G. Szwillus, J. Ziegler (Hrsg.): Mensch & Computer 2003: Interaktion in Bewegung.
Stuttgart: B. G. Teubner, 2003, S. 247-256

Ontologiebasierte Vorgehensweise zur Modellierung komponentenorientierter Web-Anwendungen

Michael Wissen, Jürgen Ziegler
Fraunhofer IAO, Stuttgart / Universität Duisburg-Essen

Zusammenfassung

Dieser Beitrag thematisiert eine durchgängige und integrierte Methodik mit entsprechenden Werkzeugen für den komponentenorientierten Entwurf sowohl content- als auch applikationsorientierter Web-Anwendungen. Hierzu werden Konzepte bestehender Entwurfsmethoden um Aspekte der Metadatenmodellierung sowie der Sichten- und Navigationsmodellierung erweitert und in einem Methodenverbund bereitgestellt. Kern der Vorgehensweise ist die Anwendung von Metamodellen und Modellierungstechniken für die verschiedenen Entwurfsbereiche einer Web-Anwendung sowie von Werkzeugen zur prototypischen Umsetzung der erstellten Modelle in eine lauffähige Web-Applikation. Das Verfahren verfolgt sowohl auf der Modellierungs- als auch der Realisierungsebene einen weitestgehend komponentenorientierten Ansatz, der zugleich die Integration extern verfügbarer Komponenten und Anwendungen vorsieht.

1 Einleitung

Die Entwicklung webbasierter Informationssysteme wird technologisch seit einigen Jahren durch eine Vielzahl unterschiedlicher Systeme wie z.B. Web-Editoren und insbesondere Content-Management-Systeme unterstützt. Der Fokus dieser Technologien liegt im Wesentlichen auf der Gestaltung und flexiblen Erzeugung der grafischen Präsentation von Inhalten, dem Einstellen und Verwalten der Inhalte innerhalb eines Redaktionsprozesses sowie der Anbindung an Datenbanken und existierende Applikationen. Im Unterschied zu konventionellen Applikationen sind webbasierte Anwendungen oft einer ständigen Überarbeitung sowohl hinsichtlich ihrer Struktur als auch ihrer Inhalte unterworfen. Darüber hinaus bedeutet die Erneuerung des Informationsangebotes, z.B. aufgrund der Einbindung externer Dienste oder der Änderung der Navigationsstruktur, oft eine weitgehende Neugestaltung der Anwendung. Existierende Entwicklungsmethoden und Modellierungstechniken sind für diese Situationen oft nicht geeignet und kommen entsprechend selten zum Einsatz.

Neben content-intensiven, auf Informationsvermittlung ausgerichteten Applikationen gewinnen zunehmend auch webbasierte Business-Applikationen mit stark strukturierten Informationstypen und entsprechender Anwendungslogik an Bedeutung und treten oft in Mischformen mit inhaltsorientierten Systemteilen auf. Die Mischung von Informationsobjekten unterschiedlichen Strukturierungsgrades, eine hohe Veränderungsdynamik und die Einbeziehung externer Informationskomponenten und -dienste sind deshalb charakteristische Eigenschaften heutiger Web-Anwendungen. Entwurfstechniken und Entwicklungsprozesse müssen gleichermaßen Antworten auf Fragestellungen im Zusammenhang mit dieser veränderten Situation bieten und gleichzeitig dem stark gewachsenen Spektrum an unterschiedlichen Gestalter- und Entwicklerrollen gerecht werden. Insbesondere kleine und mittlere Unternehmen sind mit der Einführung und der ständi-

gen Anpassung von Standardmethoden überfordert, mit entsprechenden Konsequenzen für die Produktivität, Wartbarkeit und Qualität der resultierenden Anwendungen.

2 Vorgehensweisen zur Modellierung von Web-Anwendungen

Der Einsatz systematischer Entwurfsmethoden in der Entwicklung von Web-Anwendungen ist bislang wenig verbreitet. Untersuchungen wie Barry & Lang (Barry & Lang, 2001) zeigen, dass besonders im Medienbereich spezifische Methoden kaum eingesetzt werden, und falls doch, diese einen geringen Standardisierungsgrad aufweisen. Selbst Standardmethoden wie UML (Rational, 1997) werden bislang nur selten im Web-Bereich angewendet.

Stattdessen erfolgt die Entwicklung von Web-Sites zumeist ad-hoc und ohne einen systematischen Ansatz. Zahlreiche Tools wie z.B. HTML-Editoren, Database Publishing Wizards, Web Site Managers und Web Form Editoren unterstützen eine „Quick and Dirty"-Vorgehensweise, deren Einsatz zu einem hohen Wartungsaufwand bzgl. der erstellten Web-Sites führt. Darüber hinaus stellen die genannten Werkzeuge einen Mangel an Mechanismen zur Qualitätskontrolle und -sicherung dar, was in Verbindung mit fehlenden Möglichkeiten zur Wiederverwendung einzelner Teilbereiche des Entwurfs zu einer kompletten Neugestaltung der Web-Anwendung führen kann.

