3. Modellierung von Web-Anwendungen

Wieland Schwinger, Nora Koch

Zusammenfassung

Die Modellierung von Web-Anwendungen ist in der Praxis (noch) nicht weit verbreitet. Dabei setzt sie den bei der Ad-hoc-Entwicklung von Web-Anwendungen immer wieder auftretenden Problemen, wie zum Beispiel unzureichende Erfüllung der Anforderungen, mangelhafte Spezifikation und fehlende Dokumentation des Systems, eine modellbasierte Vorgehensweise entgegen. Die dabei entwickelten Modelle stellen eine solide Ausgangsbasis für die Implementierung einer Web-Anwendung dar. Diese Modelle berücksichtigen statische und dynamische Aspekte der Content-, Hypertext- und Präsentationsebene einer Web-Anwendung. Während das Content-Modell ähnlich dem einer Nicht-Web-Anwendung ist, stellt das Hypertextmodell die verschiedenen Navigationsmöglichkeiten auf Basis der vorhandenen Information dar. Das Präsentationsmodell bildet die Hypertextstrukturen auf Seiten und deren Verlinkung ab. Der Einbeziehung von Kontextinformation wie Benutzer, Zeit, Ort und benutztes Endgerät sowie der daraus abzuleitenden Anpassung der entsprechenden Web-Anwendung wird zunehmend während der Modellierung Beachtung geschenkt - eine Folge der immer wichtiger werdenden ubiquitären Web-Anwendungen. Das Spektrum existierender Methoden und Werkzeuge für die Modellierung von Web-Anwendungen und ihre Schwerpunkte werden aufgezeigt, um dem Leser Hilfestellung bei der Auswahl einer geeigneten Modellierungsmethode zu geben. Code-generierende Werkzeuge, die die Verwendung unterschiedlicher Endgeräte berücksichtigen, stellen eine wesentliche Unterstützung für die modellbasierte Entwicklung und die jeweilige Modellierungsmethode dar.

3.1. Einleitung

Möchte man eine Hundehütte bauen, so sind zwei geschickte Hände, das notwendige Material, sowie etwas Werkzeug ausreichend, um je nach persönlicher Kreativität, flink darauf loshämmernd und sägend ein

ansprechendes Ergebnis zu erzielen. Niemand würde sich allerdings mit derselben naiven Unbeschwertheit an den Bau eines Wolkenkratzers machen - das Ergebnis wäre sicherlich fatal! Was jedem beim Bau eines Wolkenkratzers einleuchtend ist, wird beim Bau komplexer Web-Anwendungen oftmals ignoriert. Sollen komplexe Web-Anwendungen entwickelt werden, sind eine systematische Vorgehensweise und eine Spezifikation der zu erstellenden Web-Anwendung in Form von Modellen empfehlenswert.

Dieses Kapitel beschäftigt sich mit der modellbasierten Entwicklung von Web-Anwendungen. Dazu wird zunächst in Unterkapitel 3.2 ein Einblick in die allgemeinen Grundlagen der Modellierung gegeben, dem in Unterkapitel 3.3 eine Diskussion der Besonderheiten der Modellierung von Web-Anwendungen folgt. Danach werden, ausgehend von einer Anforderungsbeschreibung, die unterschiedlichen Modelle einer Web-Anwendung beschrieben. Ein vereinfachtes web-basiertes Konferenzbegutachtungssystem dient dazu als durchgängiges Beispiel. Abschließend wird ein Überblick über existierende Methoden und Werkzeuge zur Modellierung von Web-Anwendungen gegeben. Ein Ausblick über zukünftige Entwicklungstrends im Bereich der Modellierung von Web-Anwendungen bildet den Abschluss.

3.2. Grundlagen der Modellierung

In ingenieurswissenschaftlichen Disziplinen werden Modelle, die der Realisierung vorausgehen, mit großem Erfolg angewandt, um Komplexität zu reduzieren, Entwurfsentscheidungen zu dokumentieren und Kommunikation innerhalb von Projektteams zu erleichtern. Ziel der Modellierung ist, eine für die Realisierung ausreichend detaillierte Spezifikation des zu erstellenden Systems bereitzustellen. Ergebnis der Modellierung sind Modelle, welche in vereinfachter und möglichst lesbarer Art und Weise die relevanten Aspekte des zu erstellenden Systems repräsentieren.

