbestellung des Info-Newsletters für alle Besucher der Website über ein entsprechendes Formular möglich sein. Außerdem können registrierte Mitglieder den Empfang des Newsletters in ihrer Profilverwaltung an- bzw. abschalten.

Verwaltersuche

Unter diesem Menüpunkt soll die Suche nach Immobilienverwaltern ermöglicht werden. Über ein Suchformular soll das Durchsuchen der registrierten Immobilienverwalter nach Name, Ort oder Post-

leitzahl ermöglicht werden. Die Suchergebnisse werden dann seitenweise angezeigt, wobei die Suchmaske stets oberhalb der Resultate stehen bleiben soll.

Oberhalb des Suchformulars sollen drei wechselnde Immobilienverwalter mit gleicher Priorität angezeigt werden, ähnlich wie beim aktuellen Fördermitglied auf der Startseite.

Über den Verband

Dieser Bereich soll zur Darstellung statischer Inhalte dienen, also überwiegend zur Anzeige von Text und Bildern. Es sollen beliebig viele Unterseiten mit ebenfalls statischen Inhalten möglich sein

Veranstaltungskalender

Neben allgemeinen Informationen sollen Veranstaltungen innerhalb verschiedener Unterkategorien angezeigt werden. Zusätzlich soll zu jeder Veranstaltung eine Detailbeschreibung mit Informationen zu Termin, Veranstaltungsort, Gebühr, Referent und Inhalt abrufbar sein. Außerdem soll eine Übersicht über alle Referenten bereitgestellt werden.

Partner / Fördermitglieder

Hier sollen die Namen und Logos aller Fördermitglieder des Verbandes mit Link zu deren Website angezeigt werden. Bei Bedarf soll für jedes Fördermitglied eine Unterseite mit beliebigem Inhalt abrufbar sein.

Auftragnehmer

In diesem Bereich sollen registrierte Dienstleister und Handwerker aufgelistet werden. Außerdem sollen eine Suche nach Name, Ort oder Postleitzahl und eine Filterung nach verschiedenen Gewerke-Kategorien möglich sein. Weiterhin sollen sich Dienstleister und Handwerker selbst registrieren können, die anschließende Freischaltung soll jedoch durch ein Verbandsmitglied erfolgen. Der Auftragnehmer muss dazu bei der Registrierung ein persönliches Referenzmitglied auswählen, welches anschließend eine E-Mail mit einem Hyperlink zur Freischaltung des neu registrierten Auftragnehmers erhält. Erst dann ist das neue Mitglied sichtbar und über die Suchfunktion auffindbar.

Marktplatz

Hier sollen Anzeigen für Stellenangebote, Stellengesuche oder berufliche Zusammenarbeiten abrufbar sein. Alle Anzeigen sollen für eine Dauer von maximal drei Monaten sichtbar bleiben.

Aktuelles

In diesem Bereich sollen aktuelle Dokumente, meist in Form von PDF-Dateien, zum Download bereitgestellt werden.

Literaturempfehlungen

Hier sollen Fachbücher mit dem entsprechenden Cover und Informationen über Autor, Verlag, Erscheinungsjahr usw. aufgelistet werden.

Interner Bereich

Hier können ähnlich wie im Bereich „Über den Verband“ beliebige Informationen hinterlegt werden, welche aber nur für registrierte Mitglieder sichtbar sind. Weiterhin können hier registrierte Immobilienverwalter Stellenanzeigen eintragen, welche dann im Marktplatz erscheinen. Diese werden allerdings moderiert und müssen erst von einem Administrator freigeschaltet werden, bevor sie sichtbar sind. Jede Stellenanzeige kann von dem Mitglied, von welchem sie erstellt wurde, nachträglich editiert oder gelöscht werden. Außerdem kann hier jedes Mitglied sein persönliches Profil verwalten.

Weitere Funktionen

Alle bisher genannten Inhalte und Funktionen sollen in einem geschützten Administrations-Bereich editiert und verwaltet werden können. Es sollen demzufolge zunächst einmal Mitglieder hinzugefügt, editiert, gelöscht oder freigeschaltet werden können. Ebenso sollen die Beiträge der aktuellen Meldungen sowie die Termine des Veranstaltungskalenders verwaltet werden können. Weiterhin sollen neue Einträge zu den Literaturempfehlungen, zu den Referenten und zu den Gewerken der Auftragnehmer hinzugefügt bzw. vorhandene Einträge editiert oder gelöscht werden können. Außerdem sollen auf bestimmten Seiten wie „Über den Verband“ oder im internen Bereich allgemeine Inhalte wie Texte, Bilder, Downloads usw. eingefügt werden können.

Aus den beschriebenen Inhalten und Funktionen ergibt sich eine bestimmte Seitenstruktur, die in Abbildung 2-5 ersichtlich ist:

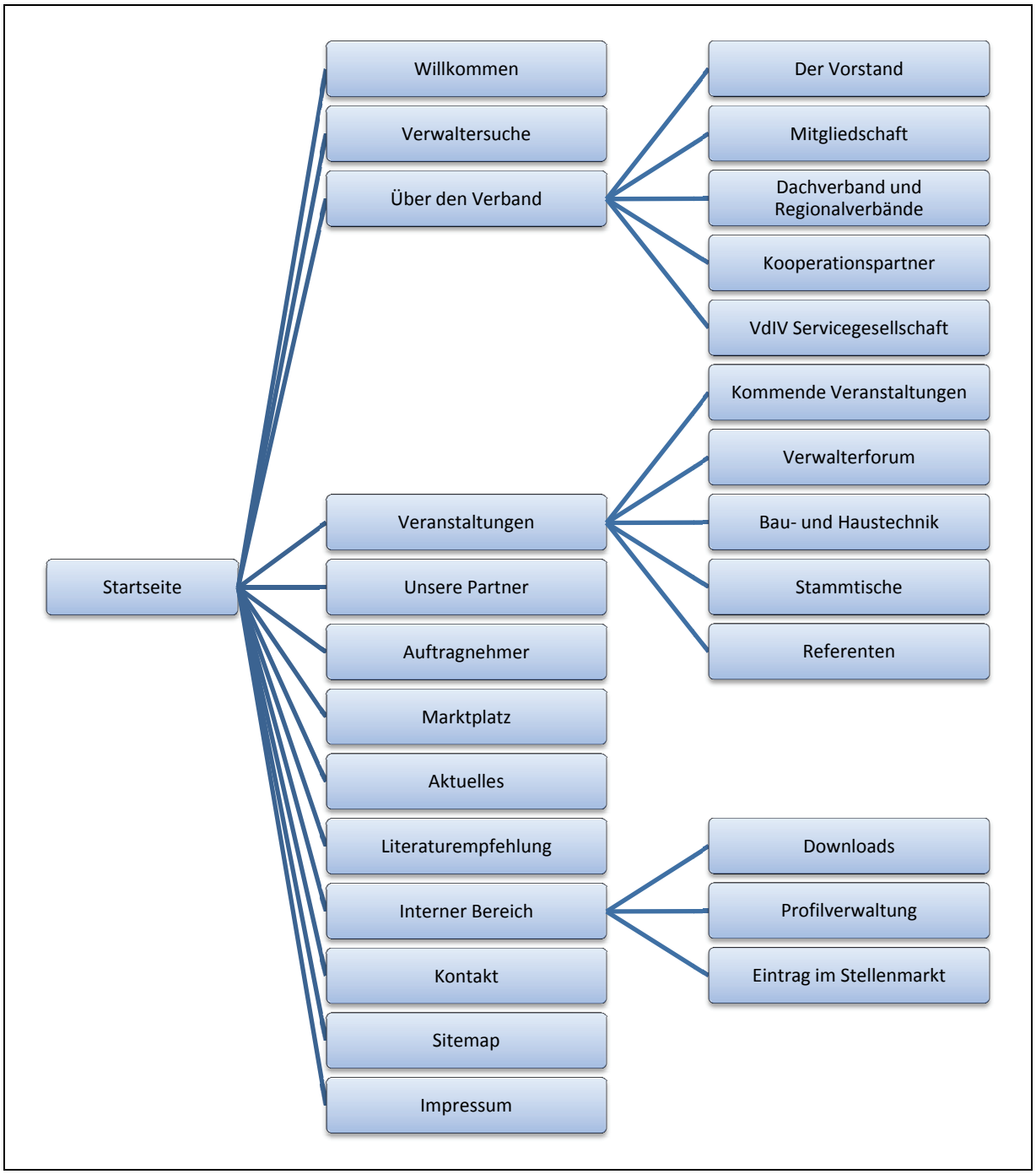

Abbildung 2-5: Seitenstruktur des geplanten Internetauftritts

2.2.3 Technische Anforderungen

Die gesamte Website soll mit Hilfe eines Content-Management-Systems zu verwalten sein. Dazu wird in diesem Fall TYPOlight verwendet. Dadurch sind zugleich die Anforderungen an Programmiersprache und Datenbanksystem erfüllt, da der zur Verfügung stehende Webspace nur PHP als Program-

miersprache unterstützt und MySQL als Datenbanksystem anbietet. TYPOlight ist daher gut geeignet, da es wie viele andere Content-Management-Systeme in PHP implementiert wurde und neben weiteren Datenbanksystemen auch MySQL unterstützt.

Außerdem soll das fertige System leicht zu warten und zu erweitern sein. Deshalb ist eine gute Dokumentation der geplanten Extensions erforderlich.

3 Vergleich von TYPOlight mit Typo3

In diesem Kapitel wird das Content-Management-System TYPOlight, welches für den geplanten Internetauftritt verwendet werden soll, mit Typo3, einem der verbreitetsten Content-Management-Systeme der Welt, verglichen. Dazu werden verschiedene qualitative und übergeordnete Faktoren, Kriterien der Architektur und Infrastruktur, der Inhaltserstellung sowie der Benutzerverwaltung und des Workflows untersucht und verglichen. Als Referenz dient dazu die Vergleichsmatrix der Website **cms-vergleich.de**[6], welche mit ausdrücklicher Erlaubnis des Betreibers für diese Diplomarbeit verwendet werden darf. Deren Vergleichspunkte werden nun in diesem Kapitel abgearbeitet und für beide Systeme kurz erläutert. Zum Schluss sollen ein Fazit gezogen und die Gründe für die Bevorzugung von TYPOlight vor Typo3 für die Umsetzung des geplanten Internetauftrittes erläutert werden.

3.1 Produkt

Typo3 wurde ursprünglich von Kasper Skårhøj entwickelt. Mittlerweile hat sich eine sehr große Entwicklergemeinschaft um mehrere Kernentwickler herausgebildet, deren zentrale Anlaufstelle die Website http://typo3.org ist.

TYPOlight wird seit 2004 von Leo Feyer entwickelt. Anlaufstelle für Updates und Extensions ist die Website http://typolight.org, welche ebenfalls von Leo Feyer betreut wird. Auch die Entwicklergemeinschaft um TYPOlight befindet sich in einem stetigen Wachstum und umfasst derzeit etwa 1800 Mitglieder[7].

3.2 Qualitative und übergeordnete Faktoren

Dieses Kapitel untersucht Lizenz, Support, Dokumentation, Erweiterbarkeit und Performance von TYPOlight und Typo3 und vergleicht die genannten Punkte miteinander.

3.2.1 Lizenz

Art der Lizenz

Typo3 steht unter der GPL-Lizenz, was zur Folge hat, dass für das System keine Lizenzgebühren verlangt werden, sämtlicher Quellcode offengelegt wird und es außerdem von jedem Benutzer verän-

[6] Vgl. O.V.: CMS-Vergleich.de – Content-Management-System, 2007,
Internet: http://www.cms-vergleich.de/cms/?s=cmsliste&cid=4,
[7] Stand: 19.05.2008

dert, vervielfältigt und weiterverbreitet werden darf. Bedingung für den letztgenannten Punkt ist allerdings, das sämtliche Programme, die ganz oder nur teilweise von Typo3 abgeleitet werden, ebenfalls unter der GPL-Lizenz veröffentlicht werden müssen, was wiederum die zwingende Offenlegung des Quelltextes zur Folge hat.

TYPOlight wurde unter der LGPL-Lizenz veröffentlicht, einer Lizenz mit weniger strengen Vorgaben als bei der GPL. Dies hat zur Folge, dass Extensions von TYPOlight, die keine Veränderungen am Systemkern beinhalten, nicht zwingend unter einer Open-Source-Lizenz veröffentlicht werden müssen.[8]

3.2.2 Support

Testmöglichkeiten

Für **TYPOlight** ist auf der Projekt-Website eine komplexe Demo-Website mit vollem Admin-Zugang verfügbar,[9] so dass jeder interessierte Benutzer alle standardmäßig vorhandenen Features ohne vorherigen Installationsaufwand sofort ausprobieren kann.

Auf der Developer-Resource-Website[10] von **Typo3** ist keine Online-Demo zu finden. Allerdings bieten zahlreiche Drittanbieter die Nutzung von Typo3-Demos auf ihren Websites an. Hierfür ist jedoch teilweise eine kostenlose Registrierung notwendig. Eine registrierungsfreie Demo-Website auf Basis von Typo3 ist auf der Website von OPEN SOURCE CMS zu finden,[11] so dass auch dieses System unmittelbar getestet werden kann.

Schulungs- und Supportangebote über Hersteller / Anbieter

Auf der Website von **TYPOlight** gibt es keine Möglichkeit für individuelle Supportanfragen, eine Bearbeitung von Problemen ist nur über das angeschlossene Forum möglich. Allerdings veranstaltet die Entwicklergemeinschaft von TYPOlight im April 2008 erstmals ein „TYPOlight-Usertreffen“. Dort werden Workshops, Diskussionen und Vorträge sowohl für Entwickler, die sich mit der Weiterentwicklung von TYPOlight befassen, als auch für Agenturen, die TYPOlight zur Umsetzung von Internetauftritten verwenden, angeboten.

Die **Typo3**-Website bietet die Möglichkeit, über ein Kontaktformular oder per E-Mail professionelle Unterstützung anzufordern. Die Anfragen werden derzeit[12] von Jens Kristian Mygent, zweiter Begu-

---

[8] Vgl. Quade, Jürgen / Kunst, Eva-Katharina: Linux-Treiber entwickeln. Eine systematische Einführung in Gerätetreiber für den Kernel 2.6, 2. Auflage, 2006, S. 50
[9] Zu finden unter http://demo.typolight.org
[10] Zu finden unter http://typo3.org
[11] Zu finden unter http://demo.opensourcecms.com/typo3/
[12] Stand: 01.04.2008

tachter des Typo3 Association Boards, direkt beantwortet oder an die betreffenden Instanzen weitergeleitet. Kompetente Unterstützung ist demzufolge gewährleistet.

Schulungs- und Supportangebote über Drittfirma

Einziger Anbieter für **TYPOlight**-Support ist derzeit die Firma iNEt-Robots aus München, bei welcher es sich allerdings um das eigene Unternehmen von Leo Feyer, dem Entwickler von TYPOlight handelt. Eine Unterstützung durch externe Drittfirmen ist momentan also noch nicht gegeben.

Im Gegensatz dazu sind auf der Website von **Typo3** Listen für mehrere Länder mit Support-Anbietern für dieses System abrufbar. Die Auflistung für den deutschsprachigen Raum umfasst derzeit[13] 21 Firmen, welche bestimmte Kriterien zur Aufnahme in den Datenbestand erfüllen müssen. Vielfältiger und kompetenter Support ist also gewährleistet.

Vor-Ort-Service

Weder für **TYPOlight** noch für **Typo3** wird derzeit ein expliziter Vor-Ort-Service angeboten.

Telefon-Hotline

Für **TYPOlight** wird gegenwärtig keine telefonische Unterstützung angeboten.

Für **Typo3** wird im Gegensatz dazu eine Hotline durch die Firma „Externes Management Arend" betrieben, welche montags bis samstags von 7:00 Uhr bis 22:00 Uhr erreichbar ist.

Internet-Communities für Benutzer / Entwickler

Sowohl hinter **TYPOlight** als auch hinter **Typo3** steht eine engagierte Entwicklergemeinschaft mit einem sehr aktiven Forum. Auf Grund des höheren Bekanntheitsgrades und der längeren Entwicklungshistorie von Typo3 ist dessen Community selbstverständlich weitaus größer als die von TYPOlight.

3.2.3 Dokumentation

Benutzerhandbuch

Auf der Website von **TYPOlight** wird ein 66 Seiten umfassendes Handbuch im PDF-Format als Download bereitgestellt, welches sowohl für Benutzer als auch Administratoren gedacht ist. Zusätzlich ist ein weiteres PDF-Handbuch mit einem Umfang von 99 Seiten verfügbar, welches speziell an Redakteure gerichtet ist. Gedruckte Handbücher sind für dieses System momentan noch nicht auf dem Markt.

[13] Stand: 01.04.2008

Auf Grund der weitreichenden Verbreitung von **Typo3** gibt es für dieses System auch dementsprechend umfangreiche Dokumentationen. Neben einem offiziellen **Typo3 Book** gibt es noch umfangreiche Literatur diverser Autoren und Verlage. Außerdem gibt es auf der Website von Typo3 zahlreiche elektronische Dokumente, u.a. Anleitungen für den Einstieg und den Umgang mit Typo3.

Administrationshandbuch

Das Administrationshandbuch von **TYPOlight** ist Teil des Benutzerhandbuchs, denn dieses enthält einige Kapitel wie Installation, Backend-Einstellungen oder Systemwartung, welche speziell an Administratoren gerichtet sind. Diese Kapitel sind jedoch relativ kurz gefasst und beschreiben nur die grundlegendsten Möglichkeiten für Administratoren. Weiterhin sind auf der TYPOlight-Website mehrere Tutorials verfügbar, u.a. zur Installation und zur Template-Erstellung.

Für **Typo3** sind ein Online-Installationshandbuch sowie zahlreiche weitere Dokumente abrufbar, die speziell an Administratoren gerichtet sind. Dazu gehören unter anderem Artikel aus Fachzeitschriften, die verschiedene Themen rund um Typo3 behandeln, sowie Anleitungen und Dokumentationen für zahlreiche Typo3-Extensions.

Entwicklerhandbuch

Für **TYPOlight** ist kein echtes Entwicklerhandbuch verfügbar. Es existieren bisher lediglich zwei kurze Online-Tutorials, welche die Grundlagen der Entwicklung eines Moduls kurz erläutern. Daher soll in einem späteren Kapitel dieser Diplomarbeit auch genauer auf die Entwicklung von TYPOlight-Modulen eingegangen werden.[14]

Auf der **Typo3**-Website existiert neben mehreren PDF-Dokumenten speziell für Entwickler sogar ein Handbuch mit Programmier-Richtlinien. Zusätzlich zu den offiziellen Dokumenten der Typo3-Entwicklergemeinschaft sind noch zahlreiche Fachbücher von Drittanbietern verfügbar. Als Beispiel sei hier nur „Das Typo3 Profihandbuch“ von Melanie Meyer, Irene Höppner und Franz Ripfel genannt.

Technische Referenz

Für **TYPOlight** sind Referenzen für alle verfügbaren Hooks (Callback-Funktionen für bestimmte Ereignisse) sowie für das globale Konfigurations-Array[15] abrufbar.

Für **Typo3** existieren neben Referenzen für **TypoScript** sowie für die globale Konfiguration (TSconfig) noch einige weitere Dokumente auf der Entwickler-Website.

[14] Siehe auch Kapitel 6 - Tutorial: Programmierung einer Extension für TYPOlight
[15] Siehe auch Kapitel 5.1.1 - Model

API-Referenz

Sowohl für **TYPOlight** als auch für **Typo3** ist eine vollständig offengelegte API-Referenz auf der jeweiligen Entwickler-Website verfügbar.[16]

Kontextbezogene Online-Hilfe

Im **TYPOlight**-Backend kann bei Bedarf unter jedem Eingabefeld ein kurzer Hilfetext eingeblendet werden. Außerdem sind an verschiedenen Stellen detailliertere Hilfe-Informationen abrufbar.

Typo3 bietet ebenfalls in jedem Bereich des Backends die Möglichkeit, eine Hilfe-Seite mit kontextbezogenen Informationen zu öffnen.

Dokumentation von Fehlermeldungen

In **TYPOlight** neu auftretende Fehler können in einem entsprechenden Bereich im TYPOlight-Supportforum gemeldet und dokumentiert werden,[17] ansonsten existiert keine Referenz über Fehlermeldungen.

Auf der **Typo3**-Website existiert ein spezieller Bereich, der nur zur Dokumentation von gefundenen Fehlern und deren Bearbeitungsstatus dient.[18]

3.2.4 Erweiterbarkeit

Erlaubnis zur Manipulation des Quellcodes

Da **TYPOlight** unter der LGPL-Lizenz und **Typo3** unter der GPL-Lizenz veröffentlicht wurden,[19] besteht für beide Systeme die ausdrückliche Erlaubnis, Modifikationen am Quellcode vorzunehmen.

Offengelegte API

Wie bereits in Kapitel 3.2.3 beschrieben, ist sowohl für **TYPOlight** als auch für **Typo3** eine vollständige API-Referenz abrufbar. Außerdem können die API-SChnittstellen beider Systeme selbstverständlich für die Entwicklung von Extensions genutzt werden.

Erweiterung über Module/Extensions möglich

TYPOlight ist komplett modular aufgebaut und bietet somit die Möglichkeit, auf einfache Art und Weise neue Module ins Gesamtsystem zu integrieren. Eine genauere Beschreibung dazu folgt im Kapitel 6. Außerdem bringt das System schon eine Reihe nützlicher Module wie Event-Kalender,

[16] Für TYPOlight: http://www.typolight.org/api, für Typo3: http://typo3.org/documentation/api/
[17] Zu finden unter http://www.typolight.org/forum/viewforum.php?id=5
[18] Zu finden unter http://typo3.org/development/bug-fixing
[19] Siehe auch Kapitel 3.2.1 - Lizenz

Newssystem oder Newsletter-Verwaltung mit sich. Jedes Modul kann bei Bedarf deaktiviert werden und, sobald es benötigt wird, wieder aktiviert werden.

Typo3 bringt in der Standardinstallation nur die wichtigsten Kernfunktionen mit sich und bietet ebenfalls die Möglichkeit, das System über Extensions zu erweitern. Diese Möglichkeit wurde bisher von der großen Entwicklergemeinschaft auch ausgiebig genutzt, da bis jetzt ca. 3000[20] Extensions für Typo3 existieren.

3.2.5 Performance

Load Balancing

Sowohl für **Typo3** als auch für **TYPOlight** kann Load-Balancing nur unter Zuhilfenahme einer speziell dafür ausgelegten Software wie **Pound** oder **balance** realisiert werden. Hierbei wird das entsprechende System jeweils auf mehreren speziellen Active-Active-Clustern installiert, welche alle Zugriff auf die gleiche Datenbank und ein gemeinsam genutztes Dateisystem haben. Anfragen an eine bestimmte Webseite werden nun über die Load-Balancing-Software an einen bestimmten Cluster weitergereicht, welcher diese dann bearbeitet und die Antwort ausliefert.[21] Die Systeme an sich unterstützen also kein Load Balancing.

Caching-Mechanismus

TYPOlight bietet zum einen Caching-Mechanismen für HTML-Seiten an. Diese werden, sobald sie einmal generiert wurden, in einer Caching-Tabelle in der Datenbank abgelegt und bei einer erneuten Anforderung direkt aus der Caching-Tabelle geladen und angezeigt. Hierbei kann für jede Seite individuell festgelegt werden, ob die Seite gecacht werden soll und wie lang die Cache-Verfallszeit sein soll, also nach welcher Zeit die Seite nicht mehr aus dem Cache geladen, sondern neu generiert wird. Zum anderen werden skalierte Grafiken in einem speziellen Verzeichnis zwischengespeichert und bei Bedarf von dort neu geladen.

Typo3 bringt weitaus umfangreichere Caching-Mechanismen mit sich als TYPOlight. „Neben der HTML-Ausgabe statischer Seiten werden verschiedene Konfigurationseinstellungen gecacht, außerdem Bilder-Abmessungen, Objektbäume teils in Tabellen, teils als Dateien. Auch verschiedene Extensions bringen Caching-Mechanismen mit. Bilder und andere Dateien, die auf den Server geladen werden, zählen dagegen nicht zum Cache, wohl aber die dazu erzeugten Thumbnails.“ [22]

---

[20] Stand: 14.04.2008

[21] Vgl. Scholl, Martin: Lastenausgleich mit Typo3. Wie das CMS lastverteilt betrieben wird, in: T3N, Ausgabe 7 (03/2007 – 05/2007) vom 07.03.2007

[22] Vgl. Laborenz, Kai / Ertel, Andrea / Wendt, Thomas u.a.: TYPO3 4.0: Das Handbuch für Entwickler, 2. Auflage, 2006, S. 607

Staging- und dynamische Methoden kombinierbar

Der Begriff Staging beinhaltet in diesem Fall, dass die Inhalte auf einem eigenen System bzw. Server erstellt und verwaltet werden. Zu einem definierten Zeitpunkt wird dann ein Generat aus statischen HTML-Seiten erzeugt (Staging) und auf einen Webserver exportiert. Wenn dieses Staging nun mit dynamischen Methoden kombiniert werden soll, müssen bestimmte Bereiche der Website von dem Generat statischer HTML-Seiten ausgeschlossen werden können und stets dynamisch erzeugt werden.

TYPOlight bietet keine Möglichkeiten für Staging. Die Produktionsumgebung läuft stets auf dem gleichen Server wie der Webserver, welcher für die Auslieferung der Inhalte zuständig ist. Auch können keine statischen HTML-Seiten generiert werden.

Typo3 bietet die Möglichkeit dynamische und statische Publikationen durch die Kombination der mod_rewrite Option und dem sogenannten Static File Publishing zu realisieren. Dazu muss eine entsprechende .htaccess-Datei angelegt und die Funktion „simulateStaticDocuments“ aktiviert werden. Außerdem muss über den Paramater „publish_dir“ ein Verzeichnis für das Generieren der statischen HTML-Seiten festgelegt werden. [23]

3.3 Architektur und Infrastruktur

Dieses Kapitel erläutert die Anforderungen beider Systeme an Client- und Server-Rechner, beschreibt deren Migrations- und Portierungsfähigkeiten und untersucht die Möglichkeiten zur Datensicherung und -archivierung.

3.3.1 Server

Datenbank als Content Repository / Dateisystem

Sowohl **TYPOlight** als auch **Typo3** verwenden eine Datenbank zur Speicherung von Inhalten. Zusätzlich wird jedoch auch von beiden Systemen das Dateisystem des Servers genutzt, um bestimmte Dateien wie z.B. Grafiken oder PDF-Dokumente zu speichern und zu verwalten.

Anbindung externer Datenbanken

TYPOlight gestattet lediglich die Nutzung einer Datenbank. Diese wird global für das gesamte System bei der Installation festgelegt und kann später durch erneutes Ausführen des Installationsskripts ge-

[23] Vgl. Roth, Jörg: Statische Dokumente generieren, 2003,
Internet: http://www.typo3.net/forum/list/list_post//3058/

wechselt werden. Diese Datenbank kann sich allerdings auch auf einem anderen Server als das System selbst befinden.

In der Standardinstallation unterstützt **Typo3** auch lediglich die Nutzung einer Datenbank. Allerdings wird durch die Installation bestimmter Extensions wie z.B. DBAL (Database Abstraction Layer) auch die Nutzung weiterer, externer Datenbanken ermöglicht.

3.3.2 Clients

Backend komplett Browser-basiert

TYPOlight und **Typo3** benötigen beide auf Client-Seite lediglich einen Internetbrowser zur Nutzung des vollen Funktionsumfanges. Typo3 setzt hier im Backend allerdings noch aktiviertes Javascript voraus, während das Backend von TYPOlight dagegen auch ohne Javascript voll funktionsfähig ist.

3.3.3 Migrations- und Portierungsfähigkeiten

Konzepte für die Integration bestehender HTML-Seiten

TYPOlight bietet keine Möglichkeit bestehende HTML-Seiten in die systemeigene Seitenverwaltung zu importieren.

Für **Typo3** dagegen existiert z.B. eine Extension namens Fetch URL, welche es gestattet, beliebige HTML-Inhalte von einer URL zu importieren und diese als Inhaltselement in eine Seite einzubetten. Für eine große Anzahl an statischen Seiten ist hierbei jedoch auch ein hoher Arbeitsaufwand nötig, da jede Seite einzeln importiert werden muss und die bestehende Seitenstruktur nicht automatisch in Typo3 abgebildet wird. Davon ist also abzuraten.

Importmöglichkeiten aus bestehenden Datenbanken

Direkt über das **TYPOlight**-System können keine Daten aus bestehenden Datenbanken importiert werden, hierfür ist der Umweg über ein fremdes System wie z.B. PhpMyAdmin vonnöten.

In der Standardinstallation von **Typo3** ist dies ebenfalls nicht möglich, kann aber ggf. über spezielle Extensions realisiert werden.

Importmöglichkeiten aus strukturierten Rohdaten

TYPOlight selbst bietet keine Möglichkeit, strukturierte Rohdaten zu importieren, hier ist wieder der Umweg über eine externe Anwendung nötig.

Für **Typo3** steht z.B. eine Extension zur Verfügung, die es ermöglicht, Daten aus Microsoft Office Word- und Excel-Dokumenten sowie aus Open-Office-Dokumenten im Format sxw und sxc zu importieren oder einfach nur als HTML-Seite darzustellen.[24]

3.3.4 Datensicherung und -archivierung

Backupstrategien

TYPOlight besitzt in der Standardinstallation keine Funktion zur Sicherung von Datenbank, Konfiguration oder Dateien. Es existiert jedoch eine Extension namens „Backup Database“, welche die Sicherung der Datenbank einer kompletten Website als Website-Template ermöglicht. Dieses kann später wieder über das Installationsskript importiert werden. Das Dateisystem wird dabei allerdings nicht berücksichtigt, dieses muss manuell gesichert werden.

Für **Typo3** ist die Extension „Full Backup“ verfügbar, welche die Sicherung einer kompletten Typo3-Installation inklusive Datenbank, Dateisystem (fileadmin), Uploads und Konfigurationsdateien ermöglicht.

Transparentes Backup

Sowohl unter **TYPOlight** als auch **Typo3** kann während des Backup-Vorganges weitergearbeitet werden. Bei beiden Systemen wird ein SQL-Dump der Datenbank erzeugt. Bei Typo3 wird im Gegensatz zu TYPOlight zusätzlich noch das Dateisystem (fileadmin) gesichert.

Möglichkeit zur Datenhaltung auf Spiegelserver

Sowohl **Typo3** als auch **TYPOlight** unterstützen derzeit keine Datenhaltung auf Spiegelservern.

Archivierungsmethoden

In **TYPOlight** kann eine Archivierung von Seiteninhalten lediglich indirekt durch die Nutzung der im System enthaltenen Versionierung von Datensätzen realisiert werden.

In **Typo3** kann dies besser realisiert werden, indem ein separater Workspace als Archiv genutzt wird.

Recherchemöglichkeiten im Archiv

Die einzelnen Versionen können in **TYPOlight** nicht durchsucht werden, da diese serialisiert in der Datenbank abgespeichert werden.

In **Typo3** dagegen kann der Archiv-Workspace regulär durchsucht werden.

---

[24] Vgl. Hinderink, Daniel: Typo3 spricht MS-Office, 2003,
Internet: http://www.contentmanager.de/magazin/news_h5607_typo3_spricht_ms-office.html

Sicherstellung der Datenkonsistenz bei Systemabbruch

Da es sich bei **TYPOlight** und **Typo3** um webbasierte Content-Management-Systeme handelt, liegt die Aufgabe der Sicherstellung der Datenkonsistenz bei dem Server, auf welchem die Systeme laufen, und nicht bei dem System selbst. Somit bieten beide Systeme auch keine entsprechenden Funktionalitäten an.