Verschiedene Methodenansätze wurden für spezifische Aspekte der Web-Entwicklung vorgeschlagen. Einige dieser Methoden beziehen sich auf die Modellierung der hypermedialen Struktur von Websites, wie HDM (Garzotti et al., 1993), RMM (Isakowitz & Stohr, 1995), OOHDM (Schwabe & Rossi, 1994) sowie die UML-Erweiterungen zur Modellierung Hypermedialer Systeme (Baumeister et al., 1999). Allerdings sind diese Methoden meist schlecht integrierbar mit anderen Entwicklungsperspektiven und haben keine ausreichende Werkzeugunterstützung. Zudem fehlt diesen Methoden einerseits die Möglichkeit zur rechtzeitigen Evaluation von Effizienz, Navigierbarkeit und Darstellung einer Web-Anwendung anhand eines generierten Prototyps sowie andererseits die konzeptionelle Unterstützung sich dynamisch verändernder Begriffswelten und Inhalte. Ansätze im Bereich Komponentenverwendung und -modellierung umfassen z.B. das Capsule-Konzept (Selic, 1998) in UML-RT oder Fragen des Konsistenzmanagements (Engels et al., 2002). Nutzer- und aufgabenbezogene Modellierungsmethoden wurden von Szwillus und Bomsdorf (Szwillus & Bomsdorf, 2002) vorgeschlagen. Die genannten Methoden und Modellierungsverfahren decken allerdings jeweils nur Teilaspekte der komplexen Analysesituation bei Web-Anwendungen ab und sind nicht in den Kontext weborientierter Entwicklungsumgebungen eingebettet.

3 Ontologiebasierte Vorgehensweise zur Modellierung komponentenorientierter Web-Anwendungen

In diesem Beitrag werden Methoden und Modelle vorgestellt, die den Aufbau von Web-Applikationen systematisieren und den gesamten Entwicklungsprozess bis hin zur Wartung und Pflege von Web-Anwendungen unterstützen. Grundlage der Methodologie ist ein Methodenverbund, in dem Konzepte und Techniken für die wesentlichen Modellierungsaspekte enthalten sind. Dieser Methodenverbund besteht zum einen aus verschiedenen Modellierungstechniken für die einzelnen Schritte des Entwurfsprozesses einer Web-Anwendung und andererseits aus softwarebasierten Werkzeugen, die den Erstellungsprozess teilautomatisiert unterstützen. Im Unterschied

zu existierenden Methodologien werden alle Entwurfsbereiche einer Web-Anwendung berücksichtigt und gleichzeitig durch die Möglichkeit, vom Benutzer erstellte Web-Modelle prototypisch darzustellen, Mechanismen zur Evaluation der Effizienz von Struktur, Navigation und Präsentation zur Verfügung gestellt.

Ausgangspunkt der in diesem Beitrag vorgestellten Vorgehensweise ist eine ontologische Beschreibung des Informationsbestandes innerhalb eines *konzeptuellen Modells*, die als Grundlage für den systematischen Entwurf einer Navigations- und Sichtenstruktur in Form eines *Navigationsmodells* bzw. *Sichtenmodells* dient und die vorhandenen Konzepte und Relationen der zugrunde liegenden Ontologie (Lenat & Guha 1990) als Themenbausteine einbezieht. Die Entwicklung dieser Ontologien kann entweder manuell erfolgen oder durch automatisierte Analysen von Ressourcenkorpora unterstützt werden.

Durch die Konzeption und Entwicklung sog. navigationaler Klassen können wiederkehrende Navigationsmuster modelliert und bei der Implementierung der Modelle angewendet werden. Eine wichtige Eigenschaft der navigationalen Klassen ist die automatisierte Zuordnung von Informationsobjekten zu einzelnen Themen der Navigationsstruktur unter Berücksichtigung der unterschiedlichen Sichten. Des Weiteren lassen sich verteilt bereitgestellte Dienste wie z.B. Web Services in Form von funktionalen Komponenten integrieren. Dies erfordert in der Modellierungsphase zunächst eine abstrahierte Beschreibung solcher Komponenten. In der Konfigurationsphase können auf diese Art und Weise die entsprechenden Web Services aus einem UDDI-Verzeichnis (http://uddi.org) ausgewählt und integriert bzw. zu einem späteren Zeitpunkt ausgetauscht werden.