Auch in der Informatik wird seit längerem bei der Entwicklung von Software auf die Technik der Modellierung zurückgegriffen. Gegenstand der Modellierung ist dabei die zu erstellende Anwendung. Der Umfang der Modellierung lässt sich dabei wie in Abb. 3.1 dargestellt entlang von drei orthogonalen Dimensionen aufspannen. Die erste Dimension umfasst traditionell die Ebenen der Anwendungslogik und der darauf aufsetzenden Benutzerschnittstelle im Sinne einer Kapselung des »was« und des »wie« einer Anwendung. Eine weitere Dimension bilden die Aspekte Struktur (d.h. die Objekte, deren Attribute und ihre Beziehungen untereinander) und Verhalten (d.h. Funktionen und Prozesse), sowohl der Anwendungslogik als auch der Benutzerschnittstelle. Da eine Anwendung nicht »in einem Wurf« entwickelt werden kann, sondern im Verlauf des Entwicklungsprozesses schrittweise

verfeinert und erweitert werden muss, bilden die Phasen die dritte Dimension der Modellierung von Anwendungen. Die Modellierung überführt die in der Anforderungsanalyse ermittelten Anforderungen an die Anwendung in die entsprechenden Analyse- und Entwurfsmodelle, die in weiterer Folge die Ausgangsbasis für die Implementierung sind.

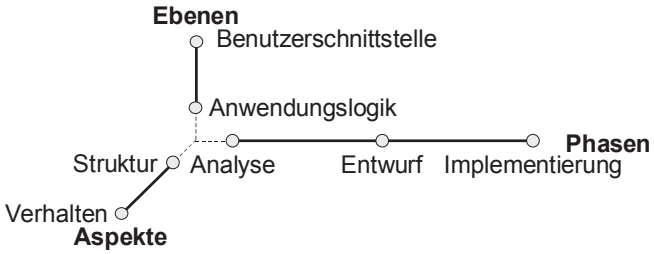

***Abb. 3.1** Anforderungen an die Modellierung von Softwareanwendungen*

Die Wurzeln der Modellierung liegen einerseits im Data Engineering und andererseits im Software Engineering. Dabei fokussiert Modellierung im Data Engineering historisch auf die strukturellen Aspekte, d.h. Datenaspekte einer Anwendung. Im Vordergrund stehen dabei die Informationseinheiten und deren Gruppierung und Beziehungen untereinander. Das wohl bekannteste Modell dazu ist das *Entity-Relationship Model (ER)* [Chen76]. Im Gegensatz dazu standen im Bereich des Software Engineering vor allem operationale Aspekte der Software im Vordergrund. Die entwickelten Modellierungsmethoden entstanden dabei vor allem aus den Anforderungen der Programmiersprachen. Objektorientierte Modellierung »vereinigt« beide Bereiche in dem sowohl Struktur als auch Verhalten gleichberechtigt betrachtet werden. Die wesentlichen Charakteristika der objektorientierten Modellierung sind eine ganzheitliche Vorgehensweise bei der Modellierung des Systems und das zentrale Konzept des Objekts, das Struktur und Verhalten umfasst.

Der objektorientierte Modellierungsstandard UML (Unified Modeling Language, [HiKa02, UML03]) ist eine Art Lingua Franca der objektorientierten Softwareentwicklung und Grundlage der meisten Modellierungsmethoden für web-basierte Anwendungen. Die unterschiedlichen Aspekte eines Softwaresystems werden in UML in Form von Modellen spezifiziert und graphisch durch verschiedene Diagramme repräsentiert. Dazu unterscheidet UML acht Diagrammarten. Das Anwendungsfalldiagramm spezifiziert in Form von Anwendungsfällen die Funktionalität des zu entwickelnden Systems aus Benutzersicht. Das Klassendiagramm beschreibt die strukturellen Aspekte des Systems in Form von Klassen, Objekten, Schnittstellen, Paketen und deren Beziehungen untereinander. Das Zustandsdiagramm beschreibt für jede Klasse das erlaubte Verhalten der Objekte, d.h. Instanzen, der entsprechenden Klasse. Die Interaktionsdiagramme (Sequenzdiagramm und Kollaborationsdiagramm) sowie das

Aktivitätsdiagramm beschreiben Verhaltensaspekte der Anwendung. Das Komponentendiagramm und das Verteilungsdiagramm schließlich zeigen für die Implementierung relevante Komponenten bzw. deren Verteilung im Netzwerk.