3.4 Inhaltserstellung

3.4.1 Templates

Template-Erstellung durch Skripting

Für **Typo3** können komplexe Templates mit Hilfe der Typo3-internen Konfigurationssprache TypoScript erstellt werden, welche definiert, wie mit Template-Markern verfahren werden soll.

Für **TYPOlight** existiert keine eigene Skriptsprache, allerdings verwenden alle Templates für dieses System PHP als Skriptsprache, so dass hier auch eine gewisse Steuerungslogik innerhalb der Templates möglich ist.

Template-Erstellung grafisch über Browser (WYSIWYG)

Weder **TYPOlight** noch **Typo3** bieten die Möglichkeit an, Templates direkt grafisch im Browser zu erstellen.

Template-Erstellung durch HTML-Editoren

TYPOlight und **Typo3** erlauben die Erstellung von Templates über gewöhnliche HTML-Editoren, da für alle Templates HTML als Grundlage verwendet wird. Allerdings ist in beiden Fällen auch die Nachbearbeitung des Templates notwendig, da bestimmte PHP-Anweisungen von TYPOlight bzw. Marker von Typo3 benötigt werden, damit das System korrekt mit den Templates arbeiten kann.

Frei definierbare und positionierbare Elemente im Template

In **TYPOlight** können neben den standardmäßigen Layout-Bereichen beliebige Bereiche über Div-Container definiert werden. Diese müssen allerdings noch in den globalen Einstellungen des Systems angegeben werden, damit sie im Backend verfügbar sind und mit Inhalten oder Modulen gefüllt werden können. Außerdem können über Methoden der Template-Klasse beliebige Module direkt im Template geladen werden.[25]

[25] Siehe auch Kapitel 6.8 - Erstellen von Templates

In **Typo3** können Elemente im Template durch Marker definiert werden und mit Hilfe von TypoScript für einen bestimmten Verwendungszweck eingesetzt werden.

Verwendung von Style Sheets möglich

TYPOlight erlaubt es zum einen, Style Sheets direkt im Backend zu erstellen und zu verwalten, welche dann für jedes Template individuell zugewiesen werden können. Zum anderen können auch bereits vorhandene CSS-Dateien importiert werden und anschließend im Backend weiterbearbeitet werden.

Typo3 erlaubt die Einbindung vorhandener Style Sheets in ein Template mit Hilfe von TypoScript.

Möglichkeit zur Erstellung von Formularen

TYPOlight beinhaltet in der Standardinstallation das Backend-Modul „Formular Generator“, welches die Erstellung von HTML-Formularen direkt im Backend erlaubt. Verfügbar sind dabei alle standardmäßigen HTML-Formular-Elemente. Diese können jeweils zusätzlich noch mit Validierungs- und Filterfunktionen versehen werden, um die eingegeben Daten auf Korrektheit zu prüfen.

Typo3 bietet die Möglichkeit, Mailformulare als Inhaltselemente zu erzeugen. Allerdings gibt es hier in der Standardinstallation nicht die Möglichkeit, die eingegebenen Werte in der Datenbank zu speichern, was bei TYPOlight jedoch möglich ist. Auch sind hier ohne entsprechende Extension keine Validierungs- und Filterfunktionen möglich.

3.4.2 Redaktion

WYSIWYG-Editor

TYPOlight und **Typo3** sind beide standardmäßig mit einem WYSIWYG-Editor ausgestattet, der bei der Eingabe von textuellen Inhalten verwendet werden kann. Bei TYPOlight kommt TinyMCE zum Einsatz, während in Typo3 htmlArea Verwendung findet. TinyMCE ist hier jedoch im Vorteil, da er im Gegensatz zu htmlArea die Ausgabe von XHTML unterstützt und außerdem eine höhere Browserkompatibilität bietet.

Vorschaufunktion

TYPOlight und **Typo3** bieten beide die Möglichkeit, angelegte Seiteninhalte in einer Vorschau zu betrachten, ohne dass diese bereits veröffentlicht werden müssen.

Direktes Publizieren aus Office-Dokumenten

TYPOlight kann derzeit noch keine Office-Dokumente verarbeiten.

Für **Typo3** gibt es eine Extension, die das direkte Importieren und Anzeigen von Microsoft Office und OpenOffice Dokumenten ermöglicht,[26] wenn diese in einem vorher definierten Ort hinterlegt werden.

Integration eigener Skripte

In **TYPOlight** können lediglich eigene Javascript-Dateien im Headerbereich für jede Seite oder jedes Seitenlayout individuell eingebunden werden. **Typo3** dagegen gestattet das Einfügen von PHP oder Javascript direkt als Inhaltselement.

Publizieren in andere Zielformate als HTML

TYPOlight bietet die Möglichkeit, jede Seite als PDF-Dokument generieren zu lassen.

In **Typo3** gibt es zusätzlich dazu noch die Möglichkeit, WML-, XML- oder RDF-Dokumente zu erzeugen.

Automatische Generierung von Print-Versionen der Seite

In **TYPOlight** ist die Generierung der Printversion einer Seite nicht direkt möglich, sondern nur über den Umweg der Generierung eines PDF-Dokuments.

Für **Typo3** dagegen gibt es zahlreiche Erweiterungen, die die Erzeugung von Print-Versionen einer Seite ermöglichen, wie z.B. die Extension „Printlink Page“.

Dateien per Upload-Button anhängbar

In **TYPOlight** können Dateien nicht direkt per Upload-Button als Inhaltselement angehängt werden, hier ist stets der Umweg über den systemeigenen Dateimanager notwendig, der zum Hochladen von Dateien verwendet werden muss.

Typo3 dagegen bietet beim Einfügen eines Download-Elements direkt einen Upload-Button an, mit dem neue Dateien unmittelbar hochgeladen werden und dem Element angefügt werden können.

Browserunabhängigkeit und -kompatibilität des erzeugten Codes

TYPOlight erzeugt XHTML-konformen HTML-Code nach W3C-Standards. Dabei kann zwischen den Dokumenttypdefinitionen „XHMTL Strict“ und „XHTML Transitional“ gewählt werden.

Bei **Typo3** bestehen keine Einschränkungen bzgl. Des erzeugten Codes, da der Entwickler die volle Freiheit in der HTML-Programmierung besitzt. Außerdem ist seit der Version 3.6 ebenfalls die XHTML-konforme Ausgabe in Typo3 möglich. Hierfür sind einige wenige TypoScript-Einstellungen im Setup der Seite notwendig.

[26] Siehe auch Kapitel 3.3.3 - Migrations- und Portierungsfähigkeiten

Mehrsprachigkeit der Oberfläche und Hilfe

TYPOlight bietet in der Standardinstallation die Auswahlmöglichkeit zwischen deutscher und englischer Sprache für das Backend. Weitere Sprachpakete sind auf der Website der Entwicklergemeinschaft von TYPOlight verfügbar.

Für **Typo3** existieren bereits über 40 Sprachpakete für das komplette Backend. In der Standardinstallation ist allerdings nur die englische Sprache verfügbar, weitere Sprachpakete müssen nachträglich hinzugefügt werden.

3.4.3 Textinhalte

Eingabe von Inhalten ohne HTML-Kenntnisse

Sowohl **TYPOlight** als auch **Typo3** bieten die Möglichkeit, auch ohne HTML-Kenntnisse z.B. HTML-Überschriften, Downloads, Tabellen oder komplexere Elemente wie Bildergalerien und Formulare einzufügen. Außerdem können HTML-Inhalte durch den integrierten WYSIWYG-Editor eingefügt werden.

Texteditierung im Rich Text Modus

Wie bereits in Kapitel 3.4.2 erwähnt, besitzen **TYPOlight** und **Typo3** beide einen integrierten WYSIWYG-Editor, welcher die Bearbeitung von Text im Rich Text Modus gestattet. Hierbei sind jeweils die Formatierung von Textgröße, Textfarbe, Textauszeichnung und Textausrichtung sowie das Einfügen spezieller Elemente wie Bilder, Tabellen u.v.a. möglich.

Einschränkungen bezüglich Rich Text möglich

Sowohl in **TYPOlight** als auch in **Typo3** kann zunächst festgelegt werden, welche HTML-Elemente generell bei der Eingabe erlaubt sind. Weiterhin kann durch Bearbeitung der Konfigurationsdateien des Rich Text Editors festgelegt werden, welche Buttons und Menüpunkte in diesem verfügbar sind. Diese Einstellungen wirken sich allerdings global auf alle Benutzer aus, eine Unterscheidung nach Benutzergruppen ist hierbei nicht möglich.

Rechtschreibprüfung

Im WYSIWYG-Editor von **TYPOlight** ist die Möglichkeit zur Rechtschreibprüfung standardmäßig aktiviert.

In **Typo3** ist dieses Feature auch möglich, jedoch muss das dafür benötigte Plugin erst aktiviert werden.

Limitierung der Zeichenanzahl

Weder in **TYPOlight** noch in **Typo3** ist die Limitierung der Zeichenanzahl bei der Eingabe von Text möglich.

Mit Anzeige der verbleibenden Zeichen

Da keine Limitierung der verfügbaren Zeichen möglich ist, entfällt in **TYPOlight** und **Typo3** die Anzeige der noch verfügbaren Zeichen.

Vorgabe von Default-Werten für Eingabefelder

Für Eingabefelder in **TYPOlight** können Default-Werte im globalen Konfigurations-Array hinterlegt werden, indem die entsprechenden Konfigurationsdateien[27] bearbeitet werden. Diese Werte sind dann allerdings global für alle Benutzer gültig.

In **Typo3** können die Default-Werte sogar abhängig vom jeweiligen Benutzer unterschiedlich festgelegt werden.

3.4.4 Multimediale Inhalte

Proportionale Änderung der Auflösung von Bildern

In **TYPOlight** und in **Typo3** kann für jedes Bild, welches als Inhaltselement eingefügt wird, die Bildbreite und/oder die Bildhöhe in Pixeln angegeben werden, wobei das Bild dann proportional skaliert wird.

Zuschneiden von Bildern

Weder **TYPOlight** noch **Typo3** bieten die Möglichkeit, Bilder direkt im Backend zuzuschneiden. Hierfür muss jeweils ein externes Bildbearbeitungsprogramm verwendet werden.

Formatkonvertierung von Bildern

Sowohl in **TYPOlight** als auch in **Typo3** werden alle Grafiken im Ursprungsformat angezeigt, sofern dieses unterstützt wird. Auch nach einer Skalierung erfolgt keine Konvertierung in ein anderes Format.

Direkte Einbindung anderer Medienformate

TYPOlight unterstützt derzeit nur die direkte Einbindung von Flash-Filmen.

[27] Siehe auch Kapitel 5.1.1 - Model

Typo3 hingegen unterstützt neben dem Flash-Format auch noch diverse Audio- und Videoformate wie z.B. WAV, AVI, MOV, MPG, WMV oder MP3, welche als Inhaltselement eingebunden werden können.

Asset Management

TYPOlight bietet zurzeit keine Möglichkeit für Asset Management, also zur Verwaltung von Mediendateien verschiedener Formate. Auch eine entsprechende Extension existiert noch nicht.

Für **Typo3** ist die Extension „Media (DAM)“ verfügbar, welche die Verwaltung und Speicherung von Medien wie Bildern, Text-, Audio- oder Videodateien ermöglicht. Durch die Eingabe von Metadaten ist hier auch das Organisieren und Durchsuchen des Datenbestandes möglich.

3.4.5 Metadaten

Zeitliche Steuerung

Sowohl in **TYPOlight** als auch in **Typo3** besteht die Möglichkeit, für jede Seite eine Start- und ein Enddatum festzulegen, zwischen welchen die Seite öffentlich zugänglich ist. In TYPOlight ist dies außerdem für jeden Artikel möglich, so dass verschiedene Bereiche auf einer Seite getrennt voneinander zeitlich gesteuert werden können[28].

Manuelle Eingabe von Schlagwörtern

In **TYPOlight** können für jeden Artikel manuell Schlagwörter (Meta-Keywords) eingegeben werden. Die Eingabe einer Seitenbeschreibung (Meta-Description) ist jeweils getrennt für jede Seite möglich.

Typo3 erlaubt unter den erweiterten Optionen beim Bearbeiten einer Seite ebenfalls die Vergabe von Schlagwörtern und einer Seitenbeschreibung getrennt für jede Seite.

Definition und Weiterverarbeitung beliebiger Metatags

In **TYPOlight** ist die direkte Verarbeitung beliebiger Metatags nicht möglich. Es können zwar in einem Seitenlayout beliebige weitere Metatags definiert werden, dies würde jedoch einen nicht vertretbaren Arbeits- und Pflegeaufwand bedeuten, da für jede Seite ein individuelles Seitenlayout benötigt würde.

In **Typo3** ist die Vergabe beliebiger Metatags über die Nutzung von TypoScript möglich.

[28] Siehe auch Kapitel 4.5 - Artikel

3.4.6 Versionskontrolle

Versionskontrolle

TYPOlight speichert für jedes Inhaltselement die Vorgängerversionen zusammen mit dem Benutzernamen des jeweiligen Autors in serialisierter Form in einer speziellen Tabelle in der Datenbank ab. Jede Version kann später bei Bedarf wiederhergestellt werden.

Typo3 bietet seit Version 3.8 komplexe Möglichkeiten zur Versionskontrolle. So gestattet das System nun, verschiedene Versionen von Seiten und Inhalten zu erzeugen und zu verwalten. Es werden dabei drei Versionsmodi unterschieden: der Modus „Branch“ versioniert ganze Seitenbäume, „Page“ ganze Seiten und die darauf abgelegten Datensätze, „Element“ dagegen versioniert nur einzelne Datensätze.[29]

Mehrsprachigkeit für Inhalte

TYPOlight bietet zwar die Möglichkeit, für jede Seite die verwendete Sprache festzulegen, allerdings ist eine echte Mehrsprachigkeit im Frontend nicht direkt möglich. Dies kann nur realisiert werden, indem die komplette Website dupliziert wird, für das Duplikat eine andere Sprache festgelegt wird und anschließend alle Inhalte übersetzt werden.

Typo3 bietet schon in der Standardinstallation vielfältige Möglichkeiten zur Unterstützung von Mehrsprachigkeit. So können beispielsweise für jede Website mehrere Sprachen definiert werden und anschließend jede beliebige Seite in den verschiedenen Sprachversionen hinterlegt werden.[30]

3.4.7 Link Management

Zyklische Überprüfung von Links

Weder **TYPOlight** noch **Typo3** können Links auf ihre Gültigkeit überprüfen. Dies muss in beiden Systemen manuell von einem Administrator oder Redakteur erledigt werden.

Automatische Anpassung aller internen Links bei Dateiänderungen

Sofern in **TYPOlight** beim Einfügen von Links die speziell dafür vorgesehenen Insert-Tags verwendet werden, werden auch alle internen Links beim Umbenennen oder Verschieben von Seiten automatisch angepasst.

---

[29] Vgl. Niederlag, Peter: Typo3 4.0: Workspaces. Optimierte redaktionelle Workflows, in: T3N, Ausgabe 2 (12/2005 – 02/2006) vom 05.12.2005

[30] Vgl. O.V.: Typo3 und die Mehrsprachigkeit, 2006,
Internet: http://www.c-dev.ch/kaepten/typo3/tutorial_1_sprachen.html

In **Typo3** wird für die Identifikation einer Seite zwingend die Seiten-ID benötigt, so dass Links immer auf diese ID verweisen müssen. Seiten können also umbenannt werden, ohne dass die Links davon beeinflusst werden. Die Nutzung der Seiten-ID als Link wirkt sich allerdings ungünstig auf die Suchmaschinenoptimierung der Website aus.

Verschieben von Dateien per Drag & Drop

In **TYPOlight** können Dateien nicht per Drag & Drop verschoben werden, hier muss jede Datei, die verschoben werden soll, separat aktiviert werden, um sie für das Verschieben vorzubereiten. Anschließend muss das Verzeichnis ausgewählt werden, in welches die Datei verschoben werden soll.

In **Typo3** können bedingt durch Javascript-Unterstützung Seiten und Dateien auch per Drag & Drop verschoben werden. Allerdings muss bei diesem Vorgang in einem kleinen Kontextmenü anschließend noch ausgewählt werden, ob die Seite bzw. Datei in das Zielelement (z.B. bei Verzeichnissen) oder nach dem Zielelement eingefügt werden soll.

Automatische Generierung und Aktualisierung einer Sitemap

In **TYPOlight** kann eine Sitemap als Modul angelegt werden und an beliebiger Stelle eingefügt werden.

In **Typo3** ist das Einfügen einer Sitemap als Inhaltselement auf jeder Seite möglich. In beiden Systemen wird die Sitemap stets automatisch anhand der aktuellen Seitenstruktur aktualisiert.

3.4.8 Suchfunktionen

Suchmöglichkeiten für Autoren und Besucher der Site

TYPOlight bringt für das Frontend ein spezielles Suchmodul mit sich, welches bei Bedarf an jeder beliebigen Stelle der Website eingebunden werden kann. Die Suche basiert dabei auf einem Suchindex in einer Datenbank-Tabelle, welcher für jede Seite abhängig vom Template und von den Seiteneinstellungen generiert wird. Im Backend können nur bestimmte Datenbanktabellen wie z.B. die Benutzertabelle gezielt anhand von Suchbegriffen durchsucht werden, für Seiten und Artikel ist dies jedoch nicht möglich.

Unter **Typo3** ist für Besucher der Website eine datenbankbasierte Suche, welche gezielt steuerbar ist, oder eine indexbasierte Suche über alle Inhalte möglich. Zur Realisierung der Suche besteht wie in TYPOlight die Möglichkeit, ein Suchformular an einer beliebigen Stelle der Website als Inhaltselement einzufügen. Im Backend können Autoren ebenfalls über die standardmäßig vorhandene Backend-Suche alle Inhalte durchsuchen.

Suche über Metainformationen

In **TYPOlight** können Inhalte über Metainformationen durchsucht werden, sofern diese mit in den Suchindex aufgenommen wurden.

In **Typo3** werden die Metainformationen bei der Suche standardmäßig mit berücksichtigt.

Suche innerhalb von Dateien

TYPOlight besitzt keine Funktionen zum Durchsuchen von Dateien.

In **Typo3** können beispielsweise Word- oder PDF-Dokumente durchsucht werden, sofern speziell dafür benötigte Programme auf dem Server installiert sind.

Globale Funktion für „Suchen / Ersetzen“

Weder **TYPOlight** noch **Typo3** besitzt die Möglichkeit zum globalen Suchen und Ersetzen von Inhalten bzw. Textstellen.

3.5 Benutzerverwaltung und Workflow

3.5.1 Kapazität

limit concurrent users

Weder **TYPOlight** noch **Typo3** schränken die Anzahl der gleichzeitigen Backend-Benutzer ein. Eine Obergrenze wird hier nur durch die Leistungsfähigkeit des Servers, auf welchem das jeweilige System installiert ist, gesetzt. Auch gibt es bei beiden Systemen keine Einstellungsmöglichkeit, um die Anzahl konkurrierender Benutzer zu beschränken.

Schließt die Benutzerverwaltung auch Endbenutzer mit ein

TYPOlight ermöglicht bereits in der Standard-Installation zusätzlich zur Verwaltung von Backend-Benutzern auch die Verwaltung von Frontend-Mitgliedern.

Bei **Typo3** muss hierfür erst eine entsprechende Extension installiert werden. Die Verwaltung von Backend-Benutzern ist hier allerdings auch in der Standardinstallation im vollen Umfang möglich.

Mandantenfähigkeit

TYPOlight und **Typo3** besitzen beide Mandantenfähigkeit, bieten also beide die Möglichkeit, eine unbegrenzte Anzahl voneinander unabhängiger Websites mit nur einer TYPOlight- bzw. Typo3-Installation zu verwalten.

3.5.2 Benutzerverwaltung

Möglichkeit zur Definition von Benutzergruppen

In **TYPOlight** und **Typo3** können Benutzer jeweils einer oder mehreren Gruppen zugeordnet werden, von welcher der entsprechende Benutzer auch bestimmte Rechte erben kann. In TYPOlight existiert für die Verwaltung im Backend ein eigenes Modul „Benutzergruppen“, in Typo3 ist die Verwaltung der Gruppen zusammen mit der Benutzerverwaltung im Bereich „User Admin“ möglich.

Vererbung von Benutzerrechten

Sowohl in **TYPOlight** als auch in **Typo3** erbt ein Benutzer seine Rechte von den Benutzergruppen, denen er angehört.

Vererbung von Administratorrechten

In **TYPOlight** kann jeder Benutzer als Administrator bestimmt werden, es erfolgt allerdings keine Vererbung bestimmter Rechte.

In **Typo3** ist ebenfalls keine Vererbung von Benutzerrechten möglich, da neue Benutzer nur von Administratoren angelegt werden können und dabei auch die Vergabe individueller Benutzerrechte erfolgt.

Einschränkung von Benutzerrechten

In **TYPOlight** und in **Typo3** hat ein Administrator die Möglichkeit, für jeden Benutzer individuelle Berechtigungen mit Hilfe der Benutzerverwaltung festzulegen und diese einzuschränken.

3.5.3 Workflow

Mehrstufige Freigabekontrolle

In **TYPOlight** existieren keine mehrstufigen Methoden zur Freigabekontrolle. Eine einfache Freigabekontrolle kann nur realisiert werden, in dem beispielsweise das Feld zur Veröffentlichung eines Artikels für normale Redakteure gesperrt wird und die Veröffentlichung nur nach Kontrolle eines Chefredakteurs erfolgt, welcher die entsprechenden Rechte im Backend besitzt.

In **Typo3** kann eine mehrstufige Freigabekontrolle durch die Bearbeitung von Inhalten in Workspaces realisiert werden, indem einem normalen Redakteur beispielsweise nur die Bearbeitung im Draft-Workspace gestattet wird, während die Übernahme in den Live-Workspace durch den Chefredakteur erfolgen muss.

Möglichkeit zur freien Festlegung der Freigabewege

Weder in **TYPOlight** noch in **Typo3** existieren Möglichkeiten zur Festlegung von Freigabewegen.

Locking-Mechanismus

Datensätze, die sich gerade in Bearbeitung befinden, werden weder in **TYPOlight** noch in **Typo3** gelockt, d.h. es kann in beiden Systemen u.U. zu Komplikationen kommen, wenn zwei verschiedene Benutzer zur gleichen Zeit den gleichen Datensatz bearbeiten wollen.

Erinnerungsfunktionen

TYPOlight und **Typo3** besitzen beide ein Taskcenter, welches die Verteilung von Aufgaben an Backend-Benutzer ermöglicht. Diese werden beim Login an die anstehenden Aufgaben erinnert und können optional per E-Mail benachrichtigt werden.

3.5.4 Inhaltsbezogene Rechte

Zugriffsbeschränkung nach Dateien

Sowohl **TYPOlight** als auch **Typo3** bieten die Möglichkeit, für jeden Benutzer ein bestimmtes Verzeichnis als Mountpoint zu bestimmen, dessen Dateien und Unterverzeichnisse dieser Benutzer dann bearbeiten darf.

Zugriffsbeschränkung nach Datensätzen

Eine Zugriffsbeschränkung für jeden einzelnen Datensatz ist weder in **TYPOlight** noch in **Typo3** möglich, allerdings können in beiden Systemen die Zugriffsberechtigungen für jeden Benutzer auf frei wählbare Seiten und deren Unterseiten beschränkt werden.

Zugriffsbeschränkung nach Datenfelder

In **TYPOlight** und in **Typo3** kann für jede Benutzergruppe oder sogar speziell für jeden einzelnen Benutzer genau festgelegt werden, welche Datenfelder er editieren darf. Alle gesperrten Datenfelder sind für den jeweiligen Benutzer im Backend nicht sichtbar.

Differenzierung der Zugriffsbeschränkungen

In **TYPOlight** ist eine Zugriffsbeschränkung nach Erstellen, Bearbeiten und Löschen nur für das Dateisystem möglich. Außerdem kann für bestimmte Module wie „Nachrichten“ oder „Kalender“ die Bearbeitung für einen bestimmten Benutzer auf ausgewählte Nachrichten- bzw. Kalenderarchive eingeschränkt werden.

In **Typo3** können für jeden Benutzer explizit das Hochladen, Kopieren, Verschieben, Löschen, Umbenennen, Erstellen und Bearbeiten von Dateien oder das Entpacken von Dateiarchiven freigegeben

werden. Für Verzeichnisse können das Verschieben, Löschen, Umbenennen und Erstellen, weiterhin das Kopieren sowie außerdem das rekursive Löschen für jeden Benutzer erlaubt oder verweigert werden.

3.6 Benchmark-Test

Um die Leistung von TYPOlight mit der von Typo3 vergleichen zu können, soll ein Benchmark-Test unter möglichst realitätsgetreuen Bedingungen durchgeführt werden. Dieses Kapitel beinhaltet die Beschreibung des Tests und dessen Bedingungen sowie die anschließende Auswertung der Testergebnisse.

3.6.1 Testbeschreibung

Der Benchmark-Test dient zur Ermittlung der Performance von TYPOlight im Vergleich zu Typo3 bei der Auslieferung von textuellen Inhalten. Er wird für jedes System separat und unabhängig voneinander durchgeführt. Dazu werden an jedes System Anfragen gesendet, welche die Auslieferung von Webseiten mit Inhalten unterschiedlicher Textlänge zur Folge haben. Dabei sollen die Seiten im ersten Testlauf nur aus dem Cache-Speicher ausgeliefert werden, im zweiten Testlauf aber auch bei jeder Anfrage neu generiert werden, um auch die Geschwindigkeit beim Seitenaufbau des jeweiligen Systems zu messen. Zusätzlich wird als Vergleichsreferenz zu jeder Seite des jeweiligen Systems eine statische HTML-Seite direkt vom Webserver angefordert, welche den gleichen Inhalt wie die aktuell zu vergleichende Seite aufweist.

Zunächst sollen an beide Systeme eine bestimmte Anzahl **aufeinanderfolgender** Anfragen gesendet und die mittlere Antwortzeit auf diese gemessen werden. Diese Vorgehensweise dient zur Bestimmung der allgemeinen Performance, also um zu ermitteln, welches System die angefragten Daten am schnellsten ausliefert. Hierbei sollte nach jeder Anfrage eine kurze Pause bis zur nächsten Anfrage eingelegt werden, da nur so sichergestellt ist, dass beide Anfragen auch wirklich nacheinander abgearbeitet werden, und nicht etwa durch zwei konkurrierende Kindprozesse des Webservers.

Anschließend soll die Messung außerdem für eine bestimmte Anzahl **konkurrierender** Anfragen durchgeführt werden, um die Reaktionszeiten der Systeme unter Last zu ermitteln. Hierfür muss jedoch zunächst durch Ausprobieren unter gleichzeitiger Beobachtung der Speicherauslastung des Servers die richtige Anzahl konkurrierender Anfragen gefunden werden. Dieser Schritt ist notwendig, da bei einer zu hohen Speicherbelastung Daten im Hauptspeicher zum Teil auf die Festplatte ausgelagert werden. Der Zugriff auf diese ausgelagerten Daten ist wesentlich langsamer als auf die Daten

im Hauptspeicher, was sich wiederum negativ auf die Antwortzeiten auswirken würde. Die Anzahl konkurrierender Anfragen sollte also so gewählt werden, dass die Speicherauslastung höchstens bei ca. 70% liegt.

Zur Messung der Antwortzeiten wird das Programm ApacheBenchmark verwendet. Hierbei handelt es sich um eine kleine Kommandozeilenapplikation, welche das Senden aufeinanderfolgender oder konkurrierender Anfragen an beliebige Webserver ermöglicht und außerdem eine detaillierte Statistik über die Antwortzeit der jeweiligen Anfrage erstellen kann.

3.6.2 Testbedingungen

Um alle Faktoren auszuschalten, welche das Testergebnis beeinflussen könnten, werden optimale Testbedingungen benötigt.

Dazu ist zunächst die gleiche Hardware-Konfiguration des Servers für alle Tests nötig, um Abweichungen in der Antwortzeit durch unterschiedliche Prozessorgeschwindigkeiten oder Speicherauslastungen ausschließen zu können. Weiterhin müssen alle Tests auf dem gleichen Betriebssystem durchgeführt werden, da die Speicher- und Prozessverwaltung zwischen verschiedenen Betriebssystemen erheblich voneinander abweicht. Zusätzlich muss der Testrechner nach jedem Testlauf neu gestartet werden, um den Hauptspeicher zurückzusetzen. Außerdem muss für alle Tests die gleiche Netzwerkkonfiguration verwendet werden, da sonst die Antwortzeiten durch kürzere oder längere Transportzeiten der Datenpakete beeinflusst werden.

Da die Tests auf einem Apache-Webserver durchgeführt werden, müssen auch bei dessen Konfiguration bestimmte Punkte beachtet werden, um einer Verfälschung des Testergebnisses vorzubeugen.

Zunächst müssen die Parameter „MaxKeepAliveRequests“ und „KeepAliveTimeout“ des Apache-Webservers so eingestellt werden, dass alle Anfragen eines Testlaufs über die gleiche TCP-Verbindung abgewickelt werden. Dies verhindert Zeiteinbußen durch einen möglichen Neuaufbau der Verbindung. Der Parameter „MaxKeepAliveRequests“ legt fest, nach wie vielen Anfragen desselben Clients ein neuer Verbindungsaufbau erzwungen wird. Dieser Wert sollte also höher als die Gesamtzahl aller Anfragen eines Testlaufs angesetzt werden. Über den Parameter „KeepAliveTimeout“ wird bestimmt, nach wie vielen Sekunden eine Verbindung ohne Anfrage getrennt wird. Dieser Wert sollte auf 0 gesetzt werden, da so eine Verbindung dauerhaft gehalten wird, auch wenn keine Anfragen gesendet werden. [31]

[31] Vgl. Apache HTTP Server Version 2.0: Apache-Kernfunktionen, 2008,
Internet: http://httpd.apache.org/docs/2.0/mod/core.html, aufgerufen am 12.04.2008

Außerdem sollte unter einem Windows-System der Parameter „ThreadsPerChild“ stets so eingestellt werden, dass bei einer Messung der Antwortzeit von konkurrierenden Anfragen diese auch tatsächlich konkurrierend abgearbeitet werden können.[32] Bei 100 konkurrierenden Anfragen sollte dieser Wert also ebenfalls mindestens auf den Wert 100 gesetzt werden. Sollen dagegen aufeinanderfolgende Anfragen gemessen werden, so sollte dieser Wert auf 1 gesetzt werden, damit die Anfragen auch wirklich nacheinander vom Webserver abgearbeitet werden, und nicht etwa durch zwei konkurrierende Threads.

3.6.3 Testdurchführung

Zunächst wird ein lokales Netzwerk zwischen den zwei Rechnern, die für den Test verwendet werden sollen, eingerichtet. Eine genaue Konfiguration beider Rechner ist im Anhang 0 nachzulesen. Auf dem Rechner, welcher als Server dient, wird der Apache-Webserver zusammen mit PHP und MySQL installiert. Weiterhin werden hier sowohl TYPOlight als auch Typo3 in ihrer aktuellsten Version unter Verwendung einer jeweils neu angelegten MySQL-Datenbank installiert.

Anschließend werden für beide Systeme zwei identische HTML-Templates erstellt, welche nichts außer der Definition des Dokumenttyps, einem Header-Bereich mit einer Meta-Angabe zum verwendeten Zeichensatz sowie einen leeren Body-Bereich enthalten. In den Body-Bereich wird jeweils ein systemspezifischer Platzhalter zum dynamischen Einfügen von Inhalten eingesetzt.