Der Aufbau der Site- und Navigationsstruktur einer Web-Anwendung erfolgt anhand verschiedenartiger Komponenten. Die Webstruktur wird mit Hilfe von Containern und Partitionen erstellt, die die Eigenschaft besitzen, dass sie zur Darstellung ineinander eingebettet werden können, so dass auch verschachtelte Webseiten darstellbar sind. Erweitert wird der Methodenverbund durch Verfahren, die dem Entwickler einer Web-Anwendung Möglichkeiten zur Modellierung und Einbindung kontextabhängiger Informationsangebote bieten. Hierzu werden dem Modell sog. Sichtenklassen zur Verfügung gestellt, die eine rollen- bzw. lokalitätsabhängige Definition sichtbarer Attribute ermöglichen, auf die zur Laufzeit von einem Benutzer bzw. von einem bestimmten Ort aus zugegriffen werden darf.

Der Entwurfsprozess einer Web-Anwendung setzt sich demnach aus folgenden Schritten zusammen: Entwurf des Metadatenmodells (Themenstruktur), Aufbau des Navigationsmodells, formale Analyse der Benutzeranforderungen, Ableiten der Sichtenstruktur zur Modellierung personalisierter bzw. rollenbasierter Darstellungen auf der Grundlage des Navigationsmodells, Konfiguration externer Dienste und Definition der graphischen Darstellung und Interaktion (vgl. Abbildung 1).

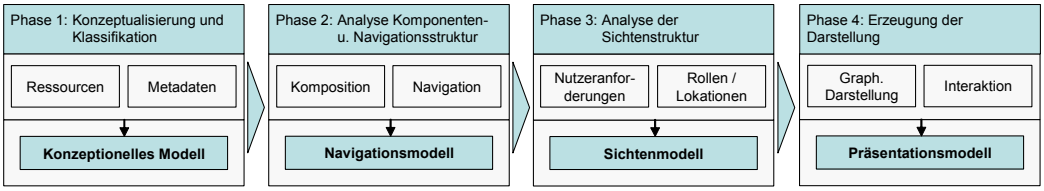


Abbildung 1: Strukturierung der Vorgehensweise

3.1 Konzeptionelles Modell

Die Ressourcen, die als Instanzen in Form von Dokumenten, Aufgaben- und Prozessbeschreibungen etc. vorliegen, dienen als Grundlage für die Erstellung des konzeptionellen Modells. Dieses

setzt sich aus einer übergeordneten Themenstruktur, bestehend aus Konzepten und zugehörigen Assoziationen, sowie der Zuordnung von Instanzen aus der Ressourcenmenge zusammen. Das konzeptionelle Modell stellt die Konzeptualisierung des Informationsbestandes dar und ermöglicht in Form einer Ontologie die Zugriffsmöglichkeit auf die instanziierten Inhalte. Die Herleitung der Themenstruktur kann prinzipiell auf zwei unterschiedliche Arten erfolgen. Zum einen können solche Begriffssysteme manuell erarbeitet und in einem weiteren Schritt den darin enthaltenen Themen die instanziierten Inhalte zugewiesen werden. Zum anderen besteht die Möglichkeit, die vorhandenen Dokumente automatisch zu klassifizieren. Hierzu wird eine Cluster-Struktur auf der Grundlage vorhandener Dokumente, die paarweise durch ein Ähnlichkeitsmaß beschrieben werden, erstellt. Dieses Ähnlichkeitsmaß kann Kriterien wie beispielsweise häufige gemeinsame Referenzen oder textliche Gemeinsamkeiten abbilden. Die erzeugte Ontologie stellt eine Navigationsmöglichkeit über Themen zur Verfügung, d.h. über die Assoziationen zwischen den Themen ist es möglich, innerhalb der Ontologie zu navigieren bzw. diese zu explorieren. Diese Form der Navigation unterscheidet sich jedoch gegenüber der in einer Web-Anwendung definierten Navigation, deren Navigationsrelationen bzw. Links nicht den Assoziationen innerhalb der Ontologie entsprechen müssen. Allerdings lassen sich mit Hilfe einer Ontologie explorative Zugriffe innerhalb der Web-Applikation realisieren und darüber hinaus Navigationsmöglichkeiten auf dynamische Mengen und Hierarchien erweitern. Komponentenentwurf und -strukturierung

3.2 Komponentenentwurf und -strukturierung

Die Themenstruktur des konzeptionellen Modells bildet ein semantisches Netz über die Ressourcen. Im Folgenden geht es um die Strukturierung und Positionierung der Inhalte sowie die Festlegung der Navigationsstruktur. Hierzu werden einzelne Elemente einer Web-Anwendung in Komponenten beschrieben, die in Form eines Kompositionsmodells zusammengesetzt werden. Struktur und Aufbau lassen sich unter Verwendung von Containern und Partitionen modellieren, die in der späteren Web-Applikation z.B. durch Frames realisiert werden können. Zusammenhänge zwischen den einzelnen Komposita werden anschließend durch das Navigationsmodell festgelegt.

3.2.1 Definition der Komponenten

Im Folgenden werden die zur Modellierung von Web-Sites notwendigen Komponenten definiert.