3.3. Besonderheiten der Modellierung im Web Engineering

Das Handwerkszeug der Modellierung von Web-Anwendungen ist im Grunde nicht neu, jedoch stellen Methoden für die Modellierung traditioneller Anwendungen nur in unzureichendem Maße Konzepte für die spezifischen Charakteristika von Web-Anwendungen zur Verfügung (siehe auch Besonderheiten von Web-Anwendungen in Kapitel 1). So existieren zum Beispiel keine entsprechenden Konzepte, um die hypertextuellen Verlinkungen einer Web-Anwendung in einem Modell abbilden zu können. Daher wurden in den letzten Jahren spezielle Ansätze für die Modellierung von Web-Anwendungen entwickelt, die sich den oben vorgestellten drei Dimensionen der Ebenen, Aspekte und Phasen zuordnen lassen.

3.3.1. Ebenen

Für die Modellierung von Web-Anwendungen müssen der Dokumentencharakter des Content sowie die nicht-lineare Navigation im Hypertext Berücksichtigung finden. Daher wird, wie in Abb. 3.2 gezeigt, im Gegensatz zur Modellierung traditioneller Anwendungen bei der Modellierung von Web-Anwendungen in drei Ebenen anstelle von zwei unterteilt: *Content,* die der Web-Anwendung zu Grunde liegende Information und Anwendungslogik, *Hypertext*, die Gliederung des Content in Knoten und Verweise zwischen den Knoten und *Präsentation*, die Gestaltung der Benutzerschnittstelle. Die meisten Methoden zur Modellierung von Web-Anwendungen folgen dieser Trennung in drei Ebenen [Frat99]. Dies ermöglicht Wiederverwendung, erlaubt Modellevolution und hilft die Komplexität zu reduzieren. Für einen definierten Content können unterschiedliche Hypertextstrukturen und Zugriffsmodelle spezifiziert werden, die den spezifischen Anforderungen unterschiedlicher Benutzergruppen und Endgeräte best möglich gerecht werden. Die Zielsetzung bei der Erstellung des Content-Modells ist die explizite Festlegung der Informationsstruktur, vergleichbar mit dem Datenbankschema in der Datenmodellierung, und die Eliminierung von Redundanzen. Somit bleibt die Struktur der Information unverändert, selbst wenn sich die Information häufig ändert. Um das Navigieren effizienter zu gestalten, wird im Gegensatz dazu auf der Ebene der Modellierung des Hypertexts manche Information redundant auf mehreren Knoten angeboten

und somit diese Information über mehrere Zugriffspfade auffindbar gemacht. Um einer möglichen Desorientierung entgegen zu wirken, und, um die kognitive Belastung des Benutzers möglichst gering zu halten, wird bei der Modellierung des Hypertexts auf wiederkehrende Navigationsmuster zurückgegriffen [Bern98]. Für die Modellierung der Präsentation wiederum stehen neben ästhetischen Aspekten auch eine einheitliche Präsentationsstruktur der Seiten im Vordergrund, um einen Wiedererkennungseffekt beim Benutzer zu erreichen.

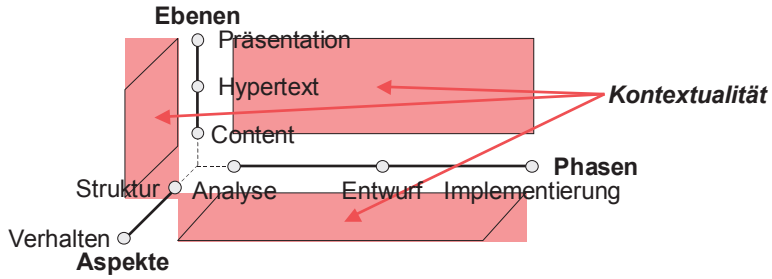

***Abb. 3.2** Anforderungen an die Modellierung von Web-Anwendungen*

Trotz der unterschiedlichen Zielsetzungen der einzelnen Ebenen, soll eine Abbildung der Ebenen auf einander möglich sein. Dazu ist ein explizites Definieren von Abhängigkeiten erforderlich. Auf diese Weise wird es zum Beispiel auch möglich, auf Basis ein und desselben Content verschiedene personalisierte Hypertext-Zugänge zu spezifizieren. Ein umfassendes Modell einer Web-Anwendung schließt alle drei aufgezeigten Ebenen ein, wobei allerdings der Schwerpunkt individuell für die zu erstellende Web-Anwendung variieren kann. Web-Anwendungen, die eine rein hypertextorientierte Benutzerschnittstelle auf eine große Datenmenge zur Verfügung stellen, werden wahrscheinlich einen Schwerpunkt bei der Modellierung der Informations- und Hypertextstruktur haben. Im Gegensatz dazu werden präsentationsorientierte Web-Anwendungen größere Ansprüche bezüglich des Präsentationsverhaltens haben, wie zum Beispiel Unternehmensportale und Online-Shopping Malls.