Als nächstes werden in beiden Systemen jeweils zehn Seiten angelegt, die das zuvor erstellte Template zur Darstellung von Inhalten verwenden. Das Caching der Seiten wird zunächst deaktiviert, so dass diese bei jedem Aufruf neu generiert werden müssen. Auf jeder Seite wird ein Inhaltselement vom Typ Text eingefügt, welches in beiden Systemen jeweils die Zeichenkette „Benchmark“ (mit anschließendem Leerzeichen) enthält. Diese Zeichenkette wird von Seite zu Seite mehrfach wiederholt, so dass zum Schluss in jedem System Seiten mit einer Textlänge von 10, 50, 100, 500, 1000, 1500, 5000, 10000, 25000, 50000, 75000 und 100000 Zeichen vorhanden sind.

Dann wird der eigentliche Test gestartet. Mit Hilfe des Programms ApacheBenchmark wird zunächst jede Seite von jedem System jeweils 500 Mal **nacheinander** abgerufen und so die mittlere Antwortzeit für jeden Seitenabruf bestimmt. Außerdem wird entsprechend zu jedem Seitenpaar aus den beiden Systemen eine statische HTML-Seite mit dem gleichen Textinhalt 500 Mal direkt vom Webserver abgefragt und so ebenfalls die mittlere Antwortzeit darauf bestimmt. Vor jedem Testlauf wird allerdings jede Seite zunächst ein einziges Mal vom Server abgefragt, um bereits im Vorfeld eine TCP-

[32] Vgl. Apache HTTP Server Version 2.0: Allgemeine Direktiven der Apache-MPMs, 2008, Internet: http://httpd.apache.org/docs/2.0/mod/mpm_common.html, aufgerufen am 12.04.2008

Verbindung aufzubauen, welche dann auch für den eigentlichen Testlauf genutzt werden kann. Diese Vorgehensweise verhindert eine Beeinflussung des Testergebnisses durch Zeitverluste beim Verbindungsaufbau. Anschließend werden alle Anfragen wiederholt, diesmal allerdings mit jeweils 100 **konkurrierenden** Anfragen in fünf Durchgängen, also insgesamt auch 500 Anfragen. Hierfür wird allerdings der Parameter „ThreadsPerChild“ des Apache-Webservers zunächst vom Wert 1 auf den Wert 100 gesetzt, um das Abarbeiten konkurrierender Anfragen zu ermöglichen.[33]

Zuletzt wird der gesamte Testablauf noch einmal wiederholt, allerdings wird hierfür vorher das Caching aller Seiten in beiden Systemen aktiviert, um auch die mittleren Antwortzeiten bei der Auslieferung von Seiten aus dem Systemcache bestimmen zu können. So werden insgesamt 100 Messungen durchgeführt, welche im nächsten Kapitel ausgewertet werden. Eine genaue Aufstellung aller Testergebnisse ist im Anhang 0 zu finden.

3.6.4 Testergebnisse

Abbildung 3-1 zeigt den Verlauf der mittleren Antwortzeiten in Abhängigkeit von der Zeichenanzahl des auszuliefernden Textes für TYPOlight und Typo3, wobei die jeweilige Seite einmal aus dem System-Cache ausgeliefert (durchgehende Linie) und einmal bei jeder Anfrage neu generiert wurde (gepunktete Linie). Außerdem werden als Vergleich die Antwortzeiten bei der Auslieferung der statischen HTML-Seite durch den Apache-Webserver grafisch dargestellt.

[33] Siehe auch Kapitel 3.6.2 - Testbedingungen

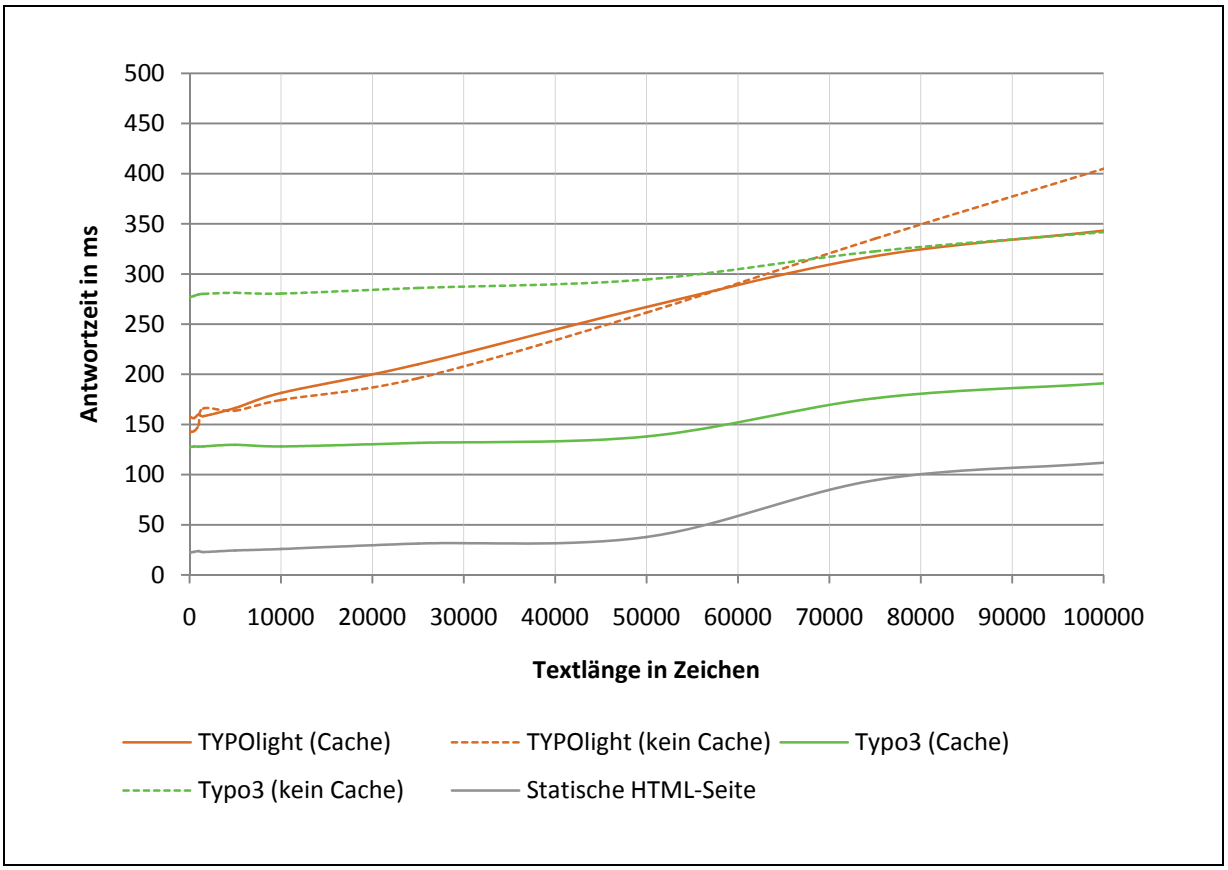

Abbildung 3-1: Mittlere Antwortzeiten bei aufeinanderfolgenden Anfragen

Auffallend ist hier, dass die Antwortzeiten beim Abruf einer Typo3-Seite bei allen Messungen bis zu einer Textlänge von ca. 50000 Zeichen nahezu konstant sind. Erst dann ist ein langsamer Anstieg der Antwortzeiten zu verzeichnen, wobei diese sowohl mit Cache als auch ohne Cache im gleichen Maße ansteigen wie beim Abruf der statischen HTML-Seite. Der Anstieg flacht jedoch ab einer Textlänge von ca. 75000 Zeichen wieder etwas ab. Insgesamt könnte man den Anstieg daher als nahezu linear bezeichnen. Diese Linearität wird jedoch abhängig von bestimmten Parametern innerhalb des Dateisystems, der Datenbankkonfiguration oder auch im Programmcode von Typo3 selbst regelmäßig unterbrochen. Eine genaue Bestimmung dieses Parameters ist jedoch im Rahmen dieses Benchmark-Tests nicht möglich.

Beim Abruf einer TYPOlight-Seite steigen die Antwortzeiten dagegen von Anfang an linear an. Bei diesem System lässt sich weiterhin feststellen, dass die Antwortzeiten beim Seitenabruf aus dem Cache fast identisch ausfallen wie bei einer Neugenerierung der Seite. Begründen lässt sich dies wahrscheinlich mit der geringen Komplexität der Testseiten, es war jeweils nur ein Inhaltselement eingebunden. So müssen bei einer Neugenerierung einer Seite im Wesentlichen zwei Datenbankab-

fragen ausgeführt werden: zum einen die Auswahl des zur Seite gehörenden Artikels,[34] zum anderen die Selektion aller zum Artikel gehörenden Inhaltselemente, wobei es sich in diesem Fall nur um ein einziges Element vom Typ Text handelt. Beim Seitenabruf aus dem Cache wird die komplette Seite mit nur einer Datenbankabfrage geladen. So lässt sich vermuten, dass die Antwortzeiten bei der Neugenerierung einer komplexeren Seite mit mehreren eingebundenen Modulen und Inhaltselementen deutlich höher ausgefallen wären als beim Abruf aus dem Cache, da dann wesentlich mehr Datenbankabfragen nötig gewesen wären.

Bei den Antwortzeiten von Typo3 ist im Gegensatz dazu ein deutlicher Unterschied zwischen der Neugenerierung einer Seite und dem Abruf aus dem Cache festzustellen. Auffallend ist hier auch, dass die Neugenerierung einer Seite bis zu einer Textlänge von ca. 75000 Zeichen wesentlich mehr Zeit in Anspruch nimmt als bei TYPOlight. Zurückzuführen ist dies vermutlich auf die weitaus größere Komplexität des Systems von Typo3. Dies wird allerdings wieder durch die sehr guten Caching-Mechanismen des Systems ausgeglichen, da die Antwortzeiten hier wieder kürzer ausfallen als bei TYPOlight. Auch steigen diese hier mit zunehmender Textlänge nicht so stark an wie bei TYPOlight. Ursache dafür könnte die in TYPOlight integrierte Funktion zur Ersetzung von sogenannten Insert-Tags sein, die auch bei Seiten aus dem Cache greift. So muss jede Seite erst komplett geparst, dabei auf Insert-Tags geprüft und diese ggf. ersetzt werden, bevor eine Auslieferung erfolgen kann. Bei längeren Seiten nimmt dieser Vorgang dann natürlich auch mehr Zeit in Anspruch als bei kürzeren. Dies ist vermutlich auch die Ursache dafür, dass TYPOlight bei der Neugenierung von Seiten ab ca. 75000 Zeichen wieder langsamer wird als Typo3.

Dieser Sachverhalt stellt ein generelles Problem bei TYPOlight im Vergleich zu Typo3 dar: TYPOlight lässt sich nur im vergleichsweise geringen Maße skalieren, während bei Typo3 jede benötigte Funktion separat ins Template eingebunden werden kann. Bei TYPOlight ist dies nicht möglich, die Funktion zur Ersetzung von Insert-Tags beispielsweise kann nicht abgeschaltet werden.

[34] Siehe auch Kapitel 4.5 - Artikel

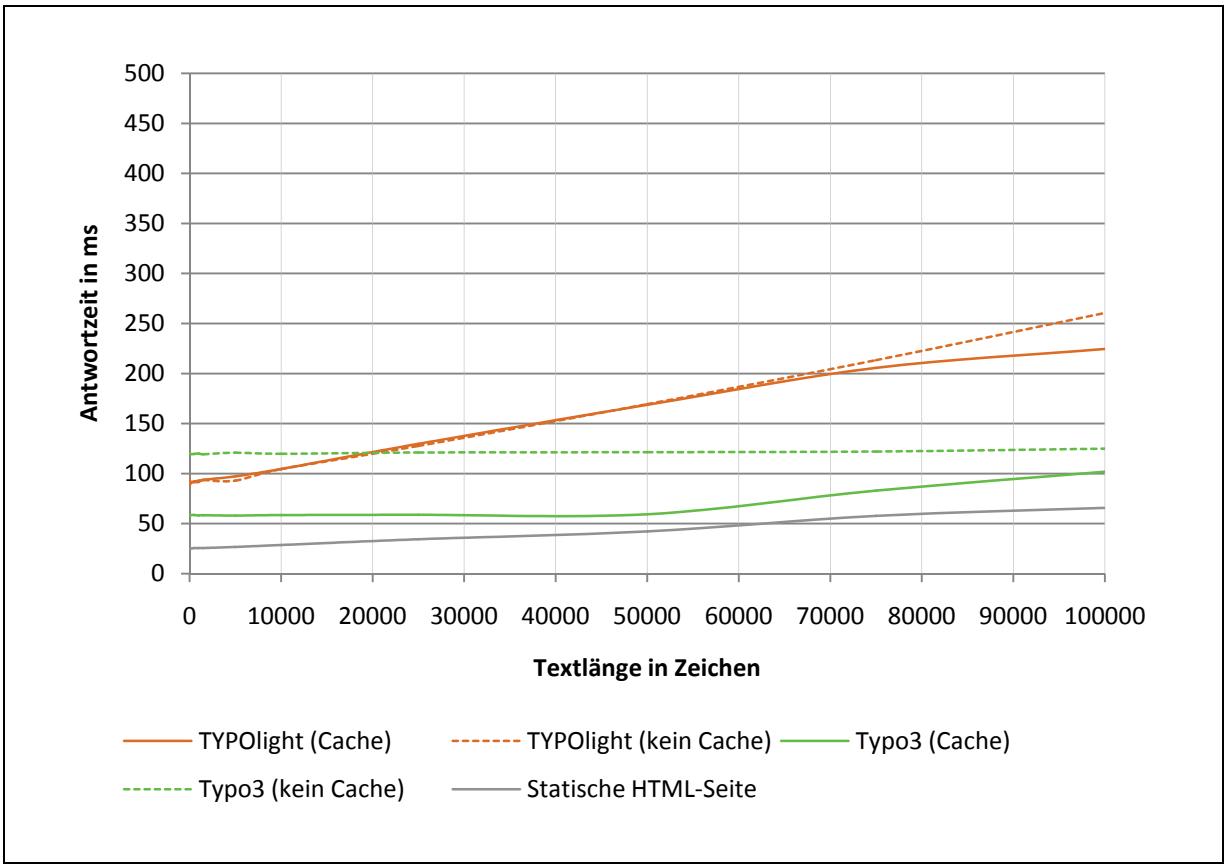

Abbildung 3-2: Mittlere Antwortzeiten bei konkurrierenden Anfragen

Abbildung 3-2 stellt ebenfalls die Antwortzeiten bei Anfragen an die beiden Systeme bzw. bei der Auslieferung einer statischen HTML-Seite dar, diesmal allerdings bei der Messung von jeweils 100 konkurrierenden Anfragen in je fünf aufeinanderfolgenden Durchgängen.

Der Verlauf der Antwortzeiten in Abhängigkeit von der Textlänge entspricht hier in etwa dem Verlauf bei aufeinanderfolgenden Anfragen, nur mit dem Unterschied, dass alle Anfragen zwischen ca. 75 und 150 Millisekunden schneller beantwortet wurden und die Antwortzeiten insgesamt näher beieinander liegen. Begründen lässt sich dies mit der hohen Rechenkapazität des Testservers, der bei aufeinanderfolgenden Anfragen noch längst nicht seine Belastungsgrenzen erreicht. Bei konkurrierenden Anfragen kann der Server bedingt durch seine hohe Rechenleistung mehrere Anfragen parallel in verschiedenen Threads bearbeiten, so dass der Mittelwert der Antwortzeiten über alle Anfragen insgesamt deutlich niedriger ausfällt.

Abbildung 3-2 verdeutlicht die Erkenntnisse, die anhand von Abbildung 3-1 bereits gewonnen werden konnten, noch stärker: Bei TYPOlight besteht bis zu einer Textlänge von 75000 Zeichen kein nennenswerter Unterschied mehr zwischen den Antwortzeiten bei neugenerierten Seiten und bei Abrufen aus dem Cache, bedingt durch die geringe Komplexität der Testseiten. Weiterhin ist bei Typo3 bei

der Neugenerierung von Seiten kein wesentlicher Unterschied mehr zwischen den Antwortzeiten abhängig von der Textlänge feststellbar, alle Seiten wurden mit nahezu identischer Geschwindigkeit generiert und ausgeliefert. Nur bei der Auslieferung von Seiten aus dem Cache ist wieder ab einer Textlänge von ca. 50000 Zeichen ein geringer Anstieg zu verzeichnen. Auch werden hier wieder die hervorragenden Caching-Mechanismen von Typo3 deutlich, da die Antwortzeiten hier wieder deutlich geringer ausfallen als bei einer Neugenerierung der Seite.

Zusammenfassend lässt sich sagen, dass Typo3 durch seine besseren Caching-Mechanismen Vorteile gegenüber TYPOlight besitzt. Dies wirkt sich besonders bei der Auslieferung von statischen Inhalten wie z.B. Artikeltexten positiv aus. Bei einer Neugenerierung von Seiten wies TYPOlight bis zu einer Textlänge von ca. 75000 Zeichen wiederum deutlich kürzere Antwortzeiten auf, so dass dieses System für die Auslieferung von dynamischen Inhalten, die sich bei jedem Seitenabruf ändern können, besser geeignet wäre als Typo3.

Die Auswahl des Systems sollte also stets vom individuellen Einsatzzweck abhängig gemacht werden. Hier spielen dann auch noch viele weitere Faktoren eine wichtige Rolle. Eine komplexe Seitenstruktur beispielsweise bringt auch eine komplexe Navigation mit sich. Hier können nun wieder deutliche Geschwindigkeitsunterschiede entstehen, abhängig davon, wie die Navigation systemintern gerendert wird. Dieses Prinzip lässt sich auf alle benötigten Funktionen bzw. Module übertragen: Je komplexer eine Seite aufgebaut ist, umso mehr hängt die Geschwindigkeit des Systems von der internen Optimierung jeder einzelnen Komponente ab. Dieser Benchmark-Test lässt also keine Verallgemeinerung zu, welches System generell schneller ist. Um eine perfekte Auswahl zu ermöglichen, müsste jedes System in verschiedenen Konfigurationen und mit unterschiedlich komplexen Seiten getestet werden. Dies würde allerdings den Rahmen dieser Diplomarbeit übersteigen.

3.7 Fazit

Als Fazit des Vergleichs von TYPOlight mit Typo3 kann festgestellt werden, dass Typo3 einen wesentlich höheren Funktionsumfang besitzt als TYPOlight. Zurückzuführen ist dies auf die länger währende Entwicklungshistorie und die sehr große Entwicklergemeinschaft. Dieses System ist daher für sehr große und komplexe Websites mit allen erdenklichen Funktionen besser geeignet als TYPOlight.

In einigen Punkten ist die Leistungsfähigkeit von TYPOlight jedoch auch auf gleichem Niveau wie die von Typo3. Dazu gehören z.B. die Benutzerverwaltung oder die Eingabe von textuellen Inhalten. Trotzdem ist TYPOlight eher für kleine bis mittelgroße Websites geeignet, da die weniger leistungsfähigen Caching-Mechanismen bei sehr komplexen Websites u.U. zu langen Auslieferungszeiten führen könnten.

Hauptgrund für die Auswahl von TYPOlight für die Umsetzung des Internetauftritts des Verbandes der Immobilienverwalter Mitteldeutschland e.V. sind jedoch vor allem die bereits in der Standardinstallation enthaltenen Module, die die Anforderungen an das Projekt bereits weitestgehend abdecken. Dazu gehören u.a. ein Newssystem, ein Modul zum Versenden von Newslettern, ein Kalender-/Event-Modul sowie die Verwaltung von Frontend-Mitgliedern. Weitere Gründe sind die geringere Komplexität des Systems, so dass Extensions einfacher implementiert werden können, sowie das benutzerfreundlichere Backend, welches einfacher und intuitiver zu bedienen ist als das von Typo3.

4 Umsetzung eines Internetauftritts mit TYPOlight

Nachdem die Auswahl von TYPOlight für das geplante Projekt begründet wurde, behandelt dieses Kapitel nun die prinzipielle Vorgehensweise bei der Umsetzung eines Internetauftrittes mit diesem System am Beispiel der Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. Dazu werden zunächst die Installation des Systems sowie das Anlegen von Modulen und deren anschließende Einbindung in ein Seitenlayout beschrieben. Danach wird auf den Aufbau einer Seitenstruktur eingegangen und zuletzt die Einbindung von Artikeln mit Inhaltselementen in diese erläutert.

4.1 Installation

Die Installation von TYPOlight gestaltet sich in der Regel sehr unkompliziert. Zunächst muss eine aktuelle Version des Systems von der TYPOlight-Website[35] heruntergeladen werden. Das so erhaltene ZIP- oder TAR-Archiv sollte direkt auf den Webserver übertragen und dort entpackt werden, sofern dieser das unterstützt. Ansonsten muss das Archiv zuerst lokal entpackt werden und anschließend alle entpackten Dateien per FTP auf den Webserver übertragen werden.

Ist der erste Schritt erledigt, so muss als nächstes die Datei „install.php“ im Unterverzeichnis „typolight“ der TYPOlight-Installation über einen beliebigen Webbrowser aufgerufen werden. Wenn das System also beispielsweise auf einem Webserver, der unter der Domain „http://www.typolight.org“ erreichbar ist, installiert wurde, so müsste nun zuerst die Datei „install.php“ im Verzeichnis „http://www.typolight.org/typolight/“ aufgerufen werden. Ist die Seite geladen, wird die Eingabe eines Passworts zum Starten des Installations-Skripts gefordert. Dieses Passwort lautet direkt nach der Installation standardmäßig noch „typolight“.

Nach Eingabe des Passwortes wird das eigentliche Installations-Skript gestartet. Dieses verlangt zunächst das Anlegen eines neuen Installations-Passworts, welches sich vom Standard-Passwort unterscheidet, sofern dies nicht schon geschehen ist. Erst dann kann die Installation fortgesetzt werden.

Als nächstes müssen ein Encryption-Key sowie die Zugangsdaten zur Datenbank eingegeben werden. Der Encryption-Key wird zur Verschlüsselung sämtlicher Datensätze in TYPOlight verwendet, die nicht im Klartext in der Datenbank abgespeichert werden sollen. Dieser Encryption-Key sollte daher auch nicht mehr geändert werden, sobald bereits verschlüsselte Daten in der Datenbank vorliegen. Denn diese können mit einem veränderten Schlüssel nicht mehr korrekt dechiffriert werden. Auf Wunsch kann TYPOlight bei der Installation automatisch einen solchen Schlüssel generieren.

[35] Zu finden unter http://www.typolight.org/download.html

Nachdem auch die korrekten Zugangsdaten zur Datenbank eingegeben wurden und TYPOlight eine Datenbankverbindung aufbauen konnte, überprüft das System das Vorhandensein von Datenbanktabellen entsprechend der aktuellen Modulkonfiguration.[36] Fehlende Tabellen können nun bei Bedarf erstellt werden. Bereits vorhandene Tabellen, die in der Konfiguration nicht vorgesehen sind, können gelöscht werden, sofern sie nicht mehr anderweitig benötigt werden. Bei einer Neuinstallation des Systems dürften in der Regel noch keine Tabellen in der Datenbank vorhanden sein, daher sollten nun alle benötigten Tabellen mit Hilfe des Installationsskripts angelegt werden.

Die eigentliche Installation ist nun abgeschlossen, bei Bedarf kann allerdings noch eine Beispiel-Website importiert werden. Diese ist für unerfahrene Benutzer in TYPOlight gut geeignet, um das System besser kennenzulernen. Wenn jedoch eine neue Website angelegt werden soll, ist es nicht empfehlenswert, die Beispiel-Seiten zu importieren.

Zuletzt muss noch ein Admin-Benutzer durch Eingabe von Name, Benutzername und Passwort angelegt werden, da sonst kein Zugang zum Backend möglich ist.

Sind alle Schritte erledigt, so kann das Backend von TYPOlight durch Klick auf den entsprechenden Link ganz unten im Installations-Skript aufgerufen werden. Hier ist der Login durch Eingabe von Benutzername und Passwort des soeben angelegten Admin-Benutzers möglich.

4.2 Module

Module stellen das Kernprinzip von TYPOlight dar. Fast das gesamte System ist sowohl im Frontend als auch im Backend aus verschiedenen Modulen aufgebaut. Diese dienen jeweils einer bestimmten Aufgabe: im Frontend z.B. der Darstellung einer Seitennavigation, der Auflistung von News und vielem mehr, im Backend z.B. der Verwaltung von Benutzern oder dem Verfassen von Artikeln.

Wenn eine neue Website mit TYPOlight angelegt wird, sollten daher zunächst alle Module erstellt werden, die auf jeder Seite benötigt werden, da diese später in ein Seitenlayout[37] eingebunden werden. Dazu muss zunächst der Punkt „Module“ im Bereich „Layout“ der Backend-Navigation aufgerufen werden. Hier kann über die Aktion „Neues Modul“ ein neues Modul angelegt werden. Dadurch wird die Einstellungsseite des Moduls angezeigt. Als erstes sollte ein Name für das Modul eingegeben werden, da dieser zwingend von TYPOlight benötigt wird. Optional kann für jedes Modul auch eine Überschrift vergeben werden, welche stets über dem Modul eingeblendet wird.

[36] Siehe auch Kapitel 6.4 - Konfiguration von TYPOlight
[37] Siehe auch Kapitel 4.3.3 - Anlegen eines Seitenlayouts

Unter dem Punkt „Modultyp“ kann ausgewählt werden, welche Art von Modul angelegt werden soll. Abhängig vom hier gewählten Eintrag verändern sich auch alle weiteren zur Verfügung stehenden Konfigurationsmöglichkeiten. Alle Optionen sind in der Regel selbsterklärend, da zu jedem Punkt eine kurze Beschreibung angezeigt wird.

So können nun nacheinander alle Module angelegt werden, die überall auf der Website benötigt werden. In der Regel handelt es sich dabei mindestens um ein Navigationsmodul, ggf. können auch noch weitere Module wie z.B. ein Logo, ein Navigationspfad, ein Login-Formular u.v.m. angelegt werden. Alle erstellten Module können später sowohl in ein Seitenlayout als auch direkt in einen Artikel[38] als Inhaltselement eingebunden werden.

Für die Website des Verbandes der Immobilienverwalter werden zunächst Module für die Darstellung der Seitennavigation sowie des Navigationspfades, ein Login-Formular und ein Modul für die Anmeldung zum Newsletter benötigt. Weiterhin werden das Bild im Seitenkopf, das BSI-Logo sowie die Links auf die Verwaltersuche und Handwerkeranmeldung ebenfalls als Modul angelegt, und zwar vom Typ HTML. Die bisher erstellten Module sollen auf allen Seiten sichtbar sein und müssen daher später in ein Seitenlayout[39] eingebunden werden.

Weiterhin können bereits ein News- sowie ein Kalendermodul angelegt werden, da diese später ebenfalls zur Realisierung der aktuellen Meldungen und des Veranstaltungskalenders auf der Website benötigt werden.

Zur Umsetzung der Marktplatz-Funktion wird die Third-Party-Extension „EFG“ genutzt. Diese ermöglicht die Erstellung erweiterter Formulare, welche Frontend-Mitglieder zum Eintragen von Daten in die Datenbank nutzen können. Alle Datensätze können über das Modul „Formulardaten Auflistung“ im Frontend wieder aufgelistet und von dem Mitglied, welches den Datensatz erstellt hat, nachträglich bearbeitet werden.

Der Bereich „Literaturempfehlungen“ wird ebenfalls durch eine Third-Party-Extension realisiert. Die Extension „Catalog“ ermöglicht das Anlegen beliebiger Artikelkataloge im Backend und deren Auflistung im Frontend.

Alle Modultypen hier zu erläutern würde den Rahmen dieser Diplomarbeit übersteigen. Eine Beschreibung aller existierenden Typen kann auf der TYPOlight-Website[40] abgerufen werden.

[38] Siehe auch Kapitel 4.5 - Artikel
[39] Siehe auch Kapitel 4.3.3 - Anlegen eines Seitenlayouts
[40] Zu finden unter http://www.typolight.org/wiki/german:modules

4.3 Layout

Ein Seitenlayout legt fest, wie eine einzelne Webseite in TYPOlight aufgebaut ist, d.h. welche Layout-Bereiche zur Verfügung stehen und welche Module auf einer Seite mit diesem Layout immer vorhanden sein sollen. Ein Seitenlayout in TYPOlight besteht im Wesentlichen aus zwei Komponenten: einem Template und einem oder mehreren Stylesheets. Dieses Kapitel soll nun das Erstellen eines Seitenlayouts mit Hilfe der beiden genannten Komponenten erläutern.

4.3.1 Templates

Templates in TYPOlight sind Vorlagen-Dateien, die überwiegend HTML-Code enthalten, aber auch einige Variablen- und Methodenaufrufe sowie Kontrollstrukturen in der Skriptsprache PHP, um eine gewisse Darstellungslogik realisieren zu können. Templates dienen in erster Linie zur Definition von Layout-Bereichen und zur Festlegung der Reihenfolge, in welcher diese dargestellt werden.

Template-Dateien müssen in TYPOlight immer auf die Endung „.tpl“ enden. Außerdem muss jede Template-Datei ein bestimmtes Präfix besitzen, welches durch einen Unterstrich getrennt dem eigentlichen Dateinamen vorangesetzt wird und die Zuordnung zu einem bestimmten Modultyp signalisiert. Das System bringt schon eine Reihe von Standard-Templates für alle Module mit sich, welche in den jeweiligen Modul-Verzeichnissen des Systems abgespeichert sind. Diese Dateien sollten jedoch nicht modifiziert werden, da sonst ein reibungsloses Funktionieren des Systems nicht mehr gewährleistet ist. Außerdem werden an den System-Templates vorgenommene Änderungen bei einem Update von TYPOlight unter Umständen überschrieben. Benutzerdefinierte Templates sollten daher mit dem richtigen Präfix im Verzeichnis „Templates“ des Systems gespeichert werden.

Templates, welche zur Verwendung mit Seitenlayouts dienen sollen, müssen mit dem Präfix „fe_“ beginnen. Das Standard-Template, welches bereits mitgeliefert wird, bietet schon viele verschiedene Einsatzmöglichkeiten und kann für alle gängigen Seitenaufteilungen ohne Änderungen eingesetzt werden. Möglich sind mit diesem Template Ein-, Zwei- oder Dreispalten-Layouts, welche zusätzlich auch mit einem Kopf- und Fußbereich versehen werden können.[41] Außerdem ermöglicht dieses Template bereits die Definition und Verwendung benutzerspezifischer Layout-Bereiche, welche entweder nach der Kopfzeile, im Hauptbereich oder vor der Fußzeile eingebunden werden können. Erst wenn eigene Layout-Bereiche an anderen Stellen oder eine besondere Seitenaufteilung benötigt werden, wird eine modifizierte Template-Datei benötigt.

[41] Siehe auch Kapitel 4.3.3 - Anlegen eines Seitenlayouts

Für den Internetauftritt des Verbandes der Immobilienverwalter Mitteldeutschland e.V. ist das Standard-Template jedoch ausreichend, da hier eine einfache Seitenaufteilung in drei Spalten sowie Kopf- und Fußbereich gegeben ist. Diese kann mit dem vorhandenen Template problemlos realisiert werden.

4.3.2 Style Sheets

Cascading Style Sheets (CSS) stellen eine deklarative Stylesheet-Sprache zur Formatierung von strukturierten Dokumenten, insbesondere HTML und XML dar. CSS-Regeln legen dabei fest, wie besonders ausgezeichnete Inhalte oder Bereiche innerhalb des Dokuments formatiert werden sollen.[42] Somit wird eines der wichtigsten Prinzipien der barrierefreien Gestaltung von Webseiten realisierbar, nämlich die Trennung von Inhalt und optischer Gestaltung. Weiterführende Informationen zu Cascading Style Sheets sind auf der Website des World Wide Web Consortiums[43] zu finden.