Atomare Komponenten und Klassen

Atomare Komponenten sind konkrete Objekte, die nicht weiter zerlegt werden können. Hierzu zählen beispielsweise Texte, Bilder, Multimediaobjekte, externe Referenzen etc. Atomare Komponenten werden als Instanzen ihrer zugehörigen *atomaren Klasse* innerhalb des konzeptuellen Modells betrachtet und dienen in erster Linie zur einheitlichen Betrachtung der verschiedenen Objekte des Navigationsmodells.

Container und Partitionen

Die Elemente, die eine einzelne Webseite beschreiben, werden in sog. *Containern* gehalten. Die Container besitzen die Eigenschaft, dass sie ineinander eingebettet werden können, so dass auch verschachtelte Webseiten darstellbar sind. Das Navigationsmodell, das die Grundlage für die spätere Darstellung der Web-Applikation bildet, beruht auf einem HiGraphen-ähnlichen Ansatz (Harel et al., 1987). Die graphische Repräsentation des Navigationsmodells betrachtet im Wesentlichen die Möglichkeiten der Anordnung von Containern und die Übergangsrelationen zwischen den einzelnen Containern. Durch die Möglichkeit, Container ineinander einzubetten entstehen *Ebenen*. Mehrere Subcontainer innerhalb eines Containers, die auf der gleichen Ebene liegen, können entweder gemäß einer XOR-Verknüpfung dargestellt werden, d.h. in diesem Fall ist im-

mer genau ein Subcontainer sichtbar, oder zur gleichzeitigen Darstellung in einzelne *Partitionen* unterteilt werden (vgl. Abbildung 2).

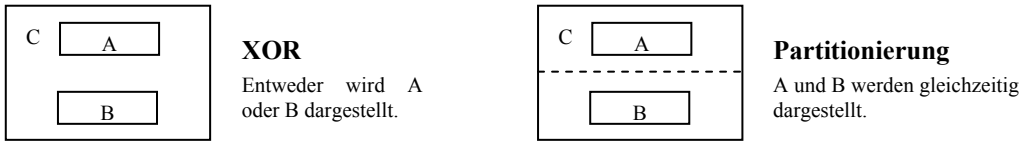


Abbildung 2: Container und Partitionen

Atomare Klassen werden Containern zugeordnet. Container, denen eine Klasse zugeordnet wurde, können nicht mehr weiter unterteilt werden. Liegen mehrere Subcontainer auf gleicher Ebene eines Containers, so muss festgelegt werden, welcher Subcontainer zunächst sichtbar ist. Dazu wird gemäß Abbildung 3 (a) ein Startzustand definiert. Sowohl Container als auch Partitionen definieren den Zielbereich, in dem die Inhalte angezeigt werden.

Platzhalter

Mit Hilfe von Platzhaltern können Teilbereiche der Web-Anwendung zu besseren Überschaubarkeit schematisch dargestellt werden. Eine detaillierte Modellierung des durch einen Platzhalter nur grob dargestellten Bereiches kann an einer anderen Stelle erfolgen (vgl. Abbildung 3 (b)).

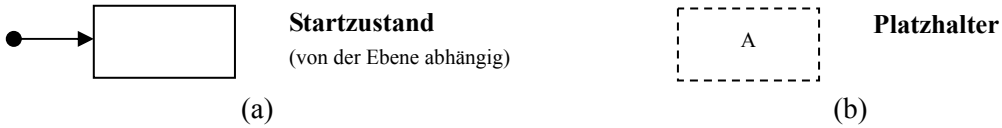


Abbildung 3: (a) Festlegung des Startzustandes; (b) Platzhalter für einen an anderer Stelle definierten Container

Ereignisse und Links

Eine Veränderung der auf einer Webseite dargestellten Inhalte wird durch ein Ereignis hervorgerufen, das typischerweise von einem Benutzer durch Interaktionen ausgelöst wird. Zusätzlich können Ereignisse automatisch verursacht werden, beispielsweise durch Überschreitung einer Zeitspanne oder durch vorangegangene Ereignisse. Hervorgerufen durch ein Ereignis wird ein *Link* aktiviert, was zu einer Veränderung der Darstellung in einer Webanwendung führt. Der Zielbereich eines Links definiert den Container bzw. die Partition, in der die Änderungen durchgeführt werden sollen. Der aktuelle Kontext wird in Form der *Ursprungsinstanz* ebenfalls in einem Link mitgeführt. Die Ursprungsinstanz ist die durch den Link repräsentierte Instanz einer Klasse. Die Übergabe des Kontextes ist notwendig, um Abhängigkeiten der neu aufzubauenden Informationen mit dem aktuellen Kontext zu bedienen. Da der Zielbereich nicht durch eine atomare Komponente definiert werden kann, müssen dem Link zusätzlich die *Zielklasse* sowie die zugehörige Relation zwischen der Ursprungsinstanz und der Zielklasse bekannt sein. Anhand dieser Informationen ist es möglich, die darzustellende Instanz bzw. Instanzmenge zu ermitteln. Handelt es sich bei der Zielklasse um eine atomare Klasse, ist die Ursprungsinstanz die Komponente, in der der Link definiert wurde. Bei dieser Komponente handelt es sich entweder um einen Container oder eine Partition. Somit ist es möglich, innerhalb der Webapplikation auf beliebige freie Objekte zu verweisen, die nicht in der Ontologie enthalten sind.