3.3.2. Aspekte

Der Anteil an Struktur und Verhalten, der in jeder der drei Ebenen Content, Hypertext und Präsentation Berücksichtigung finden muss, variiert in Abhängigkeit von der Art der zu realisierenden Web-Anwendung. Web-Anwendungen, die vor allem statische Information bereitstellen, weisen einen geringeren Anteil an Verhaltensmodellierung auf als hoch interaktive Web-Anwendungen, wie zum Beispiel e-Commerce-Anwendungen, die Suchfunktionalität, Bestellfunktionen, etc. bieten. Im Hinblick auf eine Abbildung und Integration der unterschiedlichen Ebenen empfiehlt sich die Verwendung eines einheitlichen Modellierungsformalismus für Struktur und

Verhalten, der allerdings an die spezifischen Anforderungen der einzelnen Ebenen angepasst sein muss.

3.3.3. Phasen

Es existiert noch kein Konsens in der Literatur über ein generelles Vorgehensmodell für die Entwicklung einer Web-Anwendung (siehe dazu auch Kapitel 10). In jedem Fall sollte die Reihenfolge bei der Modellierung der Ebenen nicht an eine bestimmte Vorgehensweise gebunden sein. Abhängig von der Art der Web-Anwendung sollte sowohl ein informationsgetriebenes Vorgehen, d.h. mit der Modellierung des Content beginnend, wie auch ein präsentationsgetriebenes Vorgehen, d.h. mit der Modellierung der Präsentationsaspekte der Anwendung beginnend, möglich sein. Präsentationsgetriebenes Vorgehen wird zum Beispiel in jenen Fällen empfehlenswert sein, in denen existierende Webseiten in eine Datenbank überführt werden sollen. Modellbasierte Entwicklung im Web Engineering widerspricht zu einem gewissen Grad der in Web-Projekten oftmals vorgefundenen Parallelität von Entwicklungsphasen, der Schnelllebigkeit von Entwicklungszyklen, die zum Teil auf unzureichend erforschte Anforderungen oder unausgereifte Realisierungsplattformen begründet sind, sowie dem Verlangen nach »agilen Methoden« (siehe dazu Kapitel 1). Dem setzt modellbasiertes Vorgehen die umfassende Spezifikation des Lösungsmodells bzw., falls vorhanden, die Möglichkeit der automatischen Generierung der (prototypischen) Web-Anwendung entgegen. Auch sichern Modelle im Gegensatz zu kurzlebigeren Softwarelösungen die Nachhaltigkeit der Lösungsidee.

3.3.4. Kontextualität

Aufgrund der technischen Machbarkeit kommt der Einbeziehung von Kontextinformation bei der Entwicklung von Web-Anwendungen eine entscheidende Bedeutung zu. Kontextualität beeinflusst alle drei Dimensionen der Web-Modellierung, da Content, Hypertext und Präsentation sowohl in Struktur und Verhalten entsprechend angepasst werden müssen, was bereits ausgehend von der Analysephase während des gesamten Entwicklungsprozesses berücksichtigt werden muss. Die Berücksichtigung von Kontextinformation wird daher als eigene Modellierungsdimension behandelt (siehe Abb. 3.2, [KPRS03]).

Da zur Zeit noch keine weit verbreitete Modellierungsmethode existiert, die alle aufgezeigten Dimensionen umfassend abdeckt (siehe dazu auch Unterkapitel 3.9), wird in diesem Kapitel auf UML als Notation zurückgegriffen, erweitert um einige Konzepte der UML-basierten Methode

UWE (UML-based Web Engineering) [KoKr02]. UWE bietet sich auf Grund der Einfachheit sowie der UML-Konformität an (siehe auch Methodenvergleich in Unterkapitel 3.9).