In TYPOlight dienen Style Sheets ebenfalls ausschließlich der Formatierung von speziell ausgezeichneten Bereichen. Das System bietet hierfür die Möglichkeit, eigene Style Sheets im entsprechenden Backend-Bereich anzulegen. Zu jedem Style Sheet können beliebig viele Formatierungsregeln hinterlegt werden, welche direkt über entsprechende Formularfelder eingegeben werden können. So können auch weniger versierte Benutzer ihre eigenen Style Sheets anlegen. Alle Regeln werden jedoch zunächst nur in der Datenbank abgespeichert, es wird noch keine CSS-Datei erstellt, welche zur Formatierung einer bestimmten Webseite genutzt werden könnte.

Auch der Import von bereits bestehenden CSS-Dateien ist möglich. Diese müssen dazu zunächst mit Hilfe der Dateiverwaltung auf den Server geladen und beim Import anschließend im Dateisystem ausgewählt werden. Die Datei wird nun vom System geparst, alle dabei gefundenen gültigen Formatierungsregeln werden zusammen unter einem neuen Style-Sheet-Eintrag in der Datenbank abgespeichert. Diese Regeln können auch nachträglich im Backend editiert werden.

Für die Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. wird das Style Sheet zur Formatierung der einzelnen Elemente auf konventionellem Weg erstellt, also ohne Nutzung des in TYPOlight integrierten Style-Sheet-Editors. Die so erstellte CSS-Datei wird anschließend als neues Style Sheet ins System importiert und steht nun so zur Verwendung in einem Seitenlayout zur Verfügung.

---

[42] Vgl. O.V.: Cascading Style Sheets, 2008, Internet: http://de.wikipedia.org/wiki/Cascading_Style_Sheets
[43] Zu finden unter http://www.w3.org/Style/CSS

4.3.3 Anlegen eines Seitenlayouts

Ein Seitenlayout in TYPOlight stellt im Grunde die Kombination aus einem Template und einem oder mehreren Style Sheets sowie verschiedenen Modulen dar, wobei alle Bereiche des Templates jedoch abhängig von der Konfiguration des übergeordneten Seitenlayouts dargestellt werden.

So kann über die Einstellmöglichkeiten des Seitenlayouts festgelegt werden, wie viele Spalten angezeigt werden oder ob der Kopf- bzw. Fußbereich sichtbar ist. Die Breite des gesamten Layout-Bereichs kann dabei statisch, also mit einer festen Weite, oder dynamisch, also abhängig von der Größe des Browserfensters, angelegt werden. Auch Höhe von Kopf- und Fußbereich sowie die Breite der einzelnen Spalten können frei gewählt werden. Außerdem können dem Seitenlayout natürlich alle vorher erstellten oder importierten Style Sheets zugeordnet werden.

Der Layout-Entwurf für die geplante Website sieht eine Gesamtbreite von 990 Pixeln sowie eine linke und rechte Spalte mit einer Breite von jeweils 170 Pixeln vor. Der Kopfbereich soll 170 Pixel und der Fußbereich 20 Pixel hoch sein. Alle diese Werte können direkt in der Konfiguration des Seitenlayouts eingetragen werden. Dadurch werden von TYPOlight automatisch entsprechende CSS-Regeln erstellt, welche die Layout-Bereiche des Templates auf die gewünschten Maße skalieren.

Zu den weiteren Konfigurationsmöglichkeiten eines Seitenlayouts gehören z.B. die Eingabe einer Google Analytics ID, das Festlegen eines Body-Onload-Ereignisses, das Einfügen zusätzlicher Header-Tags oder das Bereitstellen von RSS-Feeds für Nachrichten und Kalendereinträge. Außerdem kann im Seitenlayout die Dokumenttypdefinition festgelegt werden.

Ein wichtiger Punkt beim Anlegen eines Seitenlayouts ist das Einbinden von Modulen. Alle bisher erstellten Module können dazu verwendet werden und stehen dann auf allen Seiten zur Verfügung, welche dieses Seitenlayout verwenden. Jedes Modul kann beliebig oft eingebunden werden, wobei dabei auch der Layout-Bereich, in welchem dieses erscheinen soll, frei gewählt werden kann. In der Regel sollte zumindest ein Modul für die Seitennavigation eingebunden werden.

In diesem Fall können nun die Module für Navigation, Navigationspfad, Newsletter-Anmeldung usw. in die entsprechenden Layout-Bereiche eingebunden werden.

4.4 Seitenstruktur

Nachdem mindestens ein Seiten-Layout angelegt wurde, kann auch die Struktur der geplanten Website erstellt werden. Dies kann auch schon im Vorfeld erledigt werden, allerdings ist ohne verfügbares Seitenlayout die Anzeige einer Seite im Frontend nicht möglich.

Das Anlegen von Seiten ist in TYPOlight innerhalb des Backend-Moduls „Seitenstruktur“ möglich, welches in Abbildung 4-1 zu sehen ist. Dort ist zunächst nur der Seitenursprung (Markierung #1) vorhanden, dessen Name in der globalen Konfiguration unter „Einstellungen“ im Backend definiert werden kann.

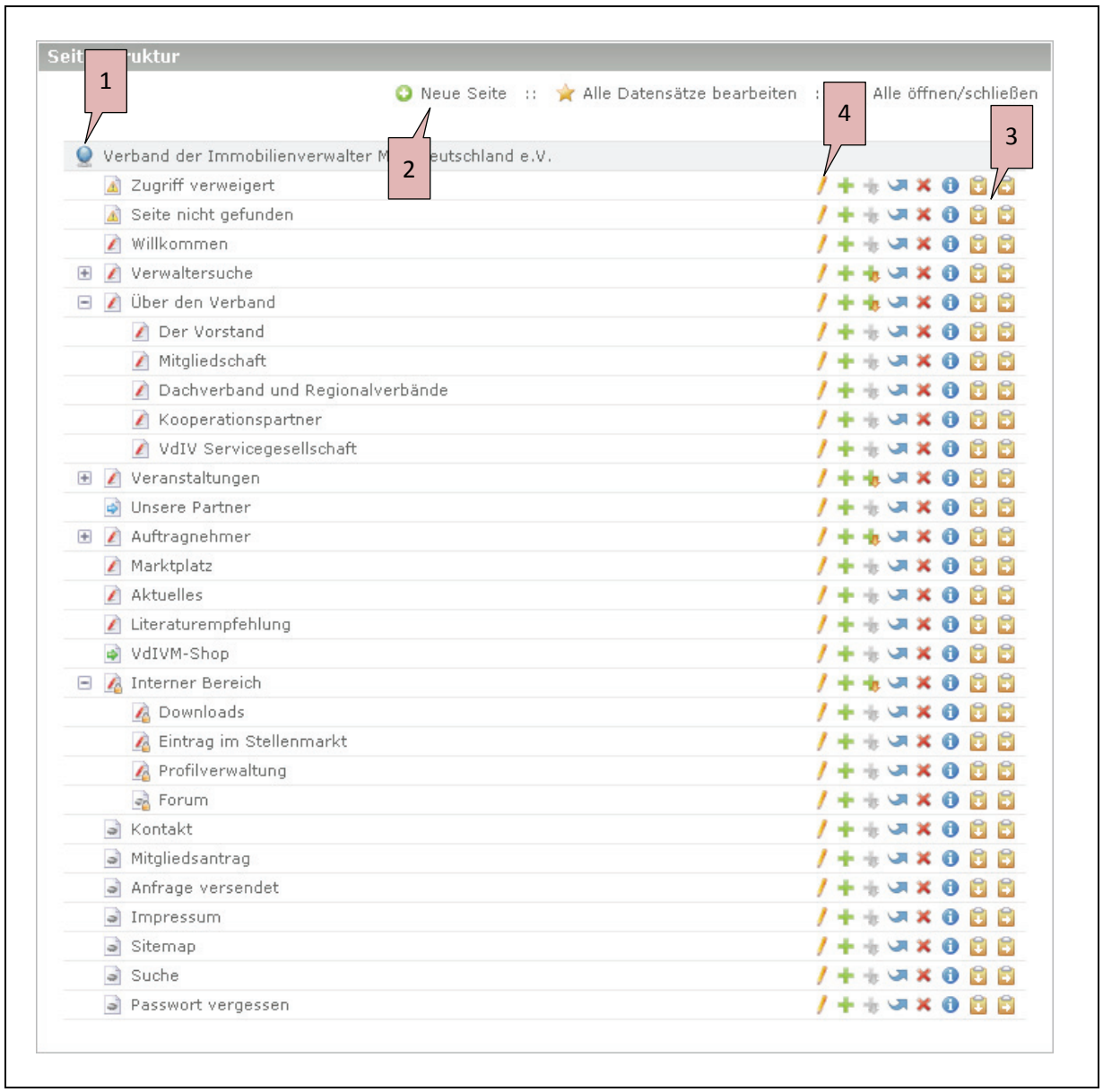

Abbildung 4-1: Screenshot des Backend-Moduls "Seitenstruktur" in TYPOlight

Ausgehend vom Seitenursprung können beliebig viele Unterseiten erstellt werden, von denen jede wiederum beliebig viele weitere Unterseiten enthalten kann. Bereits erstellte Seiten können beliebig verschoben oder anderen Seiten untergeordnet werden.

Um eine neue Seite anzulegen, muss zunächst der entsprechende Befehl (Markierung #2) oberhalb des Seitenbaumes angeklickt werden. Danach muss festgelegt werden, an welcher Position im Sei-

tenbaum die neue Seite erstellt werden soll. Dazu dienen die nun sichtbaren Einfüge-Buttons (Markierung #3) neben jeder Seite. Wird ein nach unten weisender Button aktiviert, so wird die neue Seite direkt nach der zugehörigen Seite auf der gleichen Hierarchie-Ebene in den Seitenbaum eingefügt. Der nach rechts weisende Button dient stattdessen dazu, die neue Seite eine Ebene tiefer in den Seitenbaum einzufügen. Sind in dieser Ebene bereits Seiten vorhanden, so wird die neue Seite noch vor den anderen Seiten eingefügt.

Anschließend kann die neue Seite konfiguriert werden. Für bereits vorhandene Seiten sind kann der gleiche Konfigurationsdialog durch Klick auf den Bearbeiten-Button (Markierung #4) der entsprechenden Seite aufgerufen werden. Hier kann neben einem Seitentitel auch ein Alias vergeben werden, welcher später dazu verwendet werden kann, die Seite im Browser direkt durch Eingabe des Seitenalias anstatt der Seiten-ID aufzurufen. Diese Maßnahme dient einerseits der Benutzerfreundlichkeit, andererseits auch der Suchmaschinenoptimierung der Website.

Weiterhin kann der Seitentyp ausgewählt werden. Nachfolgend werden alle möglichen Seitentypen mit einer kurzen Erläuterung aufgelistet.

Reguläre Seite

Dieser Seitentyp dient zum Anlegen einer normalen Seite, welche mit Artikeln und somit auch mit Inhaltselementen gefüllt werden kann. Dies ist also der in der Regel am häufigsten benötigte Seitentyp. Einer Seite dieses Typs kann ein zuvor erstelltes Seiten-Layout und eine Cache-Verfallszeit zugewiesen werden.

Weiterleitung zu einer externen URL

Mit diesem Seitentyp kann eine Weiterleitung zu einer Internetadresse außerhalb der eigenen Website eingerichtet werden. Dabei kann der Weiterleitungstyp ausgewählt werden, zur Verfügung stehen temporär und permanent. Abhängig davon wird beim Aufruf dieser Seite ein entsprechender HTTP-Statuscode (301 oder 307) an den Client gesendet, um die Weiterleitung zu realisieren.

Weiterleitung zu einer anderen Seite

Bei Auswahl dieses Typs kann über den anschließend angezeigten Seitenbaum eine Seite innerhalb der eigenen Website ausgewählt werden, auf welche die zu konfigurierende Seite verweisen soll. Hier kann analog zur externen Weiterleitung ebenfalls zwischen temporärer oder permanenter Weiterleitung unterschieden werden.

Startpunkt einer Website

Dieser Seitentyp dient als Seitenursprung einer neuen Website und erlaubt somit die Verwaltung mehrerer Internetauftritte mit nur einer TYPOlight-Installation. Für jede Seite dieses Typs kann eine

separate Domain oder Subdomain definiert werden, so dass der Client abhängig von der Adresse, welche er angefordert hat, direkt auf die Startseite der entsprechenden Website weitergeleitet wird. Außerdem kann dieser Seitentyp eingeschränkt zur Realisierung eines mehrsprachigen Internetauftritts genutzt werden, da für jeden Startpunkt eine eigene Sprache festgelegt werden kann. Ein Benutzer wird dann abhängig von seiner im Browser eingestellten Sprache auf den jeweiligen Startpunkt weitergeleitet, sofern diese Sprache vorhanden ist. Ansonsten kann auch ein Sprachen-Fallback für einen bestimmten Startpunkt definiert werden, so dass alle Benutzer, die in ihrem Browser eine andere als die innerhalb der Website vorhandenen Sprachen eingestellt haben, auf diesen Startpunkt umgeleitet werden.

Fehler 403 (Zugriff verweigert)

Dieser Typ wird benötigt, falls die Website geschützte Bereiche enthält, die nur für bestimmte Frontend-Mitglieder zugänglich sein sollen. Allen anderen Besuchern wird beim Versuch, eine geschützte Seite zu öffnen, diese Fehlerseite angezeigt. Somit kann die Standard-Fehlermeldung des Webservers durch eine eigene Seite ersetzt werden. Alle Seiten dieses Typs können wie reguläre Seiten bearbeitet und mit Inhalten gefüllt werden.

Fehler 404 (Seite nicht gefunden)

Eine Seite dieses Typs wird immer angezeigt, wenn der Client versucht, eine nicht vorhandene Seite aufzurufen. Dieser Seitentyp dient also analog zum vorherigen Typ zur Ersetzung der Webserver-Fehlermeldung durch eine benutzerdefinierte Seite und kann ebenfalls wie reguläre Seiten bearbeitet werden.

Neben den bereits genannten Konfigurationsmöglichkeiten der jeweiligen Seitentypen können weiterhin alle Seiten geschützt und somit nur bestimmten Mitgliedergruppen im Frontend zugänglich gemacht werden. Außerdem kann für jede Seite individuell festgelegt werden, welche Operationen für bestimmte Benutzer oder Benutzergruppen im Backend erlaubt oder verboten sind.

So können nun nacheinander alle Seiten angelegt werden, wie sie in der Seitenstruktur in Kapitel 2.2.2 definiert wurden.

4.5 Artikel

Artikel dienen in TYPOlight zur Darstellung von Inhaltselementen und Modulen. Jede Seite kann beliebig viele Artikel enthalten, die außerdem auch allen Layout-Bereichen der Seite zugeordnet werden können. Abbildung 4-2 zeigt das Backend-Modul „Artikel“ in TYPOlight.

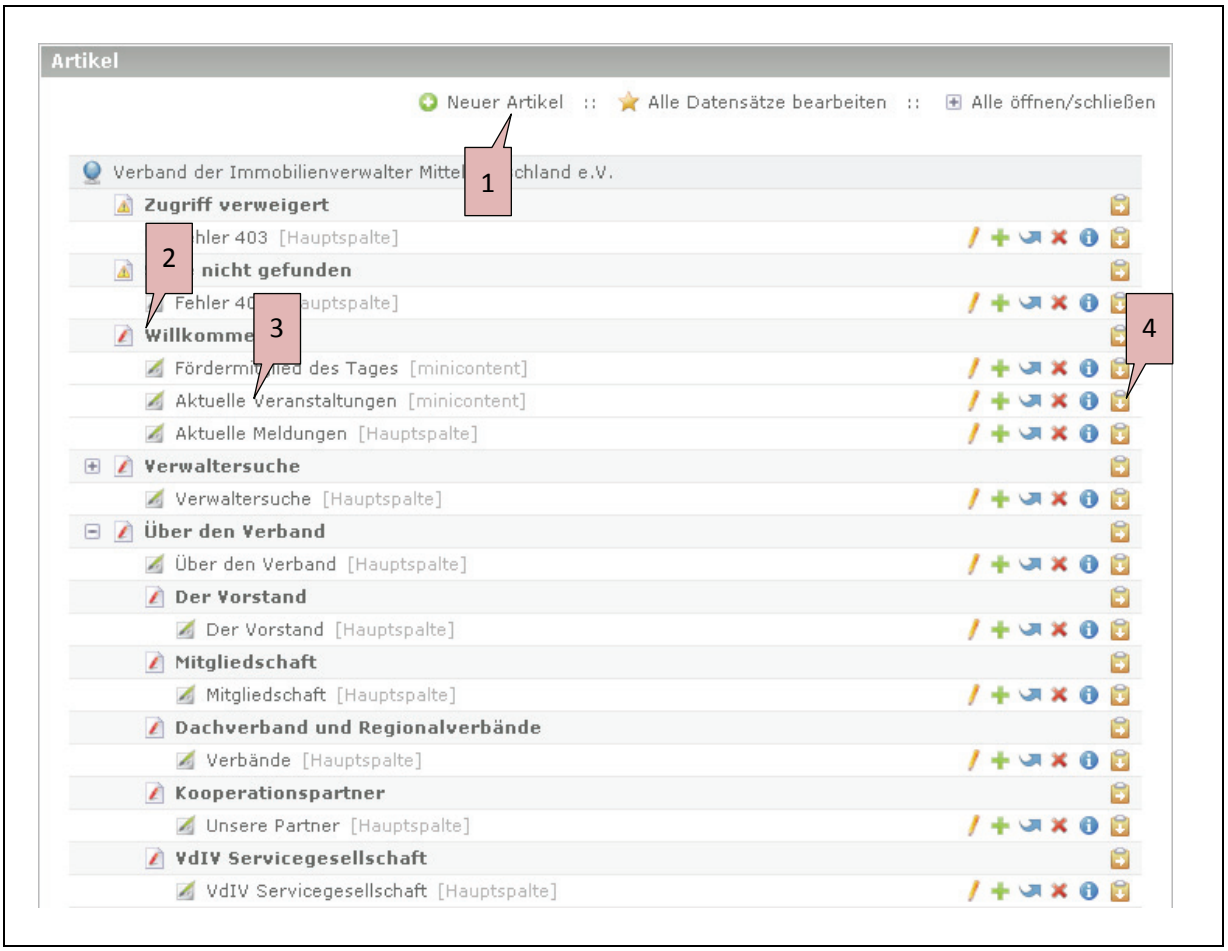

Abbildung 4-2: Screenshot des Backend-Moduls "Artikel" in TYPOlight

Das Anlegen eines neuen Artikels erfolgt ähnlich wie das Anlegen einer neuen Seite. Zunächst muss der Button „Neuer Artikel“ (Markierung #1) aktiviert werden, woraufhin im Seitenbaum neben jeder vorhandenen Seite (Markierung #2) und jedem bereits vorhandenen Artikel (Markierung #3) ein Einfüge-Button (Markierung #4) eingeblendet wird. Wird einer dieser Buttons betätigt, wird **in** die jeweilige Seite bzw. **nach** dem jeweiligen Artikel ein neuer Artikel angelegt und als nächstes dessen Konfigurationsdialog aufgerufen. Artikel können nur in Seiten eingefügt werden, das Zuordnen eines Artikels zu einem anderen Artikel ist nicht möglich.

Im Konfigurationsdialog muss zunächst ein Name und optional ein Artikelalias vergeben werden. Zusätzlich kann hier unter allen vorhandenen Backend-Benutzern ein für den Artikel verantwortlicher Autor ausgewählt werden. Weiterhin kann bestimmt werden, in welchem Layout-Bereich der übergeordneten Seite der Artikel dargestellt werden soll. Auch kann hier ein Vorschau-Text (Teaser) für den Artikel eingegeben werden, welcher optional zunächst als Platzhalter angezeigt wird und über den der eigentliche Artikel aufgerufen werden kann. Außerdem kann der Artikel direkt veröffentlicht

werden oder ein Veröffentlichungszeitraum festgelegt werden, innerhalb dessen der Artikel öffentlich abrufbar ist.

Sind alle erforderlichen Einstellungen vorgenommen worden, kann der Artikel gespeichert werden und ggf. gleich zum nächsten Schritt übergegangen werden, dem Einfügen von Inhaltselementen.

Für die Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. muss im Grunde für jede Seite mindestens ein Artikel angelegt werden, da nur innerhalb eines Artikels Module und Inhaltselemente positioniert werden können. Alternativ könnte für bestimmte Seiten wie z.B. die Startseite mit den aktuellen Meldungen oder die Seite mit dem Veranstaltungskalender ein eigenes Seitenlayout angelegt werden, in welches die entsprechenden Module eingebunden werden. So könnte bei diesen Seiten auf den Einsatz eines Artikels verzichtet werden. Dann können allerdings keine ergänzenden Informationen neben den News- bzw. Kalendermodulen hinterlegt werden. Hierfür wäre wieder ein Artikel notwendig, um z.B. Textelemente einzufügen.

4.6 Inhaltselemente

Inhaltselemente dienen in TYPOlight zur Darstellung von textuellen oder tabellarischen Inhalten oder zur Anzeige von Grafiken. Weiterhin können Dateidownloads als Inhaltselement eingefügt werden. Der Unterschied zu Modulen ist, dass Inhaltselemente jeweils nur einmal verwendet werden können, nämlich innerhalb des Artikels, in welchem sie angelegt werden. Module dagegen können beliebig oft in der gesamten Website eingesetzt werden, sie können z.B. ebenfalls die Rolle eines Inhaltselements einnehmen und innerhalb eines Artikels eingefügt werden. Hier dienen sie dann nicht zur Darstellung von Inhalten, sondern erfüllen meist eine ganz bestimmte Funktionalität.

Abbildung 4-3 zeigt die Inhaltselemente (Markierung #3), innerhalb eines Artikels. Jedes davon kann über einen entsprechenden Button ein- oder ausgeblendet (Markierung #8), verschoben (Markierung #6), dupliziert (Markierung #5), bearbeitet (Markierung #4) oder gelöscht (Markierung #7) werden. Über den Inhaltselementen werden Informationen zum übergeordneten Artikel angezeigt (Markierung #2). Diese können über den in Kapitel 4.5 bereits beschriebenen Konfigurationsdialog auch nachträglich noch bearbeitet werden.

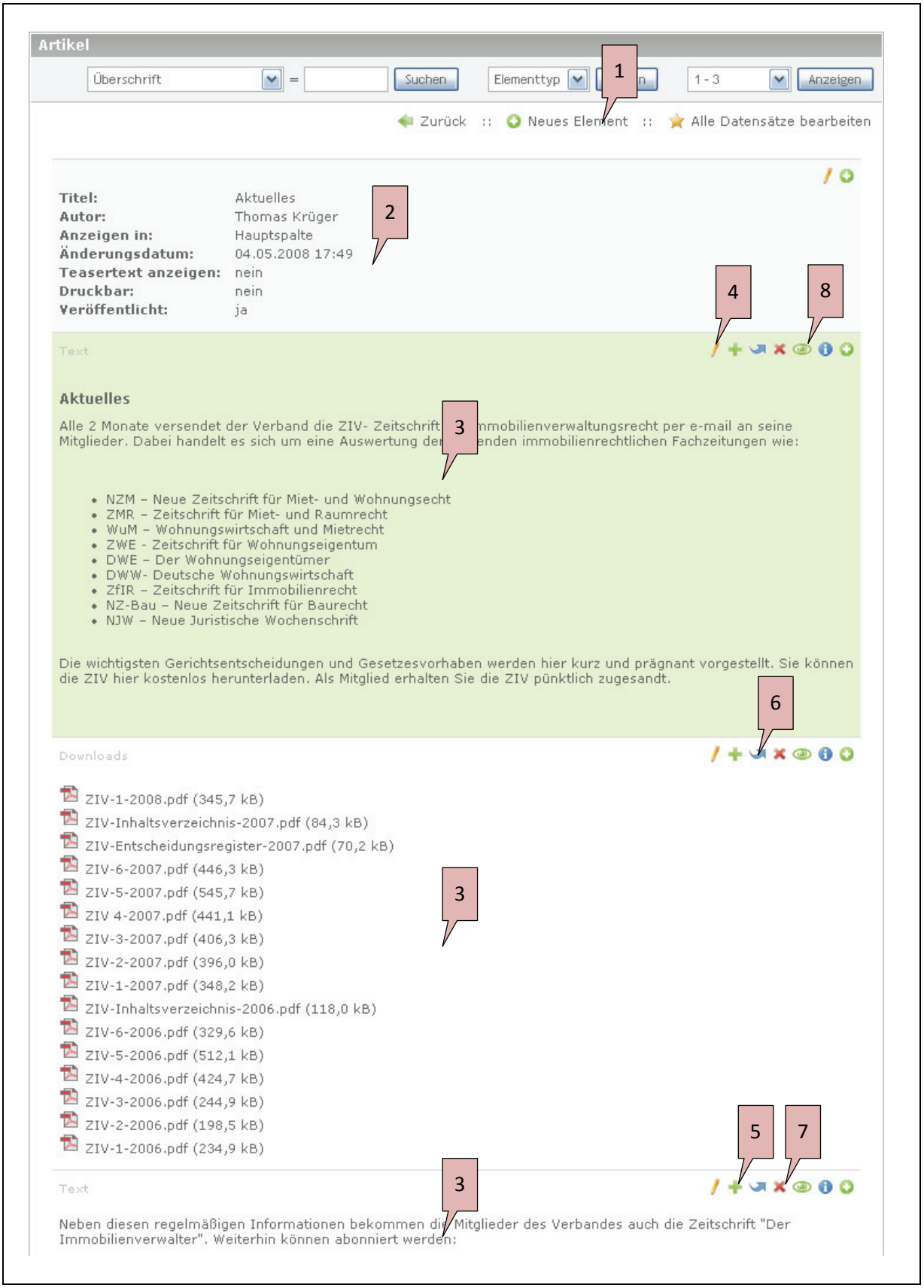

Abbildung 4-3: Screenshot eines Artikels mit Inhaltselementen im Backend von TYPOlight

Um ein neues Inhaltselement einzufügen, muss der entsprechende Button (Markierung #1) betätigt werden. Anschließend gelangt man zum Konfigurationsdialog, wo zunächst die Art des Inhaltselements ausgewählt werden kann. Abhängig davon werden verschiedene Konfigurations- und Eingabemöglichkeiten angeboten, über welche z.B. Text eingegeben oder Bilder ausgewählt werden können.

Alle Arten von Inhaltselementen hier zu erläutern würde den Rahmen dieser Diplomarbeit übersteigen. Eine Auflistung aller verfügbaren Inhaltselemente mit einer entsprechenden Beschreibung kann daher auf der TYPOlight-Website[44] abgerufen werden.

Für die Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. können nun alle vorgesehenen Texte und Bilder in die einzelnen Artikel eingefügt werden. Außerdem können die bereits angelegten Module für aktuelle Meldungen, Veranstaltungskalender usw. in die entsprechenden Artikel eingebunden werden.

Für bestimmte Funktionalitäten wie die Verwaltersuche oder die wechselnde Anzeige von Mitglieder-Logos existieren bisher jedoch noch keine geeigneten Inhaltselemente oder Module. Daher beschäftigen sich die folgenden zwei Kapitel mit dem Aufbau von TYPOlight und der Programmierung von Extensions, so dass die noch fehlenden Funktionalitäten ebenfalls realisiert werden können.

[44] Zu finden unter http://www.typolight.org/wiki/german:modules

5 Aufbau und Funktionsweise von TYPOlight

Dieses Kapitel soll den grundlegenden Aufbau von TYPOlight und dessen Umsetzung des Model-View-Controller-Prinzips erläutern. Deren Verständnis ist nötig, um Erweiterungen für das System zu entwickeln. Dabei sollen auch die wichtigsten Klassen des Systems und deren Zusammenhang untereinander dargestellt werden.

5.1 Model-View-Controller-Prinzip

TYPOlight arbeitet nach dem Model-View-Controller-Prinzip (MVC). Dieses Prinzip bezeichnet ein Entwurfsmuster zur Aufteilung von komplexen Software-Systemen in drei Komponenten: Datenmodell (engl. *Model*), Präsentation (engl. *View*) und Programmsteuerung (engl. *Controller*). Durch diese Aufteilung wird eine flexible Architektur des Systems ermöglicht, wodurch eine Änderung oder Erweiterung des Systems vereinfacht wird und außerdem eine gute Wiederverwendbarkeit der einzelnen Komponenten ermöglicht wird.[45] Nachfolgend wird in diesem Kapitel die Umsetzung dieses Prinzips in TYPOlight erläutert.

5.1.1 Model

Das Datenmodell wird in TYPOlight durch sogenannte **Data Container** repräsentiert. Dieser Begriff ist im Prinzip vergleichbar mit dem Begriff der Entitätsmengen, wie er aus dem Entity-Relationship-Modell bekannt ist. Den Großteil dieser Data Container stellen alle in der Datenbank enthaltenen Tabellen und deren Datensätze dar. Dazu gehören aber auch Verzeichnisse in dem Teil des Dateisystems, der von Benutzern des Systems bearbeitet werden kann (standardmäßig Verzeichnis „tl_files“), und außerdem teilweise auch Konfigurationsdateien, die bestimmte Daten und Einstellungen beinhalten.

Alle Data Container werden durch das globale **Data Container Array (DCA)** beschrieben. Dieses legt zunächst fest, wie zugehörige Entitäten des Data Containers gespeichert werden. Dies kann wie bereits erwähnt entweder in einer Datenbanktabelle, in einem Verzeichnis (nur bei anderen Verzeichnissen oder Dateien) oder in einer Konfigurationsdatei geschehen. Weiterhin beschreibt das Data Container Array die Beziehung der Entitäten untereinander, legt also eine Eltern-Kind-Beziehung zwischen verschiedenen Entitäten fest. Weiterhin werden im Data Container Array alle Attribute

[45] Vgl. O.V.: Model View Controller, 2008, Internet: http://de.wikipedia.org/wiki/Model_View_Controller

(Datenfelder) einer Entitätsmenge definiert und gewisse Regeln für deren zulässige Werte aufgestellt.

Zusätzlich zu der Beschreibung des Datenmodells enthält das Data Container Array gewisse Parameter, die für die Verarbeitung der Daten im Backend benötigt werden. Dazu gehört u.a. die Auflistung aller globalen Operationen, die für den jeweiligen Data Container im Backend erlaubt sind, wie z.B. das Erstellen neuer Entitäten. Außerdem wird im Data Container Array geregelt, welche Operationen für jede einzelne Entität möglich sind. Dazu können u.a. das Editieren, Löschen, Duplizieren, Verschieben und Anzeigen gehören.

Weiterhin regelt das Data Container Array, welche Attribute einer Entitätsmenge im Backend dargestellt werden und bearbeitbar sein sollen sowie nach welchen Attributen eine Entitätsmenge durchsucht, sortiert oder gefiltert werden kann.

Eine komplette Auflistung aller Konfigurationsmöglichkeiten des Data Container Arrays ist auf der TYPOlight-Website[46] zu finden.

Zusätzlich werden vom Model verschiedene Klassen zur Verfügung gestellt, die das Einfügen von neuen Daten bzw. Dateien oder das Selektieren, Bearbeiten oder Löschen von bereits vorhandenen Daten bzw. Dateien abhängig vom verwendeten Data Container ermöglichen.

5.1.2 Controller

Sowohl für Frontend als auch für Backend existieren diverse Controller-Klassen in TYPOlight, welche die Steuerung von Views sowie die Verarbeitung von Benutzereingaben übernehmen. Alle diese Klassen werden entweder direkt von der Klasse „Controller“ oder von einem ihrer Nachfahren abgeleitet.