In der graphischen Repräsentation des Navigationsmodells definieren Links den Übergang von einer Komponente zu einer anderen. Entsprechend Abbildung 4 (a) wird ein Link mit einem beschrifteten Pfeil dargestellt. Soll nach der Ausführung des Links im Sinne eines Broadcast ein

weiteres Ereignis ausgelöst werden, wird der zusätzliche Link mit in die Beschriftung aufgenommen (vgl. Abbildung 4 (b)).

Popup-Fenster

Stellt der Zielbereich eines Links ein Popup-Fenster dar, wird im Link zusätzlich spezifiziert, welche Aktion bzgl. des Popup-Fensters ausgeführt werden soll (vgl. Abbildung 5).

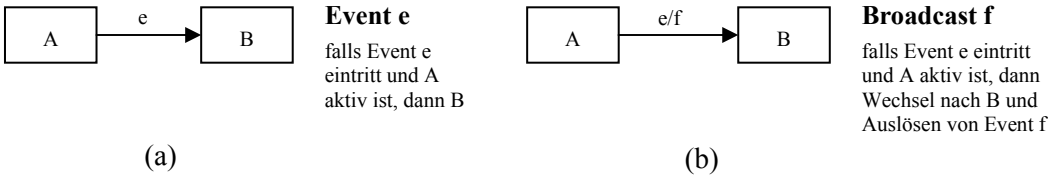


Abbildung 4: (a) Aufruf von B aufgrund von Link e; (b) Aktivierung des Link f nach Aufruf von e

Navigationale Klassen

In einem Zielbereich kann entweder eine einzelne Instanz einer Klasse dargestellt werden oder, falls zu einer Klasse mehrere Instanzen vorhanden sind, eine Übersicht der vorhandenen Instanzen, mit der Möglichkeit, an definierter Stelle die Instanzen im Einzelnen zu betrachten. Zu die

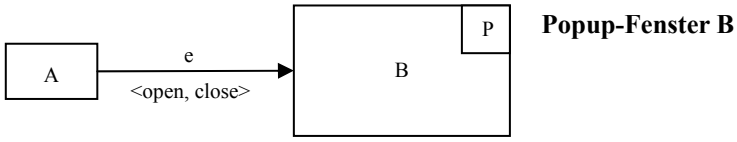


Abbildung 5: Popup-Fenster

sem Zweck werden *navigationale Klassen* definiert, die die Navigation auf einer Menge von Instanzen mit Hilfe einer Übersicht ermöglichen. Die Art und Weise, in der die Übersicht dargestellt wird, kann entweder von der navigationalen Klasse selbst bestimmt werden oder ist im zugrunde liegenden Navigationsmodell bereits festgelegt. Im letzteren Fall wird eine Darstellungsart aus einer Menge vordefinierter Navigationstypen gewählt. Zu dieser Menge gehören gemäß Abbildung 6 Typen wie beispielsweise Baum, Index, Sequenz etc. (vgl. Ziegler, 1997). Wird die Darstellungsart von der navigationalen Klasse dynamisch ermittelt, wird hierzu die Kardinalität der darzustellenden Instanzmenge herangezogen. Die Menge der zu repräsentierenden Instanzen kann mit Hilfe der Instanz der Ausgangsklasse und der Ursprungsrelation beschrieben werden. Die Auswahl der angezeigten Attribute wird in der Attributmenge der Zielinstanz festgelegt und entsprechend einer Sortierungsfunktion sortiert.

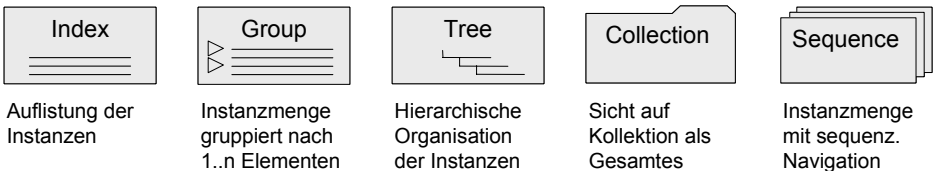
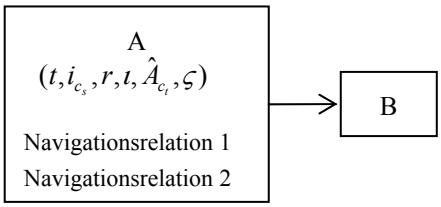


Abbildung 6: Auswahl unterschiedlicher Typen navigationaler Klassen

Die graphische Darstellung der navigationalen Klasse besteht aus zwei Komponenten, die mit einer Navigationsrelation miteinander verbunden sind. Bei der Navigationsrelation handelt sich

um einen parametrisierbaren Link, der durch die einzelnen Instanzen in der navigationalen Klasse bei dem entsprechenden Aufruf initialisiert wird (vgl. Abbildung 7). Er ist durch eine offene Pfeilspitze gekennzeichnet.