3.4. Ausgangspunkt der Modellierung

Anforderungen an eine Web-Anwendung können, wie in Kapitel 2 gezeigt, mit unterschiedlichen Techniken ermittelt, analysiert, beschrieben, überprüft und verwaltet werden. Als Modellierungstechnik für funktionale Anforderungen werden fast ausschließlich Anwendungsfälle verwendet, die sich graphisch in UML als Anwendungsfalldiagramm darstellen lassen und für eine detailliertere Beschreibung um Aktivitätsdiagramme ergänzt werden können. Die Gesamtfunktionalität einer Web-Anwendung wird als eine Menge von Anwendungsfällen modelliert, welche die Anforderungen an die Web-Anwendung aus Sicht der Akteure (Personen ebenso wie andere Systeme) beschreibt.

Eine Besonderheit der Anforderungsanalyse bei Web-Anwendungen ist die Navigationsfunktionalität, die dem Benutzer das Navigieren durch den Hypertext sowie das Auffinden von Knoten erlaubt. [BaGP01] empfiehlt hierzu die Trennung der funktionalen und navigationalen Anwendungsfälle in zwei separate Modelle. Ein anderer Ansatz, der hier gewählt wurde, erstellt nur ein Anwendungsfallmodell, das die Unterscheidung zwischen funktionalen und hypertext-spezifischen Anwendungsfällen durch das UML-Stereotyp «navigation» visualisiert.

Web-Anwendungen haben immer mindestens einen menschlichen Benutzer (meist anonym). In dem Beispiel des Konferenzbegutachtungssystems, kurz Begutachtungssystem genannt, können vier Akteure unterschieden werden: Benutzer des Begutachtungssystems, Autor eines Konferenzbeitrags, Gutachter eines Konferenzbeitrags und Mitglied des Programmkomitees. Abbildung 3.3 zeigt als Ausgangspunkt für die weitere Modellierung ein Anwendungsfalldiagramm, das einen Teil des Anwendungsfallmodells des Begutachtungssystems darstellt.

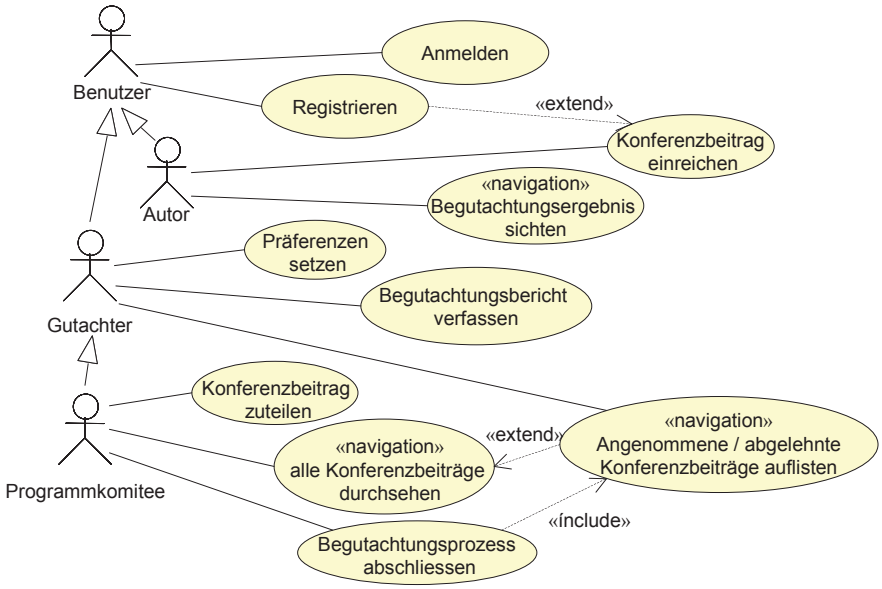

Abb. 3.3 *Anwendungsfalldiagramm für das Begutachtungssystem*

Anwendungsfälle sollten detailliert beschrieben werden. Diese Beschreibung kann pro Anwendungsfall in textueller Form oder mithilfe eines Aktivitätsdiagramms erfolgen. Letzteres wird vor allem dann eingesetzt, wenn eine komplexere Anwendungslogik dem entsprechenden Anwendungsfall zugrunde liegt. Immer öfter entspricht so ein Anwendungsfall einem Web Service, der bei der Implementierung eingebunden wird. In Abb. 3.4 ist exemplarisch der Ablauf für den Einreichprozess von Konferenzbeiträgen dargestellt.