Die wohl wichtigste Komponente im Backend ist die Klasse „DataContainer“ und deren direkte Nachfahren, welche ausgehend vom globalen Data Container Array die Verwaltung des gesamten Datenmodells ermöglichen. Dazu kontrollieren sie entsprechende Views, welche für die Auflistung von Daten bzw. für die Entgegennahme von Benutzereingaben und -interaktionen zuständig sind. Weiterhin stellen diese Klassen Methoden für das Erstellen, Bearbeiten und Löschen von Datensätzen bereit. Alle eingegeben Daten werden entsprechend der im Data Container Array definierten Regeln auf Gültigkeit geprüft, bevor sie zur Speicherung an die entsprechenden Klassenmethoden der verschiedenen Data Container im Datenmodell weitergegeben werden.

[46] Zu finden unter http://typolight.org/table-configuration.html

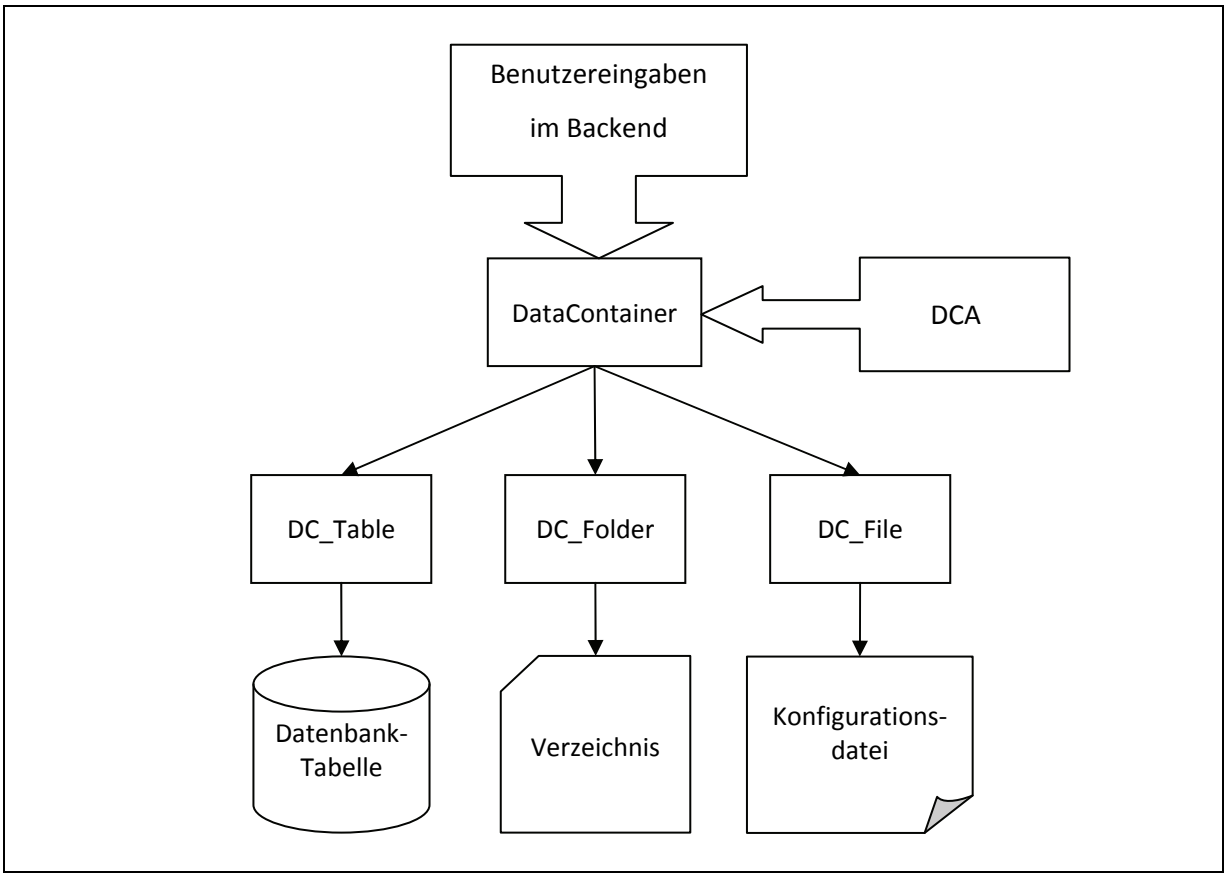

Abbildung 5-1: Schematischer Ablauf beim Speichern von Benutzereingaben im Backend von TYPOlight

Abstrakt gesehen werden alle Benutzereingaben im Backend an einen Data Container übergeben. Dieser wird durch das globale Data Container Array konfiguriert und beschrieben. Abhängig von dieser Konfiguration werden die eingegebenen Daten von der entsprechenden Data-Container-Klasse am jeweiligen Ort gespeichert, also entweder in einer Datenbanktabelle, in einem Verzeichnis (bei Dateien) oder in einer Konfigurationsdatei. Dieser Ablauf ist schematisch in Abbildung 5-1 dargestellt.

Im Backend existieren noch weitere Controller-Klassen, die abhängig vom jeweiligen Backend-Modul unterschiedlichen Einsatzzwecken dienen. Dazu gehören z.B. das Versenden von Newslettern und das Erzeugen von Stylesheet-Dateien.

Für das Frontend existieren ebenfalls zahlreiche Controller-Klassen. Die meisten dieser Klassen sind Nachfahren der Klasse „Module“ oder „ContentElement“, welche jeweils Funktionalitäten für ein bestimmtes Modul oder Inhaltselement zur Verfügung stellen.[47]

[47] Siehe auch Kapitel 5.2 - Komponenten im Frontend

5.1.3 View

Die View-Komponente wird in TYPOlight durch die Klasse „Template“ sowie alle Template-Dateien des Systems realisiert. Jede Controller-Klasse kann eine oder mehrere Instanzen der Template-Klasse oder einer ihrer Nachfahren besitzen und diese kontrollieren.

Dazu kann eine Controller-Klasse einer Template-Instanz bestimmte Werte aus dem Datenmodell zuweisen. Die Template-Instanz zeigt diese dann abhängig von der in der jeweiligen Template-Datei definierten Darstellungslogik an.

Da die Klasse „Template“ in TYPOlight von der Klasse „Controller“ abgeleitet wird, können innerhalb einer Template-Datei alle Methoden der Elternklasse, die als „public“ oder „protected“ deklariert wurden, aufgerufen werden. Dies kann z.B. bei der Darstellung von Bildern hilfreich sein. So kann die Methode „getImage“ des Controllers aufgerufen werden, um ein Bild immer mit einer festen Breite oder Höhe darzustellen, unabhängig von den Abmessungen der Ursprungsdatei.

Ein Beispiel für eine Template-Datei ist im Kapitel 6.8 zu finden.

5.2 Komponenten im Frontend

Dieses Kapitel beschreibt alle Komponenten von TYPOlight, die im Frontend zur Darstellung von Inhalten oder zur Realisierung bestimmter Funktionalitäten genutzt werden können. Dazu gehören Module und Inhaltselemente sowie eine Kombination aus beiden, also hybride Komponenten.

Als Ausgangspunkt dient für alle Typen die Basisklasse „Frontend“, von der alle weiteren Komponenten abgeleitet werden. Diese stellt bestimmte Basisfunktionalitäten bereit, die von allen Komponenten im Frontend genutzt werden können. Dazu gehören u.a. die Ermittlung von Seiten-IDs anhand der URL, die Behandlung von Fehlern bei nicht existierenden oder umgezogenen Seiten, die Weiterleitung auf andere Seiten sowie die Generierung von Abständen (Margin) zu vorhergehenden oder nachfolgenden Komponenten.

Module

Module erfüllen immer eine bestimmte Funktionalität, dienen also weniger zur Darstellung von Inhalten. Dazu können z.B. die Darstellung einer Seitennavigation, die Auflistung tabellarischer Daten aus der Datenbank oder die Realisierung einer Suchfunktion für die Website gehören.

Inhaltselemente

Inhaltselemente dienen überwiegend zur Darstellung statischer Inhalte. Dies können Texte, Überschriften, Listen, Tabellen Bilder und vieles mehr sein.

Hybride Komponenten

Hybride Komponenten besitzen sowohl Eigenschaften von Modulen als auch von Inhaltselementen, daher existiert für diesen Typ die Basisklasse „Hybrid“. Einzige Komponente dieses Typs in TYPOlight ist derzeit noch die Klasse „Form“, welche zur Darstellung und Auswertung von HTML-Formularen genutzt werden kann.

6 Tutorial: Programmierung einer Extension für TYPOlight

In diesem Kapitel wird Schritt für Schritt die Programmierung eines neuen Moduls für TYPOlight erläutert. Als Beispiel soll dazu die Entwicklung eines Gästebuch-Moduls dienen, da TYPOlight ein solches Modul in der Standardinstallation noch nicht anbietet. Dazu wird nach der Konzeption zunächst das Anlegen aller notwendigen Dateien und Verzeichnisse beschrieben. Anschließend wird der Aufbau des Datenmodells mit Hilfe des globalen Data Container Arrays erläutert. Zuletzt werden die Implementierung aller notwendigen Klassen sowie die Erstellung von Templates erklärt.

6.1 Konzeption der Extension

Das Gästebuch soll Besuchern einer Website die Möglichkeit geben, neue Einträge zu verfassen sowie vorhandene Einträge zu lesen. Es werden also zwei Frontend-Module benötigt, ein Modul zum Anzeigen vorhandener Einträge und ein Modul zum Verfassen neuer Einträge. Außerdem sollen von Administratoren der Website beliebig viele Gästebücher angelegt und im Frontend eingebunden werden können. Bei einem Gästebuch soll nur der Name gespeichert werden, bei einem Eintrag Name, E-Mail-Adresse, Homepage und IP-Adresse des Autors, Datum des Eintrags sowie der eigentliche Text. Zusätzlich sollen Administratoren die Möglichkeit haben, neue Gästebucheinträge erst nach einer Überprüfung freischalten zu können, so dass diese erst dann öffentlich sichtbar sind. Im Rahmen dieses Beispiels soll zunächst nur ein englisches Sprachpaket erstellt werden, das Erstellen weiterer Pakete erfolgt analog dazu.

Für die geplante Extension werden also zwei Klassen benötigt: Die Klasse „ModuleGuestbook“ zum Anzeigen von Gästebucheinträgen sowie die Klasse „ModuleGuestbookForm“ zum Verfassen von neuen Einträgen. Außerdem werden zwei neue Datenbanktabellen benötigt, eine zum Speichern der Gästebuchinformationen und eine zum Speichern der jeweiligen Gästebuch-Einträge.

6.2 Anlegen der Datei- und Verzeichnisstruktur

Sobald das Konzept der neuen Extension aufgestellt ist, also sobald alle benötigten Klassen definiert wurden, können alle notwendigen Verzeichnisse und Dateien angelegt werden. Dies kann manuell erledigt werden, komfortabler ist allerdings die Nutzung des im Backend integrierten „Modul Creators“. Dieser ermöglicht die automatische Erstellung aller benötigten Verzeichnisse und Dateien, sobald Angaben zum Namen und Paket des neuen Moduls, zu dessen Autor und Lizenz sowie zu allen benötigten Klassen, Datenbank-Tabellen und Template-Dateien gemacht wurden.

Folgende Verzeichnisse werden also vom Modul Creator im Verzeichnis „system/modules/“ der TYPOlight-Installation erstellt bzw. müssen manuell angelegt werden:

- Hauptverzeichnis „guestbook“ (enthält alle weiteren Verzeichnisse sowie die PHP-Dateien zu allen benötigten Klassen)
- Verzeichnis „config“ (enthält PHP-Datei mit Konfiguration des Moduls über dessen Einbindung im Frontend und Backend sowie SQL-Datei mit benötigter Datenbankstruktur)
- Verzeichnis „dca“ (enthält jeweils eine PHP-Datei mit dem Data-Container-Array zu jeder vom Modul benötigten Datenbank-Tabelle)
- Verzeichnis „languages“ (enthält alle vom Modul benötigten Sprachdateien)
- Verzeichnis „templates“ (enthält alle vom Modul benötigten Template-Dateien)

In Abbildung 6-1 ist die Verzeichnisstruktur noch einmal schematisch dargestellt:

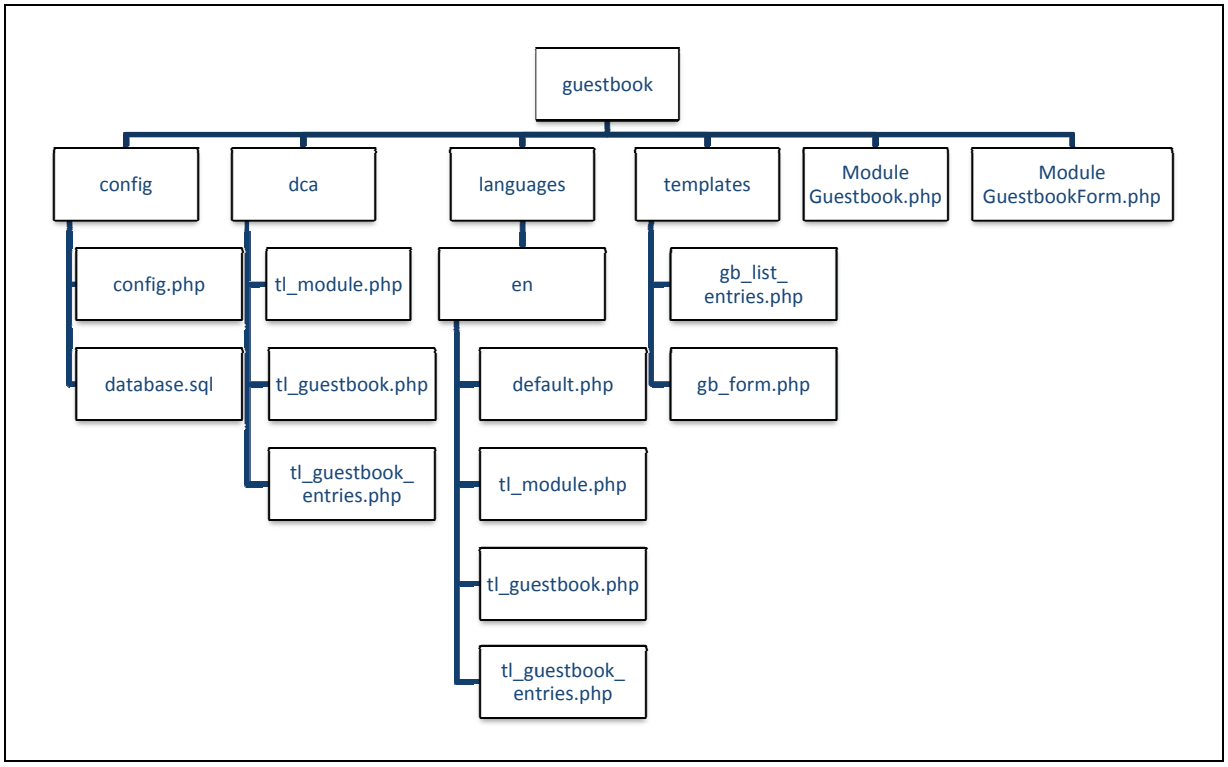

Abbildung 6-1: Datei- und Verzeichnisstruktur des Beispiel-Moduls „guestbook"

6.3 Anpassung der Datenbank

Da das Gästebuch zwei neue Datenbanktabellen benötigt, muss die bestehende Datenbank um diese erweitert werden. Außerdem werden in der Tabelle „tl_module“ neue Felder für die Konfiguration der beiden Gästebuch-Module benötigt. Alle Änderungen könnten manuell per Eingabe der entsprechenden SQL-Befehle oder über eine Verwaltungsoberfläche wie phpMyAdmin erledigt werden, der bessere Weg ist jedoch das Anlegen einer SQL-Datei im Verzeichnis „config“. Denn diese Datei wird

später benötigt um die Extension in einem bestehenden TYPOlight-System mit Hilfe des Installations-Skriptes zu installieren.

Für die Gästebücher wird die Tabelle „tl_guestbook“ über folgenden SQL-Befehl in der Datei „database.sql“ angelegt:

```
01 CREATE TABLE `tl_guestbook` (
02 `id` int(10) unsigned NOT NULL auto_increment,
03 `tstamp` int(10) unsigned NOT NULL default '0',
04 `title` varchar(255) NOT NULL default '',
05 PRIMARY KEY  (`id`)
06 ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Die Felder „id“ und „tstamp“ werden zwingend von TYPOlight benötigt und daher automatisch vom Module Creator in den SQL-Befehl eingefügt. Dieser wurde nun nur noch um die Zeile 4 für das Titelfeld des Gästebuchs ergänzt.

Analog dazu wird die Tabelle „tl_guestbook_entries“ für die Speicherung der Gästebucheinträge erstellt:

```
07 CREATE TABLE `tl_guestbook_entries` (
08 `id` int(10) unsigned NOT NULL auto_increment,
09 `pid` int(10) unsigned NOT NULL default '0',
10 `tstamp` int(10) unsigned NOT NULL default '0',
11 `name` varchar(64) NOT NULL default '',
12 `email` varchar(128) NOT NULL default '',
13 `website` varchar(128) NOT NULL default '',
14 `comment` text NULL,
15 `ip` varchar(15) NOT NULL default '',
16 `date` int(10) unsigned NOT NULL default '0',
17 `published` char(1) NOT NULL default '',
18 PRIMARY KEY  (`id`),
19 KEY `pid` (`pid`)
20 ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Hier wird neben den Feldern für die Daten des Gästebucheintrags außerdem in Zeile drei ein Feld „pid“ erstellt, welches von TYPOlight standardmäßig verwendet wird, um bei einem Gästebucheintrag eine Beziehung zum übergeordneten Gästebuch herzustellen.

Für die Erweiterung der Tabelle „tl_module“ wird ebenfalls ein SQL-Befehl erstellt:

```
21 CREATE TABLE `tl_module` (
22 `gb_guestbooks` blob NULL,
23 `gb_template` varchar(64) NOT NULL default '',
```

```
24  `gb_moderate` char(1) NOT NULL default '',
25  `gb_perPage` smallint(5) unsigned NOT NULL default '0',
26  ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Zu beachten ist hier, dass die bereits vorhandene Tabelle ebenfalls über den SQL-Befehl „CREATE TABLE“ zu modifizieren ist, jedoch nicht über den Befehl „ALTER TABLE“. Ansonsten kann das Installationsskript von TYPOlight die Struktur aller bereits existierenden Datenbank-Tabellen nicht korrekt mit den SQL-Dateien der einzelnen Module abgleichen. Außerdem ist dafür eine korrekte Groß- und Kleinschreibung aller SQL-Befehle wie in den eben gezeigten Beispielen notwendig.

6.4 Konfiguration von TYPOlight

Damit TYPOlight die beiden neuen Module ordnungsgemäß ins System einbinden kann, wird eine Datei „config.php“ im Verzeichnis „config“ der Extension benötigt.

Diese Datei enthält alle notwendigen Informationen über neue Backend- und Frontend-Module, Inhaltselemente, Formularfelder und Seitentypen, welche durch die Extension ins System eingebracht werden sollen.

Folgender PHP-Code dient zur Einbindung eines neuen Moduls im Backend von TYPOlight:

```
01  $GLOBALS['BE_MOD']['content']['guestbook'] = array
02  (
03    'tables' => array('tl_guestbook','tl_guestbook_entries'),
04    'icon' => 'system/modules/guestbook/html/icon.gif'
05  );
```

Im Konfigurations-Array der Backend-Module (Zeile 1, Schlüssel „BE_MOD“) wird im Inhaltsbereich (Zeile 1, Schlüssel „content“) ein neues Array unter dem Schlüssel „guestbook“ eingefügt. Dieses enthält zwei Elemente, welche alle benötigten Datenbank-Tabellen (Zeile 3, Schlüssel „tables“) und das Icon für das Backend-Modul (Zeile 4, Schlüssel „icon“) zuweisen.

Mit diesem PHP-Code werden die beiden neuen Module als Frontend-Module verfügbar gemacht:

```
06  $GLOBALS['FE_MOD'] = array
07  (
08    'guestbook' => array
09    (
10      'guestbook' => 'ModuleGuestbook',
```

```
11      'guestbook_form' => 'ModuleGuestbookForm'
12    )
13  );
```

Dazu wird in das Konfigurations-Array der Frontend-Module (Zeile 6, Schlüssel „FE_MOD“) zunächst eine neue Gruppe von Modulen unter dem Schlüssel „guestbook“ in Form eines Arrays angelegt (Zeile 8). In dieses Array werden unter jeweils einem bestimmten Schlüssel die Namen der Klassen, welche die Funktionalität der Module im Frontend enthalten sollen, eingefügt. Die Schlüssel werden später beim Anlegen eines Moduls in der jeweiligen Konfiguration gespeichert, wodurch dem System bekannt gemacht wird, welche Klasse für ein bestimmtes Modul geladen werden soll.

6.5 Aufbau des Data Container Arrays

Für jede Datenbank-Tabelle, die von der Extension genutzt wird, muss ein Data Container Array existieren. Dieses wird in den jeweiligen PHP-Dateien im Verzeichnis „dca“ definiert, wie in Kapitel 6.2 bereits erwähnt. Eine vollständige Referenz über alle möglichen Schlüssel und Werte des Data Container Arrays ist auf der TYPOlight-Website[48] zu finden.

Für die Verwaltung der Gästebücher im Backend müssen die Data Container Arrays für die Tabellen „tl_guestbook“ und „tl_guestbook_entries“ aufgebaut werden. Außerdem muss das Data Container Array „tl_module“ erweitert werden, um verschiedene Konfigurationen für die neuen Module aufnehmen zu können. In jedem Array gibt es dafür zunächst vier Hauptbereiche, die unter den entsprechenden Schlüsseln weitere Arrays mit genaueren Konfigurationsmöglichkeiten beinhalten.

6.5.1 Data Container Array „tl_guestbook“

Schlüssel „config“

```
01  'config' => array
02  (
03    'dataContainer' => 'Table',
04    'ctable' => array('tl_guestbook_entries')
05  )
```

Im Schlüssel „config“ muss mindestens der verwendete Data Container (Zeile 3, Schlüssel „dataContainer“) angegeben werden. Im Gästebuch-Beispiel ist dies eine Datenbank-Tabelle, daher steht dort

[48] Zu finden unter http://typolight.org/table-configuration.html

auch der Wert „Table“. Außerdem wird in diesem Fall in Zeile 4 die Tabelle „tl_guestbook_entries“ unter dem Schlüssel „ctable“ als Kindtabelle in Beziehung zur Tabelle „tl_guestbooks“ gesetzt.

Schlüssel „list“

```
06 'list' => array
07 (
08   'sorting' => array
09   (
10     'mode'                    => 1,
11     'fields'                  => array('title'),
12     'flag'                    => 1,
13     'panelLayout'             => 'search,limit'
14   ),
15   'label' => array
16   (
17     'fields'                  => array('title'),
18     'format'                  => '%s'
19   ),
20   'global_operations' => array
21   (
22     'all' => array
23     (
24       'label'               => &$GLOBALS['TL_LANG']['MSC']['all'],
25       'href'                => 'act=select',
26       'class'               => 'header_edit_all',
27       'attributes'          => 'onclick="Backend.getScrollOffset();"'
28     )
29   ),
30   'operations' => array
31   (
32     'edit' => array
33     (
34       'label'         => &$GLOBALS['TL_LANG']['tl_guestbook']['edit'],
35       'href'          => 'table=tl_guestbook_entries',
36       'icon'          => 'edit.gif'
37     )
38   )
39 )
```

Im Schlüssel „list“ werden alle Einstellungen vorgenommen, wie die Datensätze der Tabelle im Backend aufgelistet werden sollen. Im Unterschlüssel „label“ (Zeilen 15-19) wird festgelegt, welche Felder der Tabelle in der Auflistung als Stellvertreter für den ganzen Datensatz angezeigt werden. Die Konfiguration unter „sorting“ (Zeilen 8-14) legt bestimmte Felder fest, nach denen die Datensätze im Backend sortiert werden können. Im Schlüssel „global_operations“ (Zeile 20-29) werden alle Aktionen definiert, die nicht auf einen bestimmten Datensatz angewendet werden können, sondern für

die ganze Tabelle gültig sind. Das kann z.B. das gleichzeitige Bearbeiten mehrerer Datensätze sein, was in diesem Beispiel durch den Schlüssel „all“ definiert wird. Unter dem Schlüssel „operations“ (Zeile 30-38) werden alle Aktionen definiert, die für jeden einzelnen Datensatz angewendet werden können. In dieses Beispiel wurde nur exemplarisch die Aktion zum Bearbeiten eines Datensatzes unter dem Schlüssel „edit“ aufgenommen, eine vollständige Version des Data Container Arrays ist im Anhang B-1 zu finden.

Schlüssel „fields“

```
40  'fields' => array
41  (
42    'title' => array
43    (
44      'label'               => &$GLOBALS['TL_LANG']['tl_guestbook']['title'],
45      'exclude'             => true,
46      'search'              => true,
47      'inputType'           => 'text',
48      'eval'                => array('mandatory'=>true, 'maxlength'=>255)
49    )
50  )
```

Unter dem Schlüssel „fields“ werden alle Felder der Datenbank-Tabelle konfiguriert. In diesem Beispiel wird allerdings nur exemplarisch das Feld „title“ dargestellt, eine vollständige Version ist wieder im Anhang B-1 zu finden. Wichtig sind hier vor allem die Schlüssel „label“ und „inputType“. Unter „label“ wird eine Referenz auf das globale Sprachen-Array eingetragen, welche eine Bezeichnung des Datenfeldes in der jeweils verwendeten Sprache enthält. Der Schlüssel „inputType“ legt fest, welche Art von Formularfeld im Backend für das jeweilige Datenfeld beim Einfügen neuer oder beim Bearbeiten vorhandener Datensätze verwendet werden soll.

Schlüssel „palettes“

```
51  'palettes' => array
52  (
53    'default' => 'title'
54  ),
```

Unter dem Schlüssel „palettes“ können verschiedene Konfigurationen für die Anzeige von Formularfeldern beim Bearbeiten eines bestimmten Datensatzes angelegt werden. Im Gästebuch-Beispiel wird nur eine Standard-Konfiguration unter dem Schlüssel „default“ benötigt, welche lediglich die Anzeige des Formularfelds für den Titel des Gästebuchs realisiert.

6.5.2 Data Container Array „tl_guestbook_entries“

Der Aufbau des zweiten Data Container Arrays „tl_gustbook_entries“ erfolgt im Wesentlichen analog zum Aufbau des ersten Data Container Arrays. Daher soll dies hier nicht noch einmal im Detail erklärt werden, eine vollständige Version ist wieder im Anhang B-1 zu finden.

Bei diesem Data Container Array besteht allerdings ein Unterschied bei der Konfiguration im Schlüssel „config“:

```
01  'config' => array
02  (
03    'dataContainer'               => 'Table',
04    'ptable'                      => 'tl_guestbook'
05  )
```

Statt dem Unterschlüssel „ctable“ wird nun der Schlüssel „ptable“ definiert, welcher die Tabelle „tl_guestbook“ der Tabelle „tl_guestbook_entries“ überordnet.

Ein weiterer wesentlicher Unterschied besteht im Schlüssel „sorting“:

```
06  'sorting' => array
07  (
08    'mode'                    => 4,
09    'fields'                  => array('name'),
10    'headerFields'            => array('title'),
11    'panelLayout'             => 'search,limit',
12    'child_record_callback'   =>
       array('tl_guestbook_entries','listGuestbookEntries')
13  )
```

Hier wird der Unterschlüssel „mode“ auf den Wert 4 gesetzt. Dadurch wird über der Auflistung der Datensätze der Tabelle „tl_guestbook_entries“ außerdem der jeweils übergeordnete Datensatz aus der Tabelle „tl_guestbook“ dargestellt. Weiterhin wird in diesem Modus eine Callback-Funktion benötigt (Zeile 4, Schlüssel „child_record_callback“), welche den HTML-Code zur Darstellung der einzelnen Datensätze aus der untergeordneten Tabelle zurückgibt.

6.5.3 Data Container Array „tl_module“

Der Data Container „tl_module“ dient zur Aufnahme von jeweils einer Konfiguration eines bestimmten Frontend-Moduls, nachdem dieses im Backend angelegt wurde. Für das Gästebuch-Beispiel werden einige neue Konfigurationsfelder benötigt.

Zunächst muss in der Konfiguration eines Gästebuch-Moduls gespeichert werden, welches Gästebuch aus der Tabelle „tl_guestbook“ zur Anzeige von Gästebucheinträgen genutzt werden soll bzw. in welches Gästebuch neue Einträge eingefügt werden sollen. Dazu dient das Feld „gb_guestbooks“ aus der Tabelle „tl_module“. Weiterhin wird für jedes Gästebuch-Modul das Feld „gb_template“ benötigt, in welchem eine Template-Datei für die Darstellung der Einträge im Frontend angegeben werden kann. Das Feld „gb_perPage“ dient zur Einstellung der Anzahl der Einträge, die auf einer Gästebuchseite dargestellt werden, bevor zur nächsten Seite geblättert werden muss. Außerdem wird das Feld „gb_moderate“ benötigt, durch welches für jedes Gästebuch-Modul festgelegt werden kann, ob neue Einträge sofort im Frontend sichtbar sein sollen oder erst durch einen Administrator freigeschaltet werden müssen.

Die Konfiguration des Data Container Arrays erfolgt analog zu den beiden vorhergehenden Arrays, eine vollständige Version ist wieder im Anhang B-1 zu finden.

6.6 Implementierung der benötigten Klassen

Die PHP-Dateien für die benötigten Modul-Klassen wurden durch den Modul-Creator bzw. manuell bereits angelegt.[49] Nun wird innerhalb dieser Klassen die gewünschte Funktionalität zur Realisierung eines Gästebuchs im Frontend implementiert.

Der Modul-Creator legt in den jeweiligen Klassen automatisch eine leere Methode namens „compile“ an. Diese Methode wird von TYPOlight bei der Anzeige eines Moduls oder Inhaltselements im Frontend automatisch aufgerufen. Daher müssen dort die Hauptfunktionalitäten des jeweiligen Moduls implementiert werden. Bei umfangreichen Modulen können natürlich auch weitere Methoden angelegt werden, welche dann aus der Methode „compile“ heraus aufgerufen werden können.

6.6.1 Klasse „ModuleGuestbookForm“

Die Klasse „ModuleGuestbookForm“ dient zum Verfassen von neuen Einträgen und dessen Speicherung in der Datenbank. Dazu muss sie ein entsprechendes HTML-Formular im Frontend darstellen, eingegebene Benutzerdaten auf Gültigkeit überprüfen und bei Korrektheit anschließend in der Datenbank abspeichern.

Zur Darstellung des Formulars bieten sich in TYPOlight sogenannte „Widgets“ an. Diese repräsentieren jeweils einen bestimmten Typ von Benutzersteuerelementen (einfache Formularfelder oder auch

[49] Siehe auch Kapitel 6.2 - Anlegen der Datei- und Verzeichnisstruktur

erweiterte Steuerelemente) und können über assoziative Arrays konfiguriert werden. Dadurch kann jedes Widget auch mit einer Validierungsfunktion versehen werden, welche die Benutzereingaben entsprechend den vordefinierten Regeln im Konfigurations-Array auf Gültigkeit überprüft.

Für das Formular zum Verfassen neuer Gästebucheinträge kann das Data Container Array der Tabelle „tl_guestbook_entries“ verwendet werden. Dazu werden alle Felder des Arrays in einer For-Schleife durchlaufen und diejenigen, welche die Eigenschaft „feEditable“ im Unterschlüssel „eval“ besitzen, werden als Konfigurations-Array für ein neues Widget-Objekt verwendet.