Navigationale Klasse A

Zustandsänderung in B bei Interaktion innerhalb von A

t = Typ des Navigators (Index, Tree, ...)
i_{c_s} = Ursprungsinstanz
r = Ursprungsrelation
ι = Instanzfunktion
$\hat{A}_{c_t}$ = Attributmenge
ς = Sortierungsfunktion

Abbildung 7: Graphische Darstellung navigationaler Klassen

3.2.2 Kompositionsmodell

Das Kompositionsmodell definiert die Zusammensetzung beliebiger Komponenten zu einzelnen Web-Seiten. Es spezifiziert die mit Hilfe von Containern und Partitionen die Struktur einer Web-Seite und die seiteninterne Navigation.

3.2.3 Navigationsmodell

Das Navigationsmodell setzt sich aus den Kompositionsmodellen der einzelnen Web-Seiten und einer zusätzlichen übergeordneten Navigationsstruktur zusammen. Diese legt die Navigation zwischen den einzelnen Web-Seiten fest.

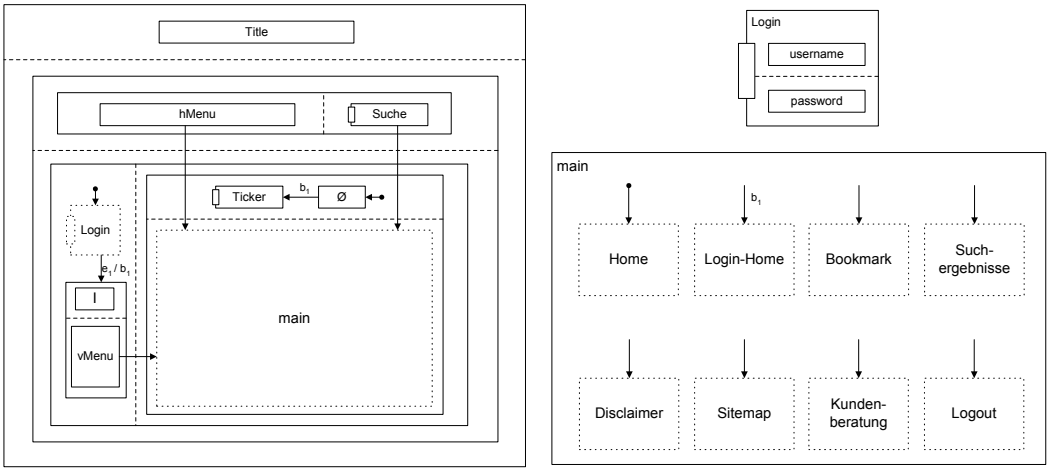


Abbildung 8: Navigationsmodell (Auszug aus einem Portal)

Abbildung 8 zeigt beispielhaft das Navigationsmodell einer Portalseite, das den Rahmen der entsprechenden Anwendung darstellt. In der gesamten Web-Applikation findet sich im oberen Bereich eine Titelzeile, unmittelbar darunter liegt eine Menüleiste mit einem rechts angehefteten Suchfeld in Form einer Applikation. Unterhalb dieser beiden Komponenten befindet sich eine weitere Partition. Diese enthält im Startzustand links zunächst ein Login-Feld, das nach erfolgreicher Anmeldung durch ein vertikales Menü mit einem darüber liegendem Bild („I“ steht hier für

„Image“) ersetzt wird. Rechts davon wird im oberen Bereich zunächst nichts angezeigt („Ø“ steht hier für die leere Menge), erst nachdem die Anmeldung abgeschlossen ist, wird der Ticker aktiviert. Der untere Bereich stellt den Hauptbereich der Anwendung dar. Er ist gleichzeitig der Zielbereich für die beiden Menüleisten und das Suchfeld. Die verschiedenen Komponenten, die in diesem Bereich dargestellt werden, können nun an anderer Stelle genauer spezifiziert werden.

3.3 Sichtenmodell

Mit dem Ziel, ein den jeweiligen Anforderungen des Nutzers angepasstes Informationsangebot zu generieren, werden auf die vorhandenen Informationen mit Hilfe von Sichtenklassen verschiedene Sichten erzeugt. Diese haben die Aufgabe, definierte Eigenschaften einer Instanz einer Klasse unter Berücksichtigung von Nebenbedingungen darzustellen. Die Modellierung der Kontextfaktoren in den Sichtenklassen erfolgt auf Basis des Komponenten- bzw. Navigationsmodells. Für jede darin enthaltene Modellklasse können Bedingungen hinzugefügt werden, die über ihre Sichtbarkeit in Abhängigkeit des Kontextes entscheiden. Z.B. ist es möglich, die Darstellung ganzer Teilbereiche für eine bestimmte Person, Benutzerrolle oder ein spezielles Endgerät zu verhindern. Eine Sichtenklasse besteht aus zwei Teilen, einer Zuordnungsangabe, die den Benutzer bzw. die Rolle festlegt, und einem Bedingungsteil, der einen beliebigen booleschen Ausdruck enthält. Dieser muss erfüllt sein, um den in der zugehörigen Modellklasse festgelegten Inhalt darzustellen. Prinzipiell ist es möglich, für verschiedene Rollen bzw. Benutzer jeweils eine Sichtenklasse zu erstellen. Dadurch lassen sich rollenbasierte bzw. benutzerspezifische Darstellungen innerhalb der Web-Applikation realisieren. Abbildung 9 zeigt die graphische Beschreibung einer Sichtenklasse.

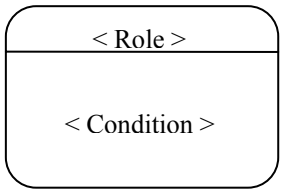


Abbildung 9: Beispiel zweier Sichtenklassen (Besucher, Investor)

Das *Sichtenmodell* setzt sich aus einer Menge von Sichtenklassen für jede einzelne Modellklasse zusammen. Sollen nur wenige Sichten modelliert werden, können die Sichtenklassen neben den zugehörigen Modellklassen positioniert und durch eine Linie verbunden werden. Im anderen Fall ist das Sichtenmodell als eigenständiges graphisches Modell zu betrachten, das alle Modellklassen auflistet und diese mit den zugehörigen Sichtenklassen verbindet.

3.4 Präsentationsmodell

Das Präsentationsmodell dient der Transformation des konzeptuellen Schemas auf eine lower-level Präsentation und enthält im Wesentlichen die zur graphischen Darstellung notwendigen Informationen. Hierzu zählen genaue Angaben über die Position und die graphische Gestaltung der darzustellenden Inhalte und Interaktionsflächen. Diese Informationen werden unter Verwendung von Stylesheets den einzelnen Sichtenklassen zugeordnet und zur Laufzeit abgerufen. Neben diesen Informationen zur graphischen Gestaltung der einzelnen Komponenten können weitere Angaben im Präsentationsmodell festgelegt werden. Das Präsentationsmodell stellt hierzu eine offene Struktur zur Verfügung, die es erlaubt, beliebige Merkmale der graphischen Gestaltung zu

bestimmen. Beispielsweise lassen sich zusätzlich Angaben über die zu verwendende Schriftart, die Schriftgröße etc. festlegen.

4 Umsetzung

Mit Hilfe parametrisierbarer Komponenten, die auf der Grundlage definierter Metadaten- und Navigationsmodelle zur Laufzeit dynamische und kontextsensitive Navigationsstrukturen erstellen, lassen sich die vom Benutzer erstellten Modelle in einer Laufzeitumgebung für Test- und Verifikationsmöglichkeiten realisieren. Hierzu werden Verfahren zur automatisierten Generierung von Navigationsstrukturen anhand der entwickelten Modelle konzipiert und implementiert, deren hohe Wiederverwendbarkeit die Umsetzung einer modellierten Web-Anwendung in ein lauffähiges System ermöglichen. Abbildung 10 zeigt die Architektur einer Modellierungs- und Laufzeitumgebung, die derzeit in Form eines Prototyps implementiert wird. Innerhalb einer Werkzeug-unterstützten Modellierungsumgebung werden unter Zuhilfenahme der ontologischen Beschreibung der Informationsinhalte die einzelnen Modelle schrittweise erstellt. Der Zugriff auf die formale Beschreibung der Modelle erfolgt durch die Laufzeitumgebung, die die verschiedenen Kontextfaktoren wie beispielsweise Ort und Zeit beim Zugriff auf die Ontologie und die darin referenzierten Inhalte berücksichtigt. Die Architektur fokussiert vor allem die systemunterstützte und weitestgehend automatisierte Verarbeitung der zur Modellierungszeit erstellten Modelle, um darauf aufbauend die Web-Anwendung zu generieren.