```
01 // Run through all fields
02 foreach($GLOBALS['TL_DCA']['tl_guestbook_entries']['fields'] as $strField =>
   $arrField)
03 {
04   if(!$arrField['eval']['feEditable'])
05   {
06     continue;
07   }
08
09   // Create new Widget
10   $strClass = $GLOBALS['TL_FFL'][$arrField['inputType']];
11   $strFile = sprintf('%s/system/modules/frontend/%s.php', TL_ROOT, $strClass);
12
13   if (!file_exists($strFile))
14   {
15     continue;
16   }
17   $arrField['eval']['required'] = $arrField['eval']['mandatory'];
18   $objWidget = new $strClass($this->prepareForWidget($arrField, $strField));
```

In diesem Codebeispiel wird der Wert im Schlüssel „inputType“ des jeweiligen Feldes dazu verwendet, die entsprechende Widget-Klasse zu bestimmen. Dann wird ein neues Widget-Objekt mit Hilfe des Konfigurations-Arrays des jeweiligen Feldes erzeugt. Dabei muss das Array über die Methode „prepareForWidget“ des Controllers zunächst so konvertiert werden, dass es vom Konstruktor der Widget-Klasse verarbeitet werden kann.

```
19   // Validate widget
20   if ($this->Input->post('FORM_SUBMIT') == 'tl_guestbook')
21   {
22     $objWidget->validate();
23
24     if ($objWidget->hasErrors())
25     {
26       $doNotSubmit = true;
```

```
27        }
28    }
29
30    $arrWidgets[] = $objWidget;
31  }
```

Falls beim Kompilieren des Moduls bereits HTTP-POST-Daten vom Benutzer übertragen wurden, werden im weiteren Verlauf der For-Schleife alle Benutzereingaben durch den Aufruf der Methode „validate“ des jeweiligen Widgets auf Gültigkeit überprüft. Wenn dabei keine Unzulänglichkeiten festgestellt werden, können die übertragenen Daten im weiteren Programmablauf über eine entsprechende Methode in der Datenbank abgespeichert werden.

Der vollständige Quellcode der Klasse ist wieder im Anhang B-1 zu finden.

6.6.2 Klasse „ModuleGuestbook“

Die Klasse „ModuleGuestbook“ dient zum Auflisten aller Einträge eines bestimmten Gästebuchs. Dazu wird zunächst die Gesamtzahl aller Einträge bestimmt und ggf. ein Seitenwechsel erzeugt:

```
01  // Pagination
02  if ($this->gb_perPage > 0)
03  {
04    $page = $this->Input->get('page') ? $this->Input->get('page') : 1;
05    $limit = $this->gb_perPage;
06    $offset = ($page - 1) * $this->gb_perPage;
07
08    // Get total number of entries
09    $objTotal = $this->Database->prepare("SELECT COUNT(*) AS count FROM
    tl_guestbook_entries WHERE pid=?" . (!BE_USER_LOGGED_IN ? " AND published=?" :
    ""))
10                       ->execute($this->gb_guestbook, 1);
11    // Add pagination menu
12    $objPagination = new Pagination($objTotal->count, $this->gb_perPage);
13    $this->Template->pagination = $objPagination->generate("\n  ");
14  }
```

Für die Realisierung der Blätterfunktion nach einer bestimmten Anzahl von Gästebucheinträgen kann hier die Pagination-Klasse von TYPOlight verwendet werden.

Anschließend erfolgt das Auslesen jedes einzelnen Eintrags aus der Datenbank:

```
15 // Get all published entries
16 $objEntriesStmt = $this->Database->prepare("SELECT * FROM tl_guestbook_entries
   WHERE pid=?" . (!BE_USER_LOGGED_IN ? " AND published=?" : "") . " ORDER BY date
   DESC");
17
18 if ($limit)
19 {
20   $objEntriesStmt->limit($limit, $offset);
21 }
22
23 $objEntries = $objEntriesStmt->execute($this->gb_guestbook, 1);
24
25 if ($objEntries->numRows)
26 {
27   $count = 0;
28   $objTemplate = new FrontendTemplate('entry_default');
29
30   while ($objEntries->next())
31   {
32      $objTemplate->name = $objEntries->name;
33      // [...] weitere Template-Variablen zuweisen
34      $arrEntries[] = $objTemplate->parse();
35   }
36 }
```

Die Datenbankabfragen können über die Datenbank-Abstraktionsklasse „Database“ von TYPOlight und deren Methoden „prepare“, „limit“ und „execute“ sehr komfortabel durchgeführt werden. Eine genaue Dokumentation dazu ist auf der TYPOlight-Website[50] zu finden.

Weiterhin wird für jeden Gästebucheintrag ein neues Objekt der Klasse „FrontendTemplate“ erzeugt, welchem dann jeweils die Daten eines Eintrags zugewiesen werden. Dieses Template-Objekt wird später dem Haupt-Template des Gästebuchs zugewiesen, so dass die Darstellung der einzelnen Gästebucheinträge unabhängig vom Rest des Gästebuchs erfolgen kann.

Der vollständige Quellcode der Klasse ist im Anhang B-1 zu finden.

[50] Zu finden unter http://www.typolight.org/api/Library/Database.html

6.7 Erstellen von Sprachdateien

Die Sprachdateien befinden sich alle im Modul-Verzeichnis „languages“ und dort im jeweiligen Unterverzeichnis, welches nach dem primären Subtag der Sprache entsprechend ISO-639 benannt werden muss (z.B. „de“ für Deutschland).

Dort sollte zunächst zu jedem vom Modul verwendeten Data Container eine PHP-Datei mit dem gleichen Namen angelegt werden. In dieser PHP-Datei können nun alle Felder des Containers unter entsprechenden Schlüsseln im globalen Sprachen-Array mit einer Beschriftung und einer kurzen Beschreibung versehen werden. Die Sprachdatei für den Data Container „tl_guestbook“ könnte beispielsweise so aussehen:

```
01 // Fields
02 $GLOBALS['TL_LANG']['tl_guestbook']['title'] = array('Title', 'Please enter a
   guestbook title.');
03
04 // Buttons
05 $GLOBALS['TL_LANG']['tl_guestbook']['new'] = array('New guestbook', 'Create a
   new guestbook');
06 $GLOBALS['TL_LANG']['tl_guestbook']['edit'] = array('Edit guestbook', 'Edit
   guestbook ID %s');
07 $GLOBALS['TL_LANG']['tl_guestbook']['copy'] = array('Copy guestbook', 'Copy
   guestbook ID %s');
08 $GLOBALS['TL_LANG']['tl_guestbook']['delete'] = array('Delete guestbook',
   'Delete guestbook ID %s');
09 $GLOBALS['TL_LANG']['tl_guestbook']['show'] = array('Guestbook details', 'Show
   details of guestbook ID %s');
```

Weiterhin können im Sprachverzeichnis noch die Dateien „module.php“ und „default.php“ angelegt werden. In der Datei „module.php“ können ein Titel und eine Beschreibung für das neue Modul hinterlegt werden, welche im Backend auf der Startseite bzw. in der Navigation angezeigt werden. Die Datei „default.php“ dient zur Aufnahme beliebiger Beschriftungen, die sowohl von Frontend- als auch von Backend-Modulen verwendet werden können.

6.8 Erstellen von Templates

Alle Template-Dateien in TYPOlight besitzen die Erweiterung „tpl“ und beinhalten eine Mischung aus HTML und PHP-Code. Die von der neuen Extension benötigten Template-Dateien wurden vom Modul-Creator im Verzeichnis „templates“ erstellt bzw. manuell angelegt und müssen nun mit Darstellungslogik versehen werden.

Dazu können mit Hilfe von HTML zunächst beliebige Layout-Bereiche und Markups erstellt werden. Diese können bei Bedarf mit Hilfe aller in PHP gebräuchlichen Kontrollstrukturen wiederholt oder nur

bedingt angezeigt werden. Außerdem können alle Template-Variablen, die innerhalb eines Moduls zugewiesen wurden, ausgegeben werden. Da die Template-Datei innerhalb der Klasse „Template“ geparst wird, kann auf alle Template-Variablen über den this-Pointer und den nachfolgenden Variablennamen zugegriffen werden. Das Template für einen einzelnen Gästebucheintrag könnte also beispielsweise so aussehen:

```
01 <div class="entry_default" id="<?php echo $this->id; ?>">
02     <div class="info">
03         <?php echo $this->by; ?>
04         <?php if ($this->website): ?>
05             <a href="<?php echo $this->website; ?>">
06         <?php endif; ?>
07         <?php echo $this->name; ?>
08         <?php if ($this->website): ?>
09             </a>
10         <?php endif; ?>
11         <span class="date"> | <?php echo $this->datim; ?></span>
12     </div>
13     <div class="comment">
14         <?php echo $this->comment; ?>
15     </div>
16 </div>
```

Alle für die Gästebuch-Extension benötigten Templates befinden sich wieder im Anhang B-1.

Wenn alle Schritte abgeschlossen sind, kann die neue Extension im Backend von TYPOlight in einen Artikel oder ein Seitenlayout eingebunden und anschließend im Frontend getestet werden.

7 Konzeption, Implementierung und Test der Extensions für TYPOlight

Für die Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. werden bestimmte Funktionen benötigt, die TYPOlight noch nicht zur Verfügung stellt. Daher werden in diesem Kapitel alle Extensions zur Realisierung der noch benötigten Funktionen konzipiert, implementiert und getestet. Dazu werden zunächst Use-Case-Diagramme in Verbindung mit einer allgemeinen Funktionsbeschreibung angelegt. Anschließend werden die benötigten Eingabemasken für die Extensions im Frontend spezifiziert und die daraus resultierenden Testfälle anhand von Formblättern beschrieben. Danach werden Klassen- und Sequenzdiagramme zur genaueren Beschreibung der Struktur sowie der Programmabläufe der Extensions angelegt. Zuletzt werden die fertigen Extensions anhand der vorher definierten Testfälle auf korrekte Funktion geprüft.

7.1 Use-Case Diagramme und Funktionsbeschreibungen

In diesem Kapitel werden alle Anforderungen an die TYPOlight-Extensions für die Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. spezifiziert. Diese werden außerdem in jeweils einem Use-Case-Diagramm grafisch dargestellt.

7.1.1 Extension „Mitglieds-Avatar“

TYPOlight bietet für Frontend-Mitglieder in der Standardinstallation keine Möglichkeit zum Hochladen von Profilfotos bzw. Firmenlogos. Es kann zwar über den Formulargenerator ein HTML-Formular mit einem Dateiupload-Feld erstellt werden, allerdings ist dann das spätere Entfernen der Grafik durch das Mitglied nicht mehr möglich. Außerdem besteht so keine Möglichkeit, hochgeladene Grafiken einem bestimmten Mitglied zuzuordnen. Dies kann zwar durch das Speichern der Grafik im jeweiligen Mitgliedsverzeichnis realisiert werden, allerdings kann dann bei einer großen Mitgliederzahl eine unübersichtliche Verzeichnisstruktur auf dem Server entstehen.

Daher soll hier die Konzeption für eine TYPOlight-Extension aufgestellt werden, welche das Verwalten von Avataren (Profilfotos, Firmenlogos o.ä.) durch Frontend-Mitglieder erlaubt.

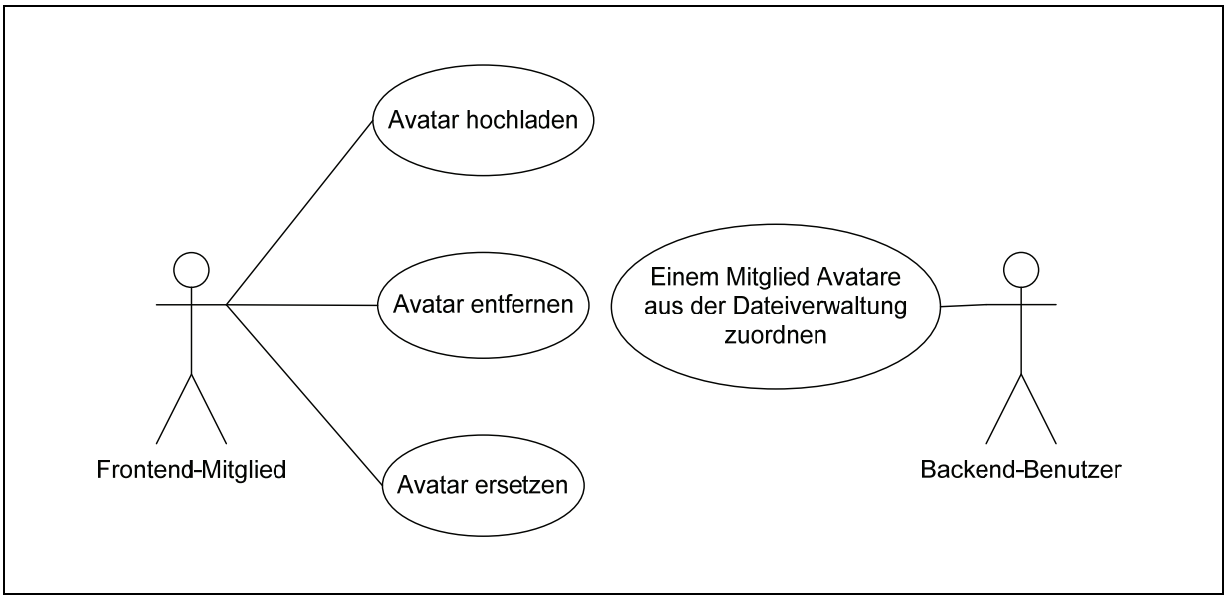

Abbildung 7-1: Use-Case-Diagramm für Extension „Mitglieds-Avatar“

Wie in Abbildung 7-1 dargestellt, soll die Extension zum Verwalten von Mitglieds-Avataren Frontend-Mitgliedern die Möglichkeit geben, ein oder mehrere Firmenlogos oder Profilfotos auf den Webserver zu laden, welche anschließend in Verbindung mit ihrem Benutzerprofil gespeichert werden. Weiterhin sollen bereits hochgeladene Avatare wieder entfernt werden oder durch andere ersetzt werden können.

Backend-Benutzer, die Zugang zur Mitgliederverwaltung in TYPOlight besitzen, sollen jedem Mitglied beliebig viele Avatare aus der Dateiverwaltung von TYPOlight zuordnen oder diese wieder entfernen können.

7.1.2 Extension „Rotierender Avatar“

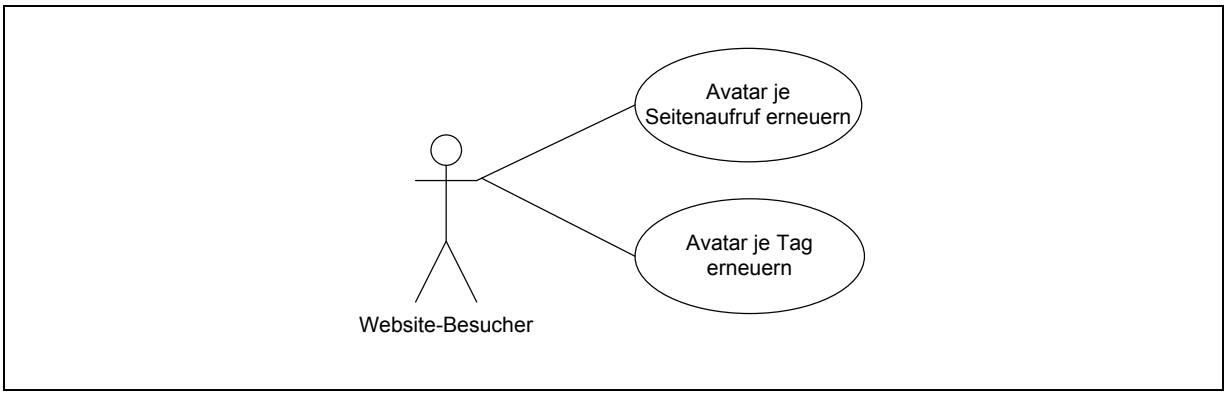

Abbildung 7-2: Use-Case-Diagramm für Extension „Rotierender Avatar“

Abbildung 7-2 zeigt das Use-Case-Diagramm für die Extension „Rotierender Avatar“. Diese soll das Anzeigen von Mitglieds-Avataren im Frontend ermöglichen, wobei bereits angezeigte Avatare erst wieder dargestellt werden, wenn alle anderen Avatare angezeigt wurden, also im rotierenden Prinzip. Dabei soll in der Modulkonfiguration festgelegt werden können, wie viele Avatare auf einmal angezeigt werden und ob die ausgewählten Avatare täglich oder bei jedem Seitenaufruf gewechselt werden. Außerdem sollen optional weitere Datenfelder angegeben werden können, die zusätzlich zum Avatar des jeweiligen Mitglieds angezeigt werden.

7.1.3 Extension „Erweiterte Auflistung“

TYPOlight bietet zwar schon ein Modul zur Auflistung von Daten aus einer Datenbanktabelle, dieses besitzt jedoch nur eingeschränkte Funktionalität. Zum einen berücksichtigt dieses Modul keine im Data Container Array definierten Fremdschlüssel, zum anderen können die Datensätze nicht nach bestimmten Werten gefiltert werden.

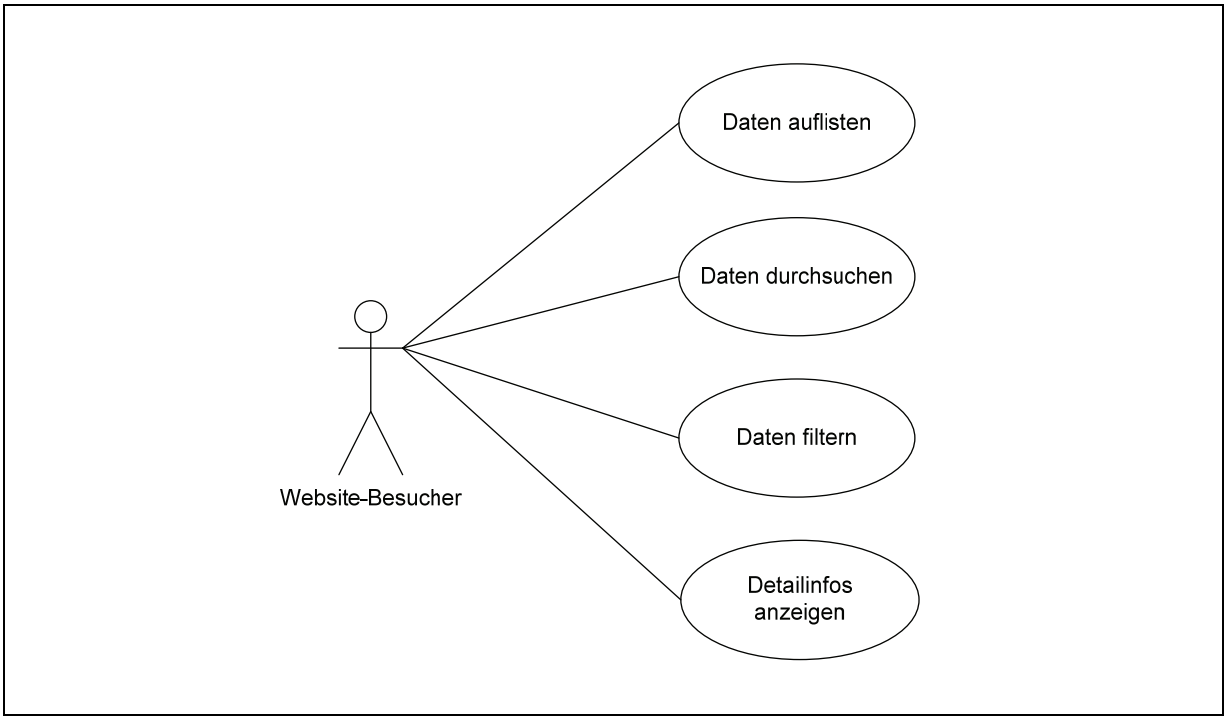

Abbildung 7-3: Use-Case-Diagramm für Extension „Erweiterte Auflistung“

Das Modul „Erweiterte Auflistung“ soll nun diese fehlenden Funktionen ermöglichen, wie in Abbildung 7-3 zu sehen ist. Dazu soll bei der Auflistung von Daten oder bei der Anzeige von Detailinfos für jedes Datenfeld anhand des Data Container Arrays überprüft werden, ob es sich dabei um einen normalen Wert oder um einen Fremdschlüssel für eine andere Datenbanktabelle handelt. Ist letzteres der Fall, so soll nicht der Fremdschlüssel, sondern der Wert aus der entsprechenden Tabel-

le, auf welchen dieser verweist, angezeigt werden. Außerdem soll die Extension die Filterung nach bestimmten Datenfeldern ermöglichen, so dass nur Datensätze angezeigt werden, die einen bestimmten Wert im entsprechenden Feld enthalten.

7.1.4 Extension „VdIV Registrierung“

TYPOlight enthält bereits ein Modul, welches die Registrierung von Frontend-Mitgliedern auf einer Website ermöglicht. Dabei besteht die Möglichkeit, vom System eine Aktivierungsemail an die bei der Registrierung hinterlegte Emailadresse zur Verifizierung derselben senden zu lassen. Diese Funktionalität muss nun für die Website des Verbandes der Immobilienverwalter Mitteldeutschland leicht abgewandelt werden.

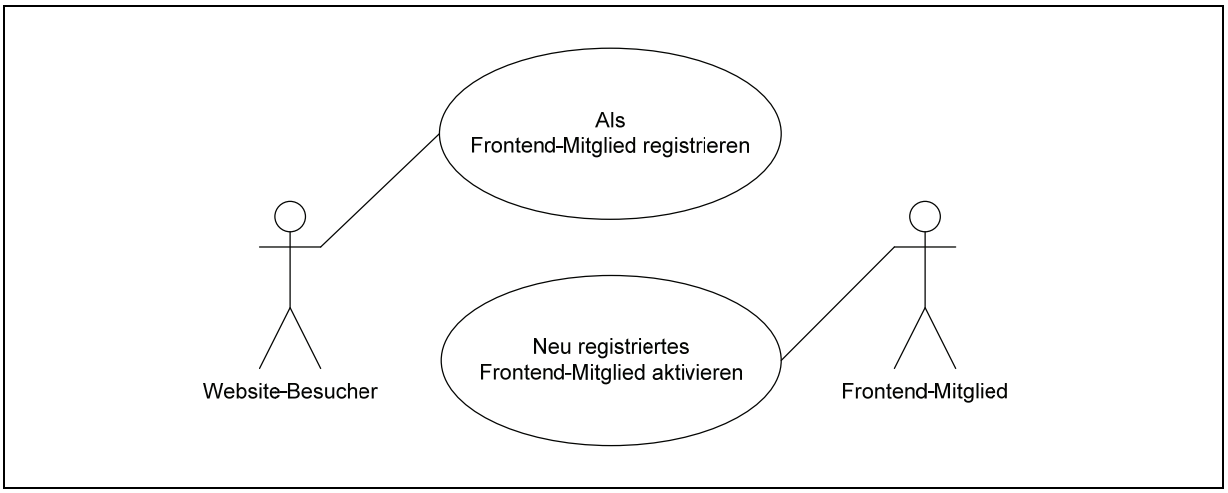

Abbildung 7-4: Use-Case-Diagramm für Extension „VdIV Registrierung“

Wie in Abbildung 7-4 zu sehen ist, soll hier die Aktivierung nicht durch das neue Mitglied selbst erfolgen, sondern durch ein bereits registriertes Mitglied, welches die Registrierung des neuen Mitglieds bestätigen soll. Ein Website-Besucher, der sich registrieren will, muss also beim Ausfüllen des Registrierungs-Formulars zusätzlich ein bereits registriertes Mitglied als Referenz aus einer Liste auswählen, welches anschließend eine Email mit einem Aktivierungslink zugestellt bekommt. Über diesen Aktivierungslink kann die Registrierung des neuen Mitglieds nun bestätigt werden.

Hintergrund dieser Funktionsweise ist die Sicherstellung der Registrierung von ausschließlich kompetenten Auftragnehmern bei der Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. So sollen Immobilienverwalter nur die Auftragnehmer aktivieren, deren Dienste sie bereits in Anspruch genommen und dabei positive Erfahrungen gemacht haben. So wird im gewissen Maße ein automatisiertes Qualitätsmanagement bei der Registrierung von Auftragnehmern erreicht.

7.1.5 Extension „Mitglieder-Kontaktformular“

TYPOlight bietet die Möglichkeit, beliebige Formulare zu erstellen, deren Inhalt nach dem Ausfüllen an eine fest definierte Emailadresse gesendet werden kann.

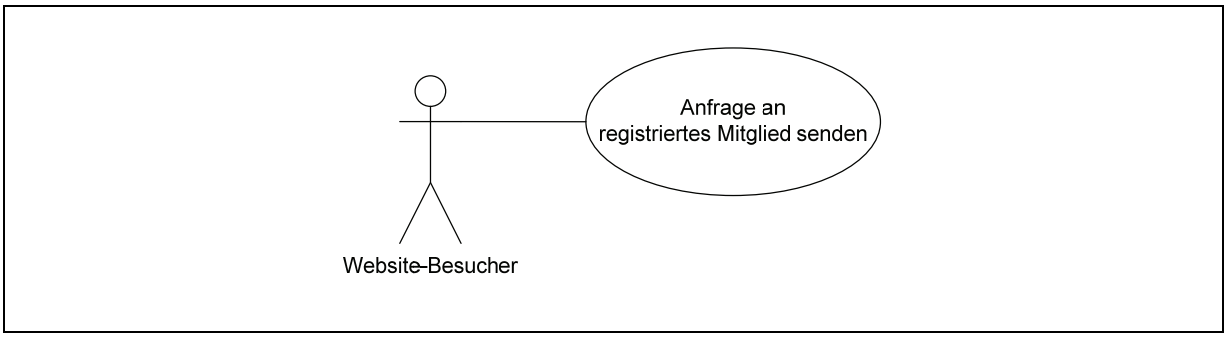

Abbildung 7-5: Use-Case-Diagramm für Extension „Mitglieder-Kontaktformular“

Durch die Extension „Mitglieder-Kontaktformular“ soll das System nun um die Möglichkeit erweitert werden, den Inhalt eines beliebigen Formulars an die Emailadresse eines registrierten Frontend-Mitglieds zu versenden, wie Abbildung 7-5 zeigt. Die Auswahl des Mitglieds erfolgt dabei durch die Übergabe der Mitglieds-ID als GET-Variable.

7.2 Definition von Testfällen

Anhand der Anforderungen, die im letzten Kapitel beschrieben wurden, ergeben sich für alle Extensions bestimmte Eingabemasken mit verschiedenen Eingabefeldern. Durch diese können fehlerhafte Benutzereingaben übertragen werden und somit Störfälle verursacht werden. Somit lassen sich Testfälle definieren, welche zum einen alle Möglichkeiten ohne einen Störfall, zum anderen aber auch alle Störfälle abdecken. Durch diese Vorgehensweise ist eine testgetriebene Entwicklung der Extensions möglich, wodurch bereits im Vorfeld das Auftreten von Programmfehlern erkannt und minimiert werden kann.

Eine tabellarische Beschreibung der Eingabemasken und möglichen Störfälle aller Extensions sowie die daraus resultierenden Testfälle sind im Anhang 0 zu finden.

7.3 Klassendiagramm

Das nachfolgende Klassendiagramm in Abbildung 7-6 repräsentiert die statische Struktur aller Extensions, die für den Verband der Immobilienverwalter Mitteldeutschland e.V. entwickelt werden. Dabei wird auch deren Verbindung zu den systeminternen Klassen dargestellt. Bei allen TYPOlight-eigenen

Klassen wird zur Vereinfachung des Diagramms auf die Darstellung von Attributen und Methoden verzichtet, bei diesen wird lediglich der Name der jeweiligen Klasse sowie des übergeordneten Paketes angezeigt.

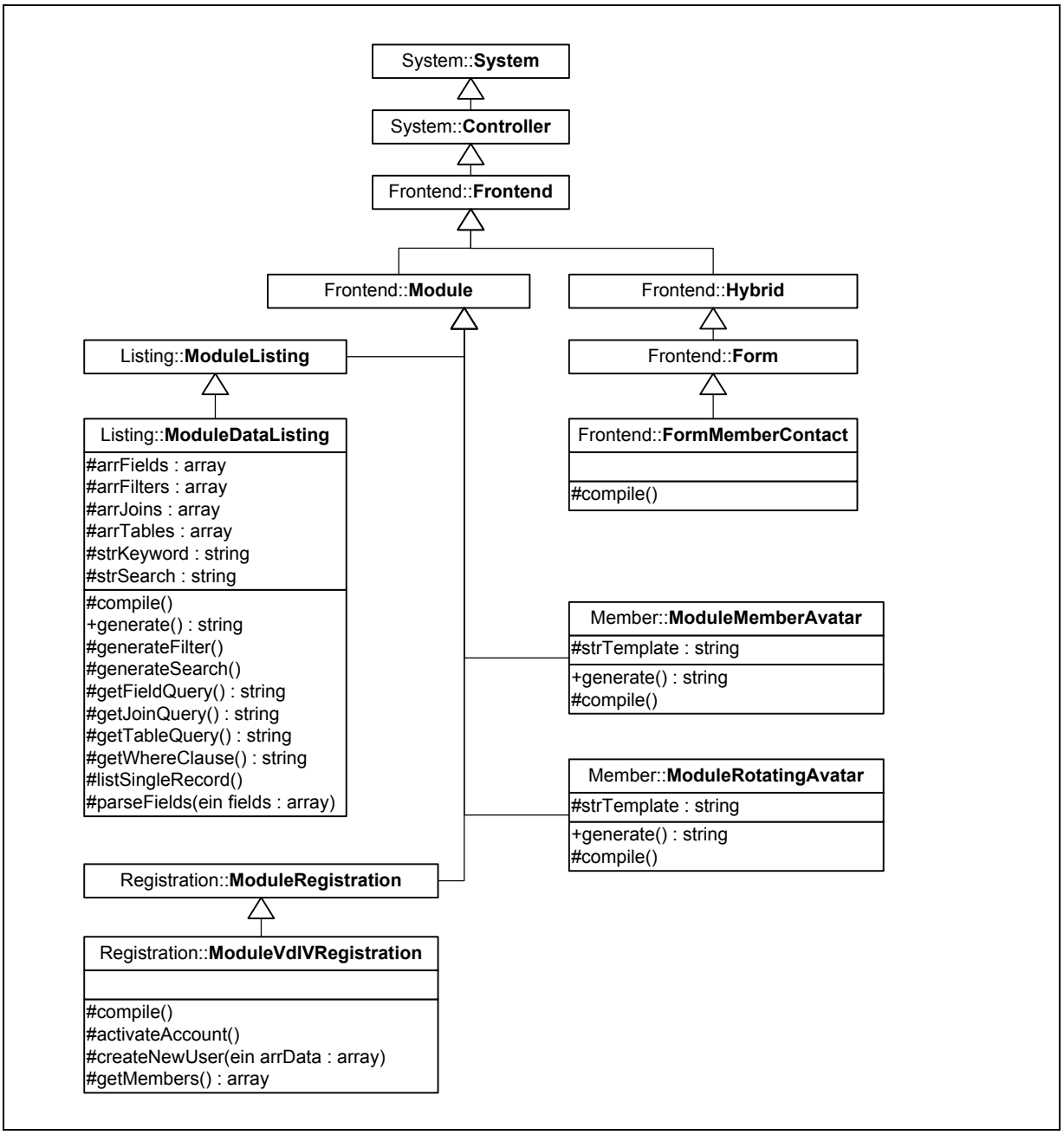

Abbildung 7-6: Klassendiagramm aller TYPOlight-Extensions für den neuen Internetauftritt

Die Klassen „**ModuleMemberAvatar**“ und „**ModuleRotatingAvatar**“ werden direkt von der Klasse „**Module**“ abgeleitet, da bisher in TYPOlight keine Klassen existieren, welche eine ähnliche Funktionalität bereitstellen.

Die Klasse „**Form**“ dient als Ausgangspunkt für die Klasse „**FormMemberContact**“, da hier die abgeleitete Klasse einen Großteil der Funktionalität der Basisklasse nutzen kann.

Für die Klasse „**ModuleVdIVRegistration**“ wird zwar die Basisklasse „**ModuleRegistration**“ verwendet, allerdings muss hier trotzdem die gesamte Funktionalität mit leichten Abweichungen erneut implementiert werden, da die Methoden der Basisklasse als „private“ deklariert wurden und somit für die abgeleitete Klasse nicht zugänglich sind. Lediglich die Methode „**generate**“ kann für die Ausgabe im Frontend wiederverwendet werden.

Ähnlich verhält es sich bei der Klasse „**ModuleDataListing**“. Diese wird zwar von der Klasse „**ModuleListing**“ abgeleitet, allerdings müssen hier fast alle Methoden neu implementiert werden, da die Funktionalität der Basisklasse größtenteils nicht ausreicht. Lediglich die Methode „**formatValue**“ der Basisklasse (im Klassendiagramm nicht dargestellt) kann für die abgeleitete Klasse wiederverwendet werden. Diese Klasse ist außerdem die komplexeste aller Extensions, weshalb hier mehrere Attribute und Methoden benötigt werden.

Für alle anderen Extensions reicht in der Regel die Methode „**compile**“ aus, um die gesamte Funktionalität zu implementieren, da hier keine große Komplexität gegeben ist.

7.4 Sequenzdiagramme

Dieses Kapitel beinhaltet die Sequenzdiagramme aller für die Website des Verbandes der Immobilienverwalter Mitteldeutschland e.V. benötigten Extensions. Zusätzlich wird jedes Diagramm und damit auch die Funktionsweise der entsprechenden Extension erläutert.

Zur Vereinfachung der Diagramme werden Datenbankabfragen nur durch den Aufruf der Methode „**execute**“ der Klasse „Database“ repräsentiert. Der Aufruf der Methode „**prepare**“, welcher diesem in der Regel immer vorausgeht, wird nicht explizit dargestellt, um die Diagramme übersichtlich zu halten.

Außerdem beginnen alle Diagramme mit dem Aufruf der Methode „**compile**“ der jeweiligen Extension, da hier die eigentliche Funktionalität abläuft. Vorgänge in der Methode „**generate**“ wurden wieder zur Bewahrung der Übersichtlichkeit weggelassen, da sie ohnehin keine Auswirkungen auf die Funktionalität der Extensions besitzen.

7.4.1 Extension „Mitglieds-Avatar“

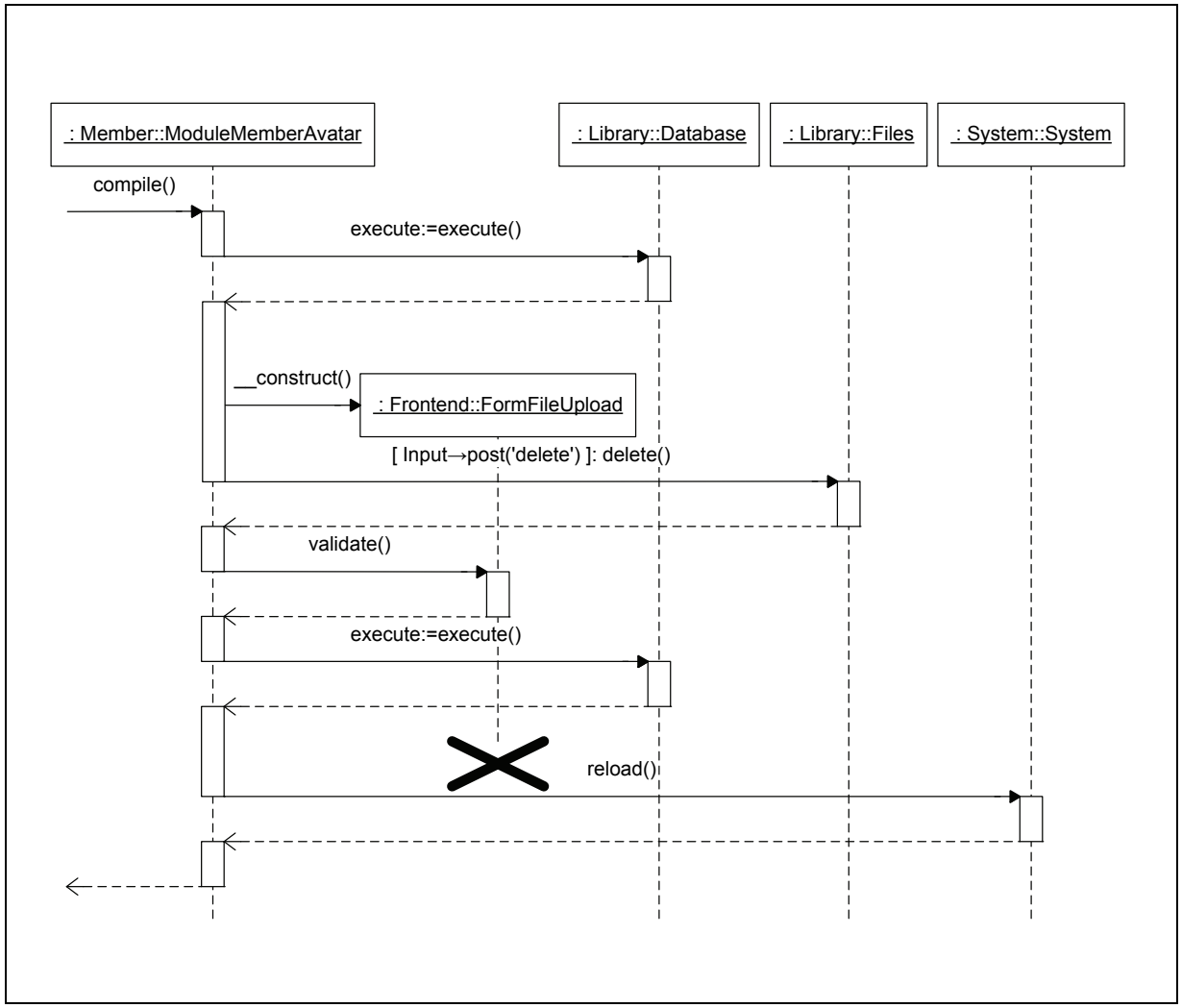

Abbildung 7-7: Sequenzdiagramm für Extension „Mitglieds-Avatar“

In diesem Modul werden zunächst durch eine Datenbankabfrage über die Methode „**execute**“ alle bereits vorhandenen Avatare des jeweiligen Frontend-Mitglieds ermittelt. Anschließend werden in einer Schleife, die abhängig von der Modulkonfiguration einmal oder mehrmals durchlaufen wird, Felder für den Datei-Upload generiert. Dazu werden ein oder mehrere Objekte der Klasse „**FormFileUpload**“ erzeugt. Falls innerhalb eines Schleifendurchlaufs POST-Daten mit dem Befehl zum Löschen des entsprechenden Avatars vorliegen, wird die zugehörige Bilddatei über den Aufruf der Methode „**delete**“ der Klasse „**Files**“ gelöscht. Anschließend wird die hochgeladene Datei, sofern vorhanden, über die Methode „**validate**“ des jeweiligen **FormFileUpload-Objektes** auf ihre Gültigkeit überprüft und ggf. auf dem Server gespeichert. Wenn bis dahin keine Fehler aufgetreten sind und alle hochgeladenen Dateien gültig waren, werden die Änderungen in der Datenbank gespeichert. Anschließend wird über die Methode „reload“ des Systems die Seite neu geladen, damit Änderungen auch sofort sichtbar sind.

7.4.2 Extension „Rotierender Avatar“

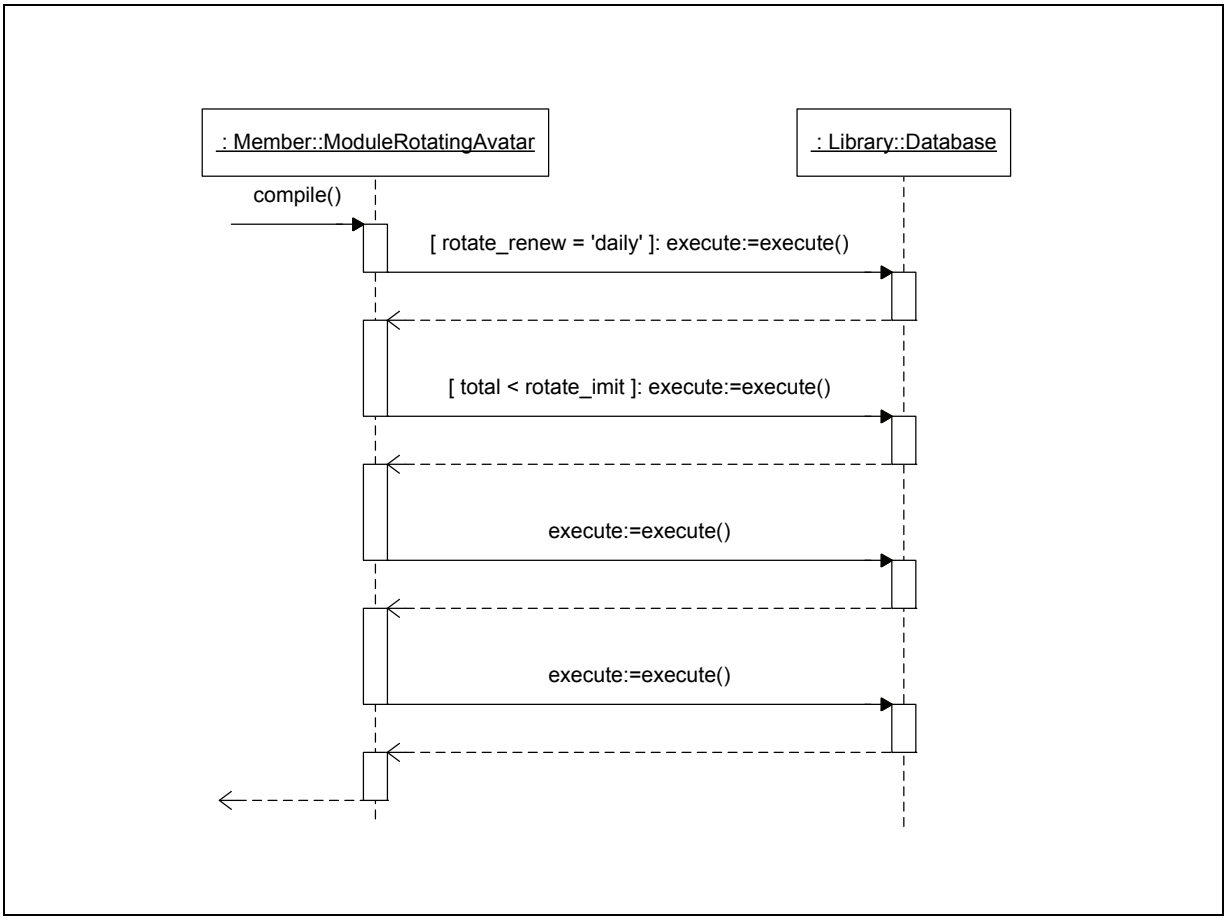

Abbildung 7-8: Sequenzdiagramm für Extension „Rotierender Avatar“

Bei der Ausführung dieses Moduls werden hauptsächlich mehrere Datenbankabfragen über die Methode „execute“ ausgeführt und deren Resultate verarbeitet.

Falls das Modul so konfiguriert wurde, dass die Avatare täglich gewechselt werden sollen, wird zunächst in einer Datenbankabfrage überprüft, ob bereits genügend Datensätze mit dem Zeitstempel des aktuellen Datums vorliegen. Dadurch wird sichergestellt, dass genau so viele Avatare angezeigt werden können, wie in der Modulkonfiguration vorgesehen sind. Falls noch nicht genügend Datensätze vorhanden sind, werden in einem weiteren Datenbankbefehl so viele Datensätze mit dem aktuellen Zeitstempel versehen, dass die Differenz ausgeglichen ist. Dabei werden die Datensätze mit dem ältesten Zeitstempel ausgewählt, um eine gleichberechtigte Auswahl der Mitglieder zu erreichen. Anschließend werden in der nächsten Datenbankabfrage je nach Modulkonfiguration entweder die Mitglieder mit dem Zeitstempel des heutigen Datums oder die Mitglieder mit dem ältesten Zeitstempel ausgewählt, falls die Anzeige bei jedem Seitenaufruf wechseln soll. Zuletzt werden noch alle Datensätze, die selektiert wurden, über einen weiteren Datenbankbefehl mit dem aktuellen Zeitstempel versehen, damit beim nächsten Seitenaufruf nicht dieselben Datensätze ausgewählt werden.

7.4.3 Extension „Erweiterte Auflistung“

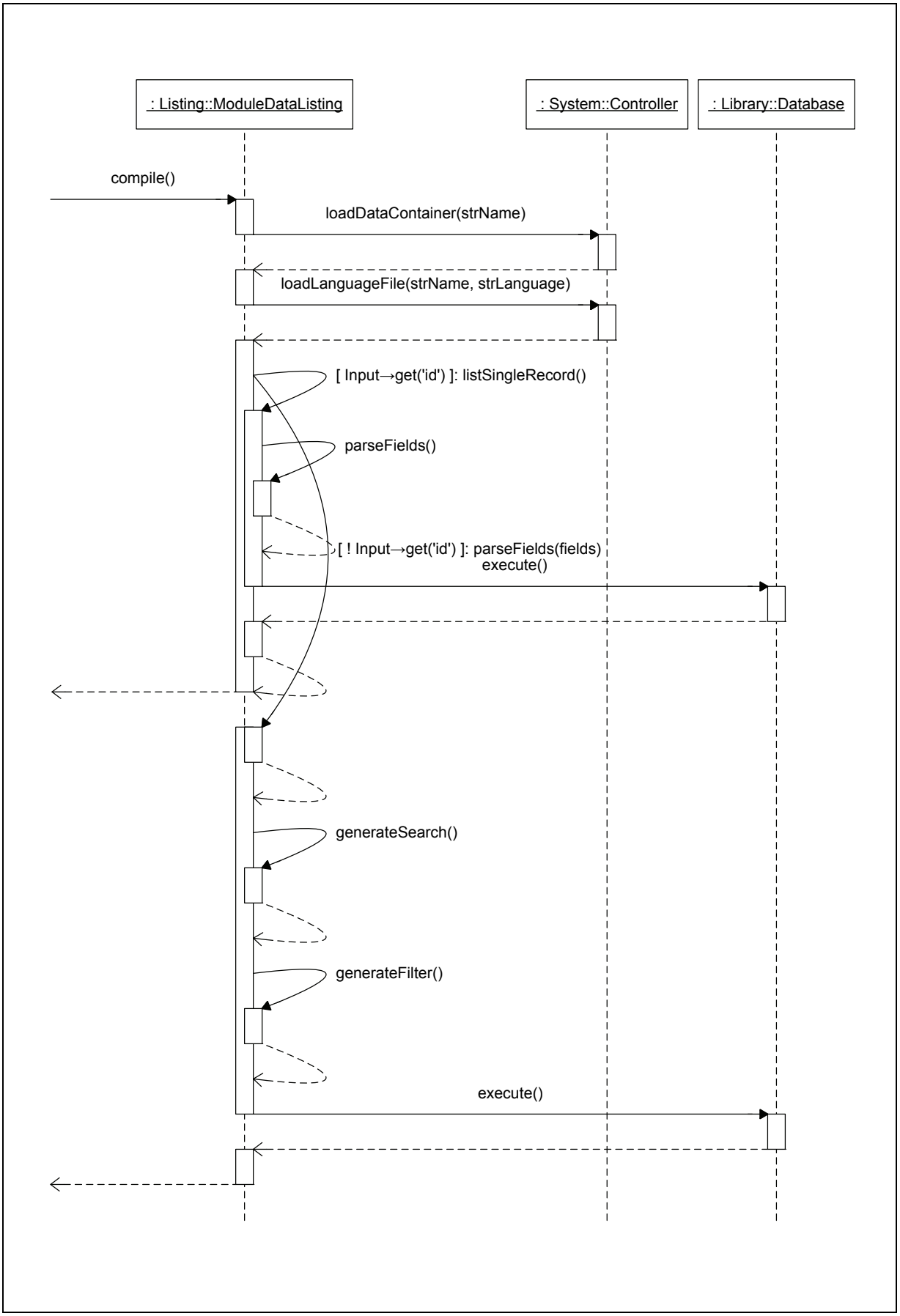

Abbildung 7-9: Sequenzdiagramm für Extension „Erweiterte Auflistung“

In diesem Modul werden zunächst das Data Container Array und die Sprachdateien der jeweiligen Datenbanktabelle über die Methoden „**loadDataContainer**“ und „**loadLanguageFile**“ geladen, da diese für die Ermittlung von Fremdschlüsseln und für die Ausgabe von Datenfeldbeschriftungen benötigt werden.

Anschließend wird die Methode „**listSingleRecord**“ aufgerufen, sofern die ID eines bestimmten Datensatzes als GET-Variable übergeben wurde. Innerhalb dieser Methode werden dann alle Datenfelder, die bei der Anzeige von Detailinformationen dieses Datensatzes einbezogen werden sollen, mit Hilfe der Methode „**parseFields**“ anhand des Data Container Arrays auf Fremdschlüssel überprüft. Wenn Fremdschlüssel vorhanden sind, werden entsprechende Join-Befehle für die nachfolgende Datenbankabfrage erzeugt und in Membervariablen gespeichert. Anschließend werden alle in der Modulkonfiguration festgelegten Datenfelder des ausgewählten Datensatzes aus der Datenbank abgefragt, wobei Fremdschlüssel unter Nutzung der Join-Befehle durch die Werte aus den entsprechenden Tabellen ersetzt werden.

Falls keine ID übergeben wurde, wird innerhalb der Methode „**compile**“ ebenfalls die Methode „**parseFields**“ aufgerufen. Allerdings werden hier nur die Felder auf Fremdschlüssel überprüft, welche bei der tabellarischen Auflistung mehrerer Datensätze angezeigt werden sollen. Auch hier werden wieder entsprechende Join-Befehle erzeugt und gespeichert.

Danach werden über die Methoden „**generateSearch**“ und „**generateFilter**“ abhängig von der Modulkonfiguration die Formularfelder erzeugt, welche im Template für die Darstellung der Such- und Filterformulare benötigt werden. Außerdem wird hier, falls bereits ein Suchbegriff eingegeben oder ein Filter aktiviert wurde, eine entsprechende SQL-Bedingung erzeugt. Diese wird bei der nachfolgenden Datenbankabfrage an die WHERE-Bedingung des SQL-Befehls angehängt, um nur die Datensätze aufzulisten, die dem Suchbegriff oder dem Filter entsprechen.

7.4.4 Extension „VdIV Registrierung“

Registrierung eines neuen Mitglieds

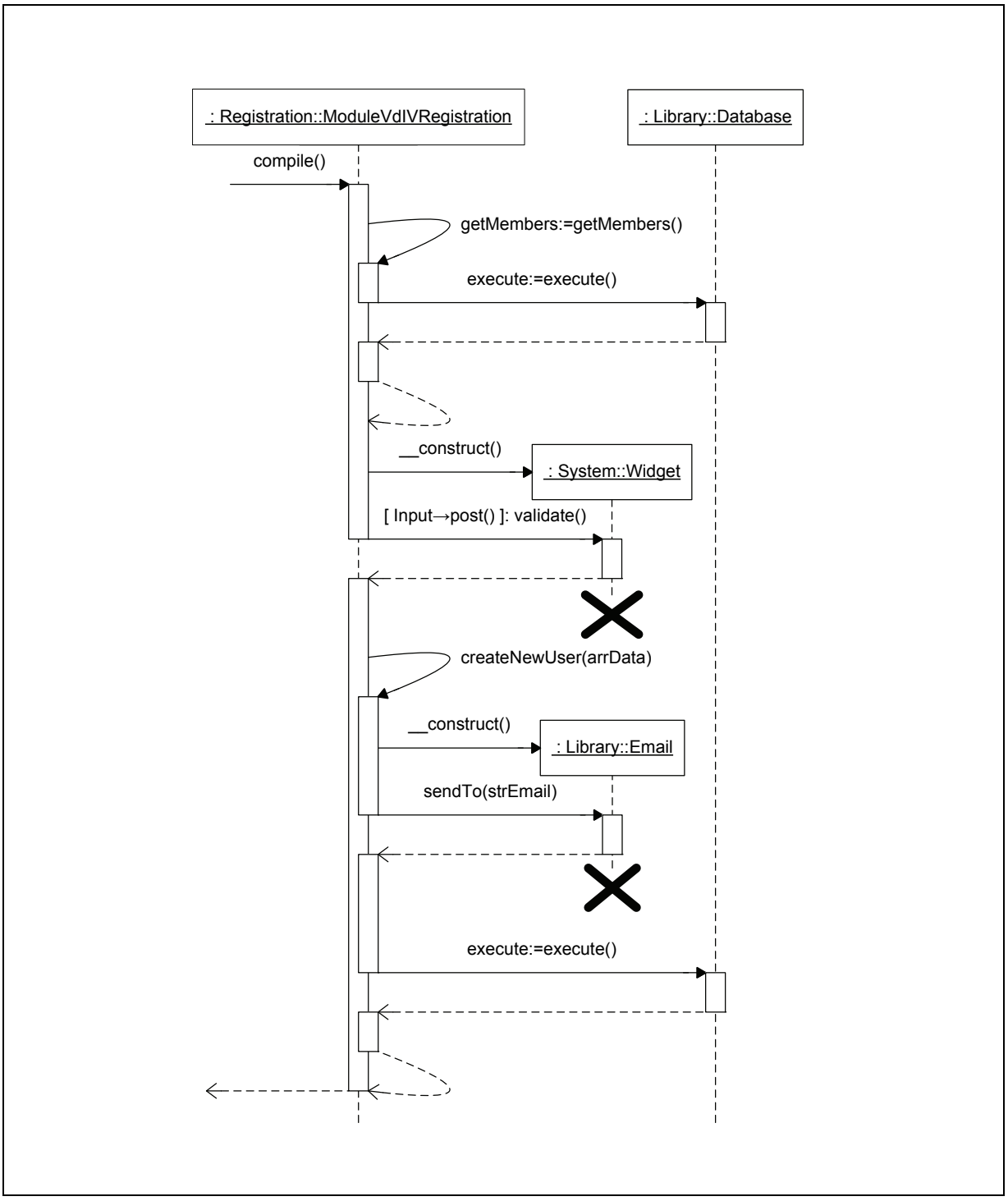

Abbildung 7-10: Sequenzdiagramm für Extension „VdIV Registrierung“ bei der Registrierung eines Mitglieds

Zunächst werden innerhalb der Methode „**getMembers**“ alle Mitglieder aus der Datenbank ermittelt, welche den in der Modulkonfiguration hinterlegten Kriterien entsprechen. Das Resultat wird den in der Konfiguration bereits vorhandenen Formularfeldern als zusätzliche Select-Box hinzugefügt. Diese

dient später zur Auswahl des Mitglieds, welches eine Email mit einem Aktivierungslink erhalten soll. Anschließend werden für alle konfigurierten Eingabefelder einschließlich der Select-Box zur Mitglieder-Auswahl **Widget**-Objekte erzeugt. Sofern das Formular schon abgesendet wurde und POST-Daten vorliegen, werden die eingegebenen Daten über die Methode „**validate**“ jedes einzelnen Widget-Objekts auf Gültigkeit überprüft.

Sofern keine Eingabefehler aufgetreten sind, wird dann die Methode „**createNewUser**“ aufgerufen. Innerhalb dieser Methode wird zunächst ein neues **Email**-Objekt erzeugt. Dieses dient zum Versenden der Aktivierungs-Email an das zuvor ausgewählte Mitglied, sobald die Methode „**sendTo**“ aufgerufen wird. Die Email enthält einen Link mit einem Hashwert, welcher für die Aktivierung benötigt wird.

Zuletzt werden die Benutzereingaben zusammen mit dem Hashwert über einen Datenbankbefehl mit Hilfe der Methode „**execute**“ in der Datenbank abgespeichert.

Aktivierung eines neuen Mitglieds

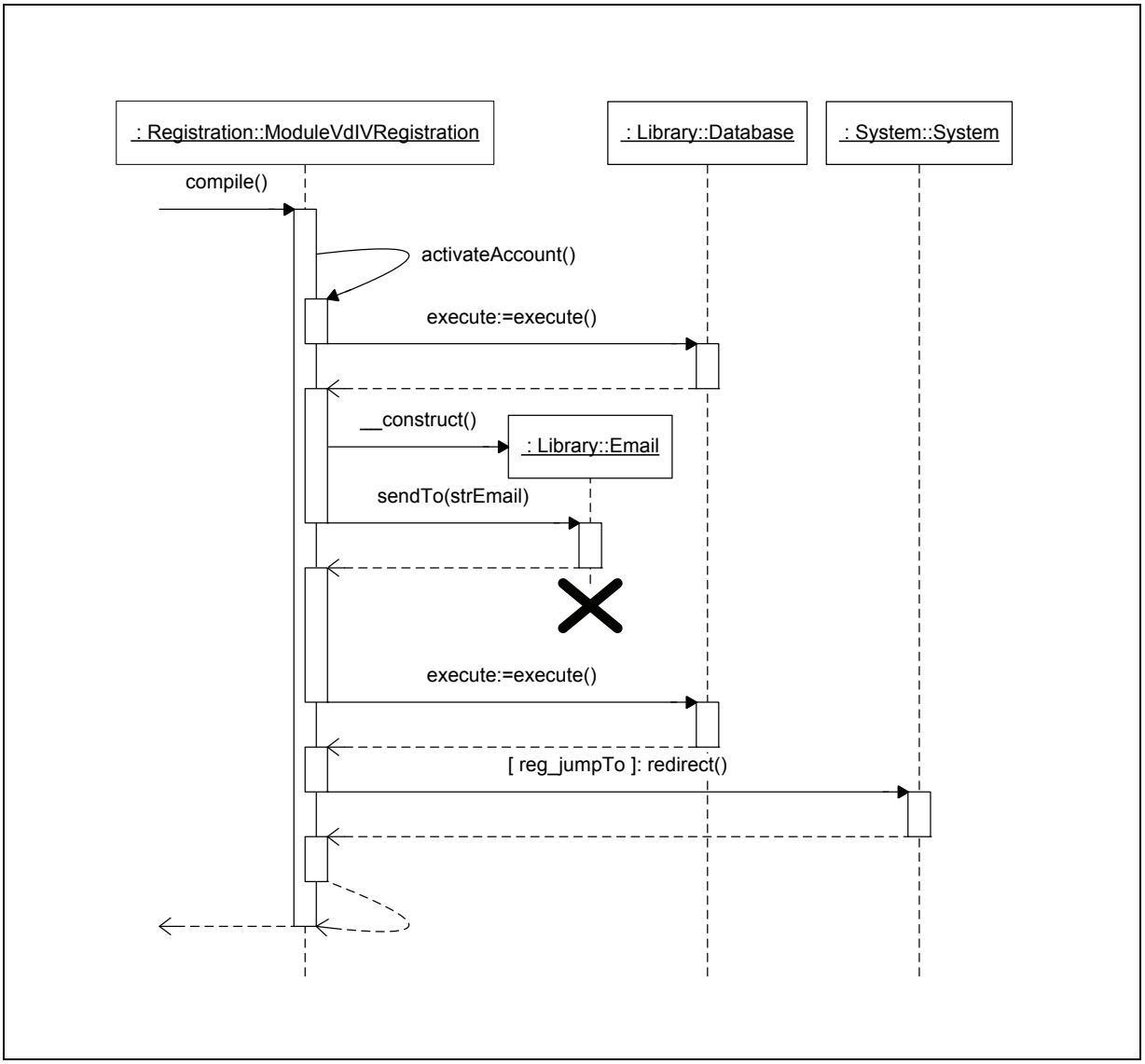

Abbildung 7-11: Sequenzdiagramm für Extension „VdIV Registrierung“ bei der Aktivierung eines Mitglieds

Zur Aktivierung eines neuen Mitglieds wird die Methode „**activateAccount**“ aufgerufen, sofern ein Hashwert als GET-Variable übergeben wurde. Innerhalb dieser Methode wird durch eine Datenbankabfrage überprüft, ob es sich um einen gültigen Hashwert handelt. Ist dies der Fall, wird ein neues **Email**-Objekt erzeugt, welches zum Versenden einer Bestätigungs-Email an das neu registrierte Mitglied beim Aufruf der Methode „**sendTo**“ dient. Anschließend wird der Hashwert aus der Datenbank gelöscht und das neue Mitglied aktiviert.

Zuletzt wird ggf. noch die Methode „**redirect**“ der Klasse „**System**“ aufgerufen, falls in der Modulkonfiguration eine Weiterleitung nach erfolgreicher Aktivierung vorgesehen ist.

7.4.5 Extension „Mitglieder-Kontaktformular“

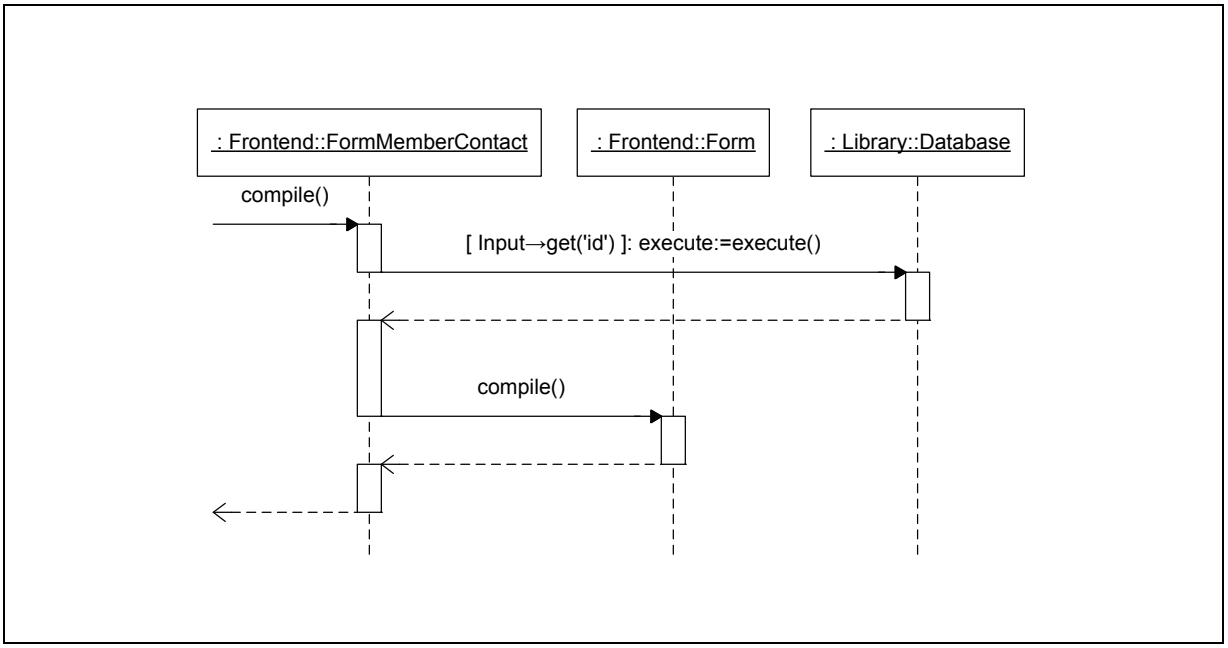

Abbildung 7-12: Sequenzdiagramm für Extension „Mitglieder-Kontaktformular“

Sofern dem Modul „**FormMemberContact**“ eine Mitglieds-ID als GET-Variable übergeben wurde, wird innerhalb der Methode „**compile**“ über eine Datenbankabfrage die Emailadresse des entsprechenden Mitglieds ermittelt. Durch diese wird die in der Konfiguration des Formulars standardmäßig hinterlegte Emailadresse überschrieben. Anschließend wird die Methode „**compile**“ der Elternklasse „**Form**“ aufgerufen, so dass das Formular ganz normal abgearbeitet wird, nun allerdings an die Emailadresse des jeweiligen Mitglieds versendet wird.

7.5 Implementierung und Test

Der PHP-Quellcode aller implementierten Extensions mit erläuternden Kommentaren ist im Anhang B-2 zu finden. Die entsprechende Dokumentation dazu befindet sich im Anhang B-3.

Alle Extensions wurden anhand der in Kapitel 7.2 definierten Testfälle erfolgreich auf Funktion geprüft. Die Ergebnisse der Tests wurden in den vorbereiteten Formblättern festgehalten, welche im Anhang 0 zu finden sind. Da bei den endgültigen Tests keine anderen Ergebnisse als die erwarteten aufgetreten sind, steht in den entsprechenden Feldern der Formblätter nur der Hinweis „wie erwartet“.

Nachdem alle Extensions implementiert und getestet wurden, können sie an den entsprechenden Stellen in der Website eingebunden werden, um die noch offenen Funktionalitäten zu realisieren.

8 Ergebnisse

Dieses Kapitel fasst kurz die Ergebnisse der Umsetzung des Internetauftrittes des Verbandes der Immobilienverwalter Mitteldeutschland e.V. zusammen. Außerdem werden die Extensions, die für diesen Internetauftritt konzipiert und implementiert wurden, kurz im Einsatz auf der Website vorgestellt.

Der neue Internetauftritt des Verbandes der Immobilienverwalter Mitteldeutschland e.V. ist unter der Adresse http://www.immobilienverwalter-mitteldeutschland.de abrufbar. Abbildung 8-1 zeigt einen gekürzten Screenshot der fertigen Website.

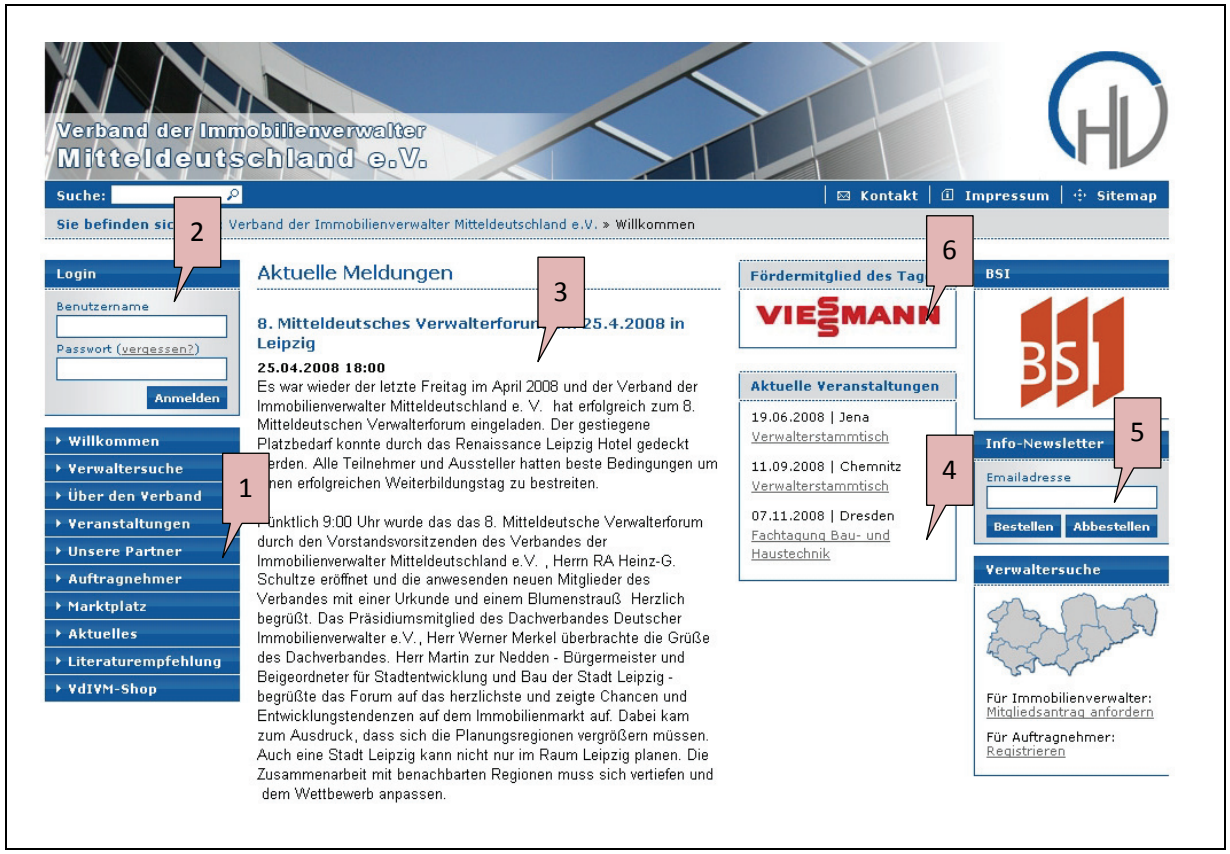

Abbildung 8-1: Ausschnitt der Startseite neuen Internetauftritts vom 28.05.2008