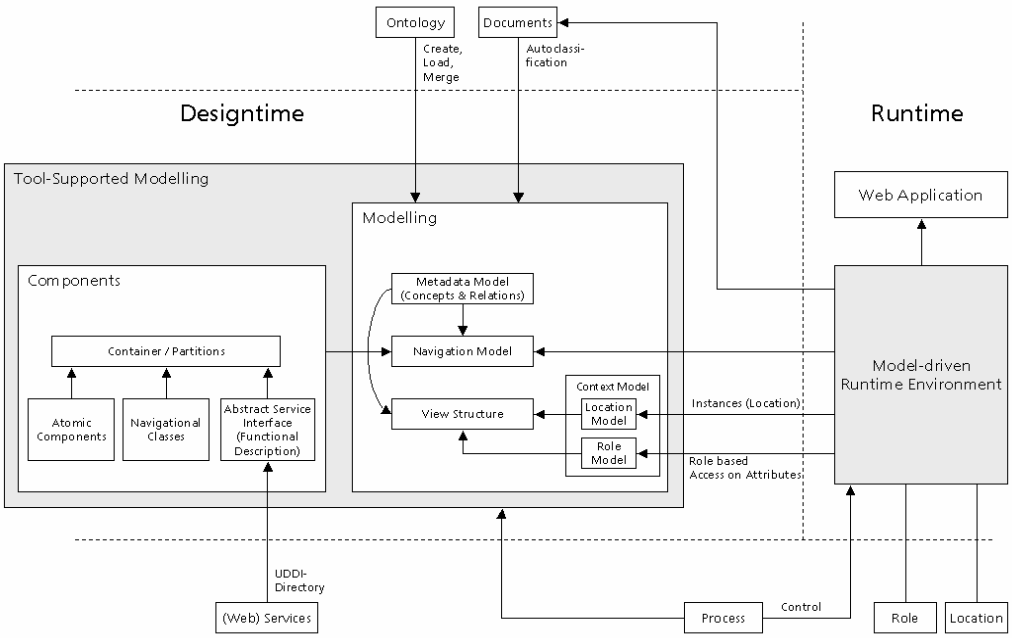


Abbildung 10: Architektur der Modellierungs- und Laufzeitumgebung

5 Ausblick

Dieser Beitrag zeigt eine Vorgehensweise zur systematischen und durchgängigen Entwicklung von Web-Anwendungen. Besonderes Augenmerk wurde bei der Konzeption auf eine möglichst leichte Verständlichkeit der Modelle und insbesondere auf Unterstützungsmöglichkeiten durch softwarebasierte Werkzeuge gelegt, die den Benutzer bei der Erstellung und Verifikation der Modelle weitestgehend entlasten sollen.

Der komponentenorientierte Ansatz ließe sich in einem weiteren Schritt auch auf im Web bereitgestellte Modellkomponenten ausweiten, die eine einheitliche Beschreibung aufweisen. Für den Entwickler einer Web-Anwendung würde sich daraus der Vorteil bieten, für bestimmte Teile seiner Anwendung auf bereits existierende eigen- oder fremdentwickelte Lösungen zurückgreifen zu können.

6 Literaturverzeichnis

Barry, C.; Lang, M. (2001): *A Survey of Multimedia and Web Development Techniques and Methodology Usage*. IEEE Multimedia. 8(3), 52-60

Baumeister, H.; Koch, N.; Mandel, L. (1999): *Towards a UML extension for hypermedia design.* In *UML'99*, Fort Collins, USA

Engels, G.; Küster, J.M.; Heckel, R. (2002): *Towards Consistency-Preserving Model Evolution.* In Proceedings ICSE Work-shop on Model Evolution, Florida, USA

Garzotti, F.; Paolini, P.; Schwabe, D. (1993): *HDM – A model-based approach to hypermedia application design.* ACM-Transactions on Information Systems, 11/1

Harel, D.; Pnueli, A.; Schmidt, J. P.; Sherman, R. (1987): *On the formal semantics of statecharts.* Proc. 2nd. IEEE Symposium on Logic in Computer Science, Ithaca, N.Y.

Isakowitz, T.; Stohr, E. A.; Balasubramanian, P. (1995): *RMM: A Methodology for Structured Hypermedia Design.* Communications of the ACM, 38/8:34-44

Lenat, D. B.; Guha, R. V. (1990): *Building large knowledge-based systems.* Addison-Wesley

Rational Software Corporation (1997): *UML Notation Guide*. Version 1.1, http://www.rational.com

Schwabe, D.; Rossi, G. (1994): *From Domain Models to Hypermedia Applications: An Object-Oriented Approach.* International Workshop on Methodologies for Designing and Developing Hypermedia Applications, Edinburgh

Selic, B. (1998): *Using UML for modeling complex real-time systems.* In F. Mueller und A. Bestavros (Hrsg.): Languages, Compilers and Tools for Embedded Systems, Band 1474 von Lecture Notes in Computer Science, Seiten 252-262. Springer Verlag

Szwillus, G.; Bomsdorf, B. (2002): Models for Task-Object-Based Web Site Management, Proceedings DSV-IS 2002, Rostock

Ziegler, J. E. (1997): *ViewNet - Conceptual design and modelling of navigation*. In S. Howard, J. Hammon & G. Lingaard (Eds.), Human-Computer Interaction: Interact'97, London: Chapman & Hall