Darauf sind verschiedene Module sichtbar, welche mit Markierungen versehen wurden. Unter anderem wurden für den Internetauftritt ein Navigations-Modul (Markierung #1), ein Login-Formular (Markierung #2), ein News-Modul (Markierung #3), ein Kalender (Markierung #4) sowie ein Formular zum Bestellen oder Abbestellen eines Newsletters (Markierung #5) aus dem Angebot der standardmäßig in TYPOlight vorhandenen Module genutzt.

Außerdem ist in dieser Abbildung die Extension „**Rotierender Avatar**“ (Markierung #6) sichtbar, die hier für den täglichen Wechsel des Avatars konfiguriert wurde.

Abbildung 8-2: Extension „VdIV Registrierung“ im Einsatz auf der Website

Abbildung 8-2: Extension „VdIV Registrierung“ im Einsatz auf der Website zeigt die Extension „**VdIV Registrierung**“, die wie ein normales Registrierungsformular dargestellt wird. Ergänzt wird dieses durch die Select-Box zur Auswahl des Referenzmitglieds, welches nach der Registrierung die Aktivierung des neuen Frontend-Mitglieds durchführen muss.

Abbildung 8-3: Extension „Mitglieder-Kontaktformular“ im Einsatz auf der Website

In Abbildung 8-3 ist die Extension „Mitglieder-Kontaktformular“ zu sehen. Diese ist von dem standardmäßig in TYPOlight integrierten Formular nur durch die Anzeige des Namens des zu kontaktierenden Mitglieds zu unterscheiden. Wird das Formular ohne Übergabe einer Mitglieds-ID als GET-Variable aufgerufen, so arbeitet es wie das Standard-Formular und kein Mitgliedsname wird angezeigt.

Abbildung 8-4: Extension „Erweiterte Auflistung“ im Einsatz auf der Website

Abbildung 8-4 zeigt die Extension „Erweiterte Auflistung“. Im dargestellten Screenshot ist bereits ein Filter aktiviert, so dass darunter nur Datensätze aufgelistet werden, die dem Filter „Webdesign“ entsprechen. Links neben dem Filter ist das Suchformular zu sehen, welches die Auswahl eines zu durchsuchenden Feldes sowie die Eingabe eines Suchbegriffes erlaubt.

Abbildung 8-5: Extension „Mitglieds-Avatar“ im Einsatz auf der Website

Die Extension „Mitglieds-Avatar“ wird in Abbildung 8-5 dargestellt. Das Modul ist in diesem Beispiel für die Verwaltung eines einzelnen Avatars konfiguriert. Im dargestellten Screenshot wurde bereits ein Firmenlogo hochgeladen. Dieses kann durch Nutzung des Upload-Feldes durch eine andere Grafik ersetzt werden oder durch setzen des Häkchens bei „Bild entfernen“ komplett gelöscht werden.

9 Zusammenfassung und Ausblick

Zusammenfassend lässt sich die Umsetzung des Internetauftritts für den Verband der Immobilienverwalter Mitteldeutschland e.V. als abgeschlossen bezeichnen. Alle zu Beginn gestellten Anforderungen an grafische Gestaltung, Inhalt und Funktionen sowie Technik werden erfüllt.

Auch wurden im Rahmen dieser Diplomarbeit Vor- und Nachteile von sowohl TYPOlight als auch Typo3 im direkten Vergleich aufgezeigt, so dass diese Erkenntnisse bei der Auswahl eines geeigneten Content-Management-Systems für zukünftige Projekte behilflich sein können.

Weiterhin wurde der interne Aufbau von TYPOlight erläutert und eine Einführung in die Programmierung von Extensions für dieses System vorgenommen. Dies ermöglicht Entwicklern, die TYPOlight erweitern möchten, einen schnelleren Einstieg als die Informationen, die auf der TYPOlight-Website zu finden sind.

Allerdings wurde nach dem Upload der fertigen Website auf den öffentlichen Server eine kleine Schwäche des gesamten Systems festgestellt: die Ladezeiten beim Aufruf einer Seite sind relativ hoch. Zurückzuführen ist dies vermutlich auf die Nutzung eines geteilten Webservers (Shared Host), welcher noch andere Internetpräsenzen beherbergt. Dadurch reicht die Leistung des Servers beim Ausliefern einer Seite wahrscheinlich nicht mehr aus, um das komplexe Seitenlayout der Website mit seinen vielen eingebundenen Modulen rasch abzuarbeiten. Dieser Umstand konnte durch den Benchmark-Test des Systems auf einem lokalen Testserver mit sehr einfachen Seitenlayouts leider nicht festgestellt werden. Hier bestehen also für die Zukunft noch Optimierungsmöglichkeiten, so könnten die Seitenlayouts beispielsweise mit weniger Modulen ausgestattet werden. Stattdessen könnten die jeweiligen Module durch statischen HTML-Code im Template ersetzt werden und die eigentliche Funktionalität auf separate Seiten mit jeweils einem eingebundenen Modul ausgelagert werden.

Aber auch für die Zukunft von TYPOlight besteht hier noch Verbesserungspotential, so könnten z.B. die Datenbankabfragen von TYPOlight beim Ausführen von Modulen optimiert oder dessen Caching-Mechanismen verbessert werden.

Insgesamt lässt sich TYPOlight durch die sehr gute Umsetzung des Model-View-Controller-Prinzips und die leichte Erweiterbarkeit trotzdem als modernes Content-Management-System beschreiben, welches noch viel Potential für die Zukunft besitzt. Insbesondere seit Anfang des Jahres 2008 befindet sich das System auf dem Weg, ein professionelles Projekt zu werden, welches nicht länger nur von einem einzelnen Entwickler betreut wird. So wurde auf dem ersten offiziellen TYPOlight-Treffen am 18. April 2008 in Würzburg das derzeitige TYPOlight-Team vorgestellt. Dieses umfasst momen-

tan[51] sechs Entwickler aus dem Bereich Webdesign und -programmierung, welche das Projekt seit einigen Monaten neben dem Kernentwickler Leo Feyer betreuen.

Ein Teammitglied entwickelt beispielsweise seit Mitte April 2008 ein Extension Repository, wie es von Typo3 her bekannt ist. Außerdem wurde am 24.05.2008 das offizielle neue TYPOlight-Logo vorgestellt, welches von einem Designer aus dem Team entworfen wurde.[52]

Die Entwicklergemeinschaft rund um das Team von TYPOlight ist derzeit mit ca. 1800 Mitgliedern weltweit noch relativ klein. Doch das Potential des Systems legt die Vermutung nahe, dass diese Gemeinschaft in absehbarer Zeit noch stark wachsen wird. So könnte TYPOlight in einigen Jahren vielleicht sogar an den Erfolg von Typo3 anknüpfen und eines der verbreitetsten Content-Management-Systeme der Welt werden.

[51] Stand: 28.05.2008
[52] Vgl. Feier, Leo: TYPOlight community, 2008, Internet: http://www.typolight.org/community.html

Anhang A

A-1 Tabellarische Übersicht des Vergleichs von TYPOlight mit Typo3

1. Produkt		
Produktname	Typo3	TYPOlight
Hersteller / Anbieter	Typo3	TYPOlight
Website	http://typo3.org	http://typolight.org
2. Qualitative und übergeordnete Faktoren		
Lizenz		
Art der Lizenz	Open Source (GPL)	Open Source (LGPL)
Support		
Testmöglichkeiten	ja	ja
Schulungs- und Supportangebote über Hersteller / Anbieter	ja	eingeschränkt
Schulungs- und Supportangebote über Drittfirma	ja	eingeschränkt
Vor-Ort-Service	nein	nein
Telefon-Hotline	ja	nein
Internet-Communities für Benutzer / Entwickler	ja	ja
Dokumentation		
Benutzerhandbuch	ja	ja
Administrationshandbuch	ja	ja
Entwicklerhandbuch	ja	eingeschränkt
Technische Referenz	ja	ja
API-Referenz	ja	ja
Kontextbezogene Online-Hilfe	ja	ja
Dokumentation von Fehlermeldungen	ja	eingeschränkt
Erweiterbarkeit		
Erlaubnis zur Manipulation des Quellcodes	ja	ja
Offengelegte API	ja	ja
Erweiterung über Module/Extensions möglich	ja	ja
Performance		
Load Balancing	nein	nein
Caching-Mechanismus	ja	ja
Staging- und dynamische Methoden kombinierbar	ja	nein

3. Architektur und Infrastruktur		
Server		
Datenbank als Content Repository / Dateisystem	ja	ja
Anbindung externer Datenbanken	via Extension	nein
Clients		
Backend komplett Browser-basiert	ja	ja
Migrations- und Portierungsfähigkeiten		
Konzepte für die Integration bestehender HTML-Seiten	eingeschränkt	nein
Importmöglichkeiten aus bestehenden Datenbanken	via Extension	nein
Importmöglichkeiten aus strukturierten Rohdaten	via Extension	nein
Datensicherung und -archivierung		
Backupstrategien	via Extension	via Extension
Transparentes Backup	ja	ja
Möglichkeit zur Datenhaltung auf Spiegelserver	nein	nein
Archivierungsmethoden	ja	eingeschränkt
Recherchemöglichkeiten im Archiv	ja	nein
Sicherstellung der Datenkonsistenz bei Systemabbruch	nein	nein
4. Inhaltserstellung		
Templates		
Template-Erstellung durch Skripting	ja	ja
Template-Erstellung grafisch über Browser (WYSIWYG)	nein	nein
Template-Erstellung durch HTML-Editoren	ja	ja
Frei definierbare und positionierbare Elemente im Template	ja	ja
Verwendung von Style Sheets möglich	ja	ja
Möglichkeit zur Erstellung von Formularen	ja	ja
Redaktion		
WYSIWYG-Editor	ja	ja
Vorschaufunktion	ja	ja
Direktes Publizieren aus Office-Dokumenten	ja	nein
Integration eigener Skripte	ja	eingeschränkt
Publizieren in andere Zielformate als HTML	ja	eingeschränkt
automatische Generierung von Print-Versionen der Seite	ja	eingeschränkt
Dateien per Upload-Button anhängbar	ja	nein
Browserunabhängigkeit und -kompatibilität des erzeugten Codes	ja	ja
Mehrsprachigkeit der Oberfläche und Hilfe	ja	ja

Textinhalte		
Eingabe von Inhalten ohne HTML-Kenntnisse	ja	ja
Texteditierung im Rich Text Modus	ja	ja
Einschränkungen bezüglich Rich Text möglich	ja	ja
Rechtschreibprüfung	ja	ja
Limitierung der Zeichenanzahl	nein	nein
Mit Anzeige der verbleibenden Zeichen	nein	nein
Vorgabe von Default-Werten für Eingabefelder	ja	ja
Multimediale Inhalte		
proportionale Änderung der Auflösung von Bildern	ja	ja
Zuschneiden von Bildern	nein	nein
Formatkonvertierung von Bildern	nein	nein
direkte Einbindung anderer Medienformate	ja	eingeschränkt
Asset Management	ja	nein
Metadaten		
Zeitliche Steuerung	ja	ja
manuelle Eingabe von Schlagwörtern	ja	ja
Definition und Weiterverarbeitung beliebiger Metatags	ja	nein
Versionskontrolle		
Versionskontrolle	ja	ja
Mehrsprachigkeit für Inhalte	ja	eingeschränkt
Link Management		
Zyklische Überprüfung von Links	nein	nein
Automatische Anpassung aller internen Links bei Dateiänderungen	nein	ja
Verschieben von Dateien per Drag & Drop	nein	nein
Automatische Generierung und Aktualisierung einer Sitemap	ja	ja
Suchfunktionen		
Suchmöglichkeiten für Autoren und Besucher der Site	ja	ja
Suche über Metainformationen	ja	eingeschränkt
Suche innerhalb von Dateien	via Extension	nein
Globale Funktion für ''Suchen / Ersetzen''	nein	nein
5. Benutzerverwaltung und Workflow		
Kapazität		
limit concurrent users (unlimited: 0)	unbegrenzt	unbegrenzt
Schließt die Benutzerverwaltung auch Endbenutzer mit ein	via Extension	ja
Mandantenfähigkeit	ja	ja

Benutzerverwaltung		
Möglichkeit zur Definition von Benutzergruppen	ja	ja
Vererbung von Benutzerrechten	ja	ja
Vererbung von Administratorrechten	nein	nein
Einschränkung von Benutzerrechten	ja	ja
Workflow		
Mehrstufige Freigabekontrolle	ja	eingeschränkt
Möglichkeit zur freien Festlegung der Freigabewege	nein	nein
Locking-Mechanismus	nein	nein
Erinnerungsfunktionen	ja	ja
Inhaltsbezogene Rechte		
Zugriffsbeschränkung nach Dateien	ja	ja
Zugriffsbeschränkung nach Datensätzen	nein	nein
Zugriffsbeschränkung nach Datenfeldern	ja	ja
Differenzierung der Zugriffsbeschränkungen	ja	eingeschränkt

A-2 Konfiguration des Client- und Server-Rechners für den Benchmark Test

Server-Rechner

CPU: Intel Core Duo T2300 (Taktfrequenz 2 x 1,66 GHz)

Hauptspeicher: 2 GB DDR-RAM

Betriebssystem: Windows XP Professional SP2 (Build 5.1.2600)

Webserver: Apache HTTP Server 2.2.8

PHP-Version: 5.2.5

MySQL-Version: 5.0.51a

TYPOlight-Version: 2.5.7

Typo3-Version: 4.1.6

Client-Rechner

CPU: AMD Athlon XP 2800+ (Taktfrequenz 2,081 GHz)

Hauptspeicher: 1 GB DDR-RAM

Betriebssystem: Windows XP Professional SP2 (Build 5.1.2600)

Benchmark-Software: ApacheBenchmark

A-3 Messwert-Tabelle des Benchmark-Tests

System		TYPOlight				Typo3				Statische HTML-Seite	
		Mit Cache		Ohne Cache		Mit Cache		Ohne Cache			
Anfragen gesamt		500	500	500	500	500	500	500	500	500	500
Konkurrier. Anfragen		1	100	1	100	1	100	1	100	1	100
Textlänge	10 Zeichen	155,281	88,813	141,156	93,688	127,250	58,094	276,219	120,000	22,219	24,875
	50 Zeichen	156,875	89,125	141,313	92,313	127,545	57,902	276,625	119,938	22,188	24,781
	100 Zeichen	157,438	90,813	142,813	91,313	127,545	58,406	277,188	119,438	22,313	24,938
	500 Zeichen	156,406	91,406	143,375	92,094	128,080	58,563	278,063	119,554	22,938	25,500
	1000 Zeichen	160,094	91,656	149,250	92,938	128,036	58,156	279,688	120,000	23,719	25,500
	1500 Zeichen	158,250	93,688	165,906	93,500	128,170	58,406	280,125	119,330	22,875	25,563
	5000 Zeichen	166,438	97,156	163,750	92,906	129,732	58,063	281,250	120,893	24,469	26,719
	10000 Zeichen	181,406	104,531	174,438	104,469	128,170	58,531	280,500	119,821	25,813	28,594
	25000 Zeichen	210,000	129,656	196,125	127,688	131,563	58,844	286,063	120,982	31,281	34,313
	50000 Zeichen	267,094	168,688	261,563	169,406	138,036	59,375	294,563	121,438	37,813	42,188
	75000 Zeichen	317,771	205,626	335,156	213,313	176,146	82,938	322,552	122,063	94,406	57,688
	100000 Zeichen	343,135	224,563	404,896	260,625	190,990	101,938	341,823	124,875	111,844	65,719
		Alle Messwerte in Millisekunden									

A-4 Formblätter für Testfälle der TYPOlight-Extensions

Formblatt für Extension "Mitglieds-Avatar"			
Eingabefelder und Störfälle			
Eingabefeld	erlaubte Werte / Formate	mögliche Störfälle	Fehlerbehandlung
Upload-Feld für Bilddatei	Bilddateien vom Typ GIF, JPG oder PNG	falsches Dateiformat	Ausgabe Fehlermeldung
		zu hohe Dateigröße	Ausgabe Fehlermeldung
		zu große Bildweite	Ausgabe Fehlermeldung
		zu große Bildhöhe	Ausgabe Fehlermeldung
		Dateiname schon vorhanden	Datei umbenennen
Checkbox zum Löschen eines Avatars	0 / 1	Datei existiert nicht mehr	Verweis auf Datei aus Datenbank entfernen
Sonstige Störfälle			
Störfall		Fehlerbehandlung	
-		-	
Resultierende Testfälle			
Testfall		Erwartetes Ergebnis	Eingetretenes Ergebnis (wenn abweichend)
Upload einer GIF-Datei		Datei wird gespeichert und angezeigt	[wie erwartet]
Upload einer JPG-Datei		Datei wird gespeichert und angezeigt	[wie erwartet]
Upload einer PNG-Datei		Datei wird gespeichert und angezeigt	[wie erwartet]
Löschen eines Avatars		Datei wird gelöscht und Verweis aus Datenbank entfernt	[wie erwartet]
Upload GIF-Datei mit zu großer Bildweite		Ausgabe Fehlermeldung	[wie erwartet]
Upload GIF-Datei mit zu großer Bildhöhe		Ausgabe Fehlermeldung	[wie erwartet]
Upload JPG-Datei mit zu großer Bildweite		Ausgabe Fehlermeldung	[wie erwartet]
Upload JPG-Datei mit zu großer Bildhöhe		Ausgabe Fehlermeldung	[wie erwartet]
Upload PNG-Datei mit zu großer Bildweite		Ausgabe Fehlermeldung	[wie erwartet]
Upload PNG-Datei mit zu großer Bildhöhe		Ausgabe Fehlermeldung	[wie erwartet]
Upload einer Datei anderen Formats		Ausgabe Fehlermeldung	[wie erwartet]
Upload einer Datei mit gleichem Dateinamen wie bereits vorhandene Datei		Datei wird unter anderem Namen gespeichert	[wie erwartet]
Löschen eines Avatars, wenn Verweisziel (Bilddatei) nicht mehr existiert		Verweis auf Datei wird aus Datenbank entfernt	[wie erwartet]

Formblatt für Extension "Rotierender Avatar"			
Eingabefelder und Störfälle			
Eingabefeld	erlaubte Werte / Formate	mögliche Störfälle	Fehlerbehandlung
-	-	-	-
Sonstige Störfälle			
Störfall		Fehlerbehandlung	
noch kein Avatar bei Mitglied vorhanden		keine Anzeige, Ausgabe der zusätzlichen Datenfelder (falls vorhanden)	
Resultierende Testfälle			
Testfall		Erwartetes Ergebnis	Eingetretenes Ergebnis (wenn abweichend)
Seitenaktualisierung		Anzeige neuer Avatare	[wie erwartet]
Seitenaktualisierung nach Datumswechsel		Anzeige neuer Avatare	[wie erwartet]
Seitenaktualisierung, wenn noch bei keinem Mitglied Avatare vorhanden		Keine Anzeige, Ausgabe zusätzlicher Datenfelder	[wie erwartet]

Formblatt für Extension "Erweiterte Auflistung"			
Eingabefelder und Störfälle			
Eingabefeld	erlaubte Werte / Formate	mögliche Störfälle	Fehlerbehandlung
Textfeld zur Eingabe eines Suchbegriffs	beliebiger Text (Zahlen, Buchstaben, Sonderzeichen)	SQL-Einschleusung	Maskierung bestimmter Zeichen zum Abfangen möglicher SQL-Befehle
Select-Box zur Auswahl des zu duchsuchenden Feldes	angezeigte Felder	-	-
Select-Box zur Auswahl eines Filters	angezeigte Filter	-	-
Sonstige Störfälle			
Störfall		Fehlerbehandlung	
-		-	
Resultierende Testfälle			
Testfall		Erwartetes Ergebnis	Eingetretenes Ergebnis (wenn abweichend)
Filterung nach einem bestimmten Wert		Anzeige aller Datensätze, die dem Filter entsprechen	[wie erwartet]
Filterung nach mehreren Werten		Anzeige aller Datensätze, die allen Filtern entsprechen	[wie erwartet]
Suche nach einem bestimmten Wert in einem bestimmten Datenfeld		Anzeige aller Datensätze, die den gesuchten Wert im ausgewählten Datenfeld enthalten	[wie erwartet]
Eingabe von SQL-Befehlen im Suchfeld		SQL-Befehl wird ignoriert, ggf. Anzeige von Suchergebnissen	[wie erwartet]

Formblatt für Extension "VdIV Registrierung"			
Eingabefelder und Störfälle			
Eingabefeld	erlaubte Werte / Formate	mögliche Störfälle	Fehlerbehandlung
Select-Box zur Auswahl eines Mitglieds	angezeigte Mitglieder	kein Mitglied ausgewählt	Ausgabe Fehlermeldung
Sonstige Störfälle			
Störfall		Fehlerbehandlung	
-		-	
Resultierende Testfälle			
Testfall		Erwartetes Ergebnis	Eingetretenes Ergebnis (wenn abweichend)
Absenden des korrekt ausgefüllten Registrierungsformulars		Mitglied ist registriert, aber noch nicht aktiviert; ausgewähltes Mitglied bekommt Aktivierungsemail	[wie erwartet]
Absenden des Registrierungsformulars, wenn noch kein Mitglied ausgewählt wurde		Ausgabe Fehlermeldung	[wie erwartet]
Aufrufen des Aktivierungslinks		Mitglied wird aktiviert und bekommt Bestätigungsemail	[wie erwartet]

Formblatt für Extension "Mitglieder-Kontaktformular"			
Eingabefelder und Störfälle			
Eingabefeld	erlaubte Werte / Formate	mögliche Störfälle	Fehlerbehandlung
[beliebig]	[beliebig]	-	[Fehlerbehandlung durch Validierungsfunktion der entsprechenden Formularfelder]
Sonstige Störfälle			
Störfall		Fehlerbehandlung	
Eingabe einer ungültigen Mitglieds-ID in der Adresszeile des Browsers		Formular wird an hinterlegte Administrator-Emailadresse versendet	
Resultierende Testfälle			
Testfall		Erwartetes Ergebnis	Eingetretenes Ergebnis (wenn abweichend)
Absenden des korrekt ausgefüllten Kontaktformulars		Anfrage wird an registrierte Emailadresse des entsprechenden Mitglieds versendet	[wie erwartet]
Absenden des korrekt ausgefüllten Kontaktformulars bei ungültiger Mitglieds-ID		Administrator erhält Anfrage	[wie erwartet]

Anhang B (CD-ROM)

B-1 Quellcode der Beispiel-Extension „Gästebuch“

Verzeichnis: Beispiel-Extension/

Die Beispiel-Extension „Gästebuch“, die in Kapitel 6 entwickelt wurde, ist vollständig auf der CD-ROM enthalten und kann ggf. einer vorhandenen TYPOlight-Installation hinzugefügt werden.

B-2 Quellcode aller TYPOlight-Extensions

Verzeichnis: VdIVM-Extensions/

Alle Extensions, die für die Website des VdIVM entwickelt wurden, sind vollständig und installationsfertig auf der CD-ROM enthalten.

B-3 Dokumentation zu allen TYPOlight-Extensions

Verzeichnis: Dokumentation/

Zu allen Extensions für die Website des VdIVM wurde mit Hilfe von PHPDocumentor (Version 1.4.2) eine Dokumentation bestehend aus HTML-Dateien generiert. Diese ist ebenfalls auf der CD-ROM enthalten.

B-4 Kopien aller Internet-Quellen als PDF-Dokument

Verzeichnis: Internetquellen/

Da sich Informationen im Internet schnell ändern können, sind auf der CD-ROM Kopien aller für die Bearbeitung dieser Diplomarbeit herangezogenen Internetseiten in Form von PDF-Dokumenten enthalten.

Neben den hier genannten Anhängen ist außerdem eine **Kopie der gesamten Diplomarbeit in Form eines PDF-Dokuments** auf der CD-ROM enthalten.

Literaturverzeichnis

Scholl, Martin: Lastenausgleich mit Typo3. Wie das CMS lastverteilt betrieben wird, in: T3N, Ausgabe 7 (03/2007 – 05/2007) vom 07.03.2007

Quade, Jürgen / Kunst, Eva-Katharina: Linux-Treiber entwickeln. Eine systematische Einführung in Gerätetreiber für den Kernel 2.6, 2. Auflage, 2006

Laborenz, Kai / Ertel, Andrea / Wendt, Thomas u.a.: TYPO3 4.0: Das Handbuch für Entwickler, 2. Auflage, 2006

Niederlag, Peter: Typo3 4.0: Workspaces. Optimierte redaktionelle Workflows, in: T3N, Ausgabe 2 (12/2005 – 02/2006) vom 05.12.2005

Internetquellen

O.V.:[53] Apache HTTP Server Version 2.0: Apache-Kernfunktionen, 2008,
Internet: http://httpd.apache.org/docs/2.0/mod/core.html, Abruf am 12.04.2008

O.V.: Apache HTTP Server Version 2.0: Allgemeine Direktiven der Apache-MPMs, 2008,
Internet: http://httpd.apache.org/docs/2.0/mod/mpm_common.html, Abruf am 12.04.2008

O.V.: Cascading Style Sheets, 2008,
Internet: http://de.wikipedia.org/wiki/Cascading_Style_Sheets, Abruf am 17.04. 2008

O.V.: CMS-Vergleich.de – Content-Management-Systeme, 2007,
Internet: http://www.cms-vergleich.de/cms/?s=cmsliste&cid=4, Abruf am 07.04.2008

O.V.: Frame (HTML), 2008,
Internet: http://de.wikipedia.org/wiki/Frame_(HTML), Abruf am 07.04.2008

O.V.: Model View Controller, 2008,
Internet: http://de.wikipedia.org/wiki/Model_View_Controller, Abruf am 26.04.2008

O.V.: Typo3 CMS: Geschichte, 2006,
Internet: http://typo3.com/Geschichte.1268.0.html?&L=2, Abruf am 07.04.2008

[53] O.V. = „Ohne Verfasser“

Hinderink, Daniel: Typo3 spricht MS-Office, 2003,
Internet: http://www.contentmanager.de/magazin/news_h5607_typo3_spricht_ms-office.html, Abruf am 14.04.2008

Roth, Jörg: Typo3 – Statische Seiten erstellen, 2003,
Internet: http://www.typo3.net/forum/list/list_post//3058/, Abruf am 19.05.2008

Feier, Leo: TYPOlight community, 2008,
Internet: http://www.typolight.org/community.html, Abruf am 29.05.2005

O.V.: Typo3 und die Mehrsprachigkeit, 2006,
Internet: http://www.c-dev.ch/kaepten/typo3/tutorial_1_sprachen.html, Abruf am 15.04.2008

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde nach meiner besten Kenntnis bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Hof, den 10.06.2008 Marcus Kliche