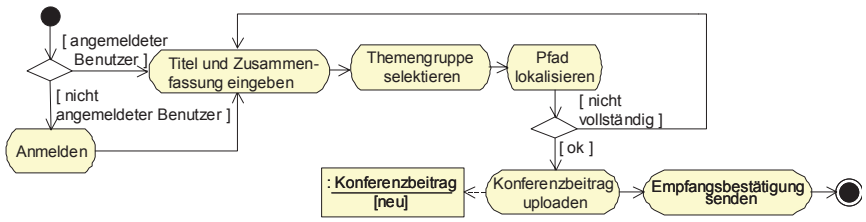

Abb. 3.4 *Aktivitätendiagramm für den Einreichprozess*

3.5. Content-Modellierung

Die durch eine Web-Anwendung zur Verfügung gestellte Information ist - nicht zuletzt wegen des Ursprungs des Web als Informationsmedium - mitentscheidend für den Erfolg einer Web-Anwendung (siehe auch Kapitel 11).

Bei statischen Web-Anwendungen ist die Strukturierung des Content im Sinne einer reinen Datenmodellierung meist ausreichend. Für komplexere Web-Anwendungen ist (im Sinne der Kategorisierung in Kapitel 1) auch die Modellierung von Anwendungslogik notwendig. Content-Modellierung umfasst somit die Erstellung des Problembereichsmodels bestehend aus statischen und dynamischen Aspekten, wie es aus dem traditionellen Software Engineering bekannt ist. Dabei müssen allerdings zusätzlich folgende Charakteristika von Web-Anwendungen berücksichtigt werden:

- *Dokumentzentrierter Charakter und Multimedialität.* Es ist erforderlich, die verschiedenen Medienformate bereits bei der Modellierung des Content mit zu berücksichtigen. Dabei sind auch die der Information zu Grunde liegenden Strukturen zu beachten. Neben softwaretechnischen Aufgaben kommen bei der Aufbereitung des Content zusätzlich auch redaktionelle und gestalterische Aufgaben hinzu. Diese als Web-Design bezeichneten Aufgaben stellen nach [Pres00c] einen eigenständigen Bereich dar und werden hier nicht genauer beschrieben (siehe hierzu auch Kapitel 11).
- *Integration existierender Daten und Software.* Web-Anwendungen bauen oftmals auf existierenden Datenbeständen und Software-Komponenten auf, die nicht in Hinblick auf Web-Anwendungen erstellt wurden. Content-Modellierung muss dabei die beiden potentiell widersprüchlichen Zielsetzungen befriedigen, einerseits bestmöglich den Content-Bedarf der zu erstellenden Web-Anwendung abdecken, andererseits die bestehenden Datenstrukturen und Softwarekomponenten mit einbeziehen.

3.5.1. Ziele

Ziel der Content-Modellierung ist es, die durch das Requirements Engineering ermittelten Informations- und Funktionsanforderungen in ein Modell zu überführen. Der hypertextuelle Charakter der Web-Anwendung und die Anforderungen an die Präsentation finden hierbei jedoch noch keine Berücksichtigung.

Ergebnis der Content-Modellierung ist ein Modell, das sowohl die strukturellen Aspekte des Content zum Beispiel in Form eines Klassendiagramms als auch in Abhängigkeit von der Art der Web-Anwendung die Verhaltensaspekte zum Beispiel in Form von Zustands- und Integrationsdiagrammen umfasst.

3.5.2. Konzepte

Für die Modellierung des Content wird wie bereits ausgeführt auf Konzepten und Methoden der Datenmodellierung bzw. der objektorientierten

Modellierung aufgebaut. Dabei wird ebenfalls nach Redundanzfreiheit, Wiederverwendbarkeit und Minimalität der vorhandenen Information gestrebt.

Abbildung 3.5 zeigt ein stark vereinfachtes UML-Klassendiagramm des Konferenzbegutachtungssystems. Es modelliert, dass eine Konferenz zu einer Reihe von Themen abgehalten wird, zu der sich Benutzer anmelden können, und zu der Konferenzbeiträge eingereicht werden können. Ein Konferenzbeitrag wird einer Begutachtung von bis zu drei Gutachtern unterzogen. Dass Autoren nicht den eigenen Konferenzbeitrag beurteilen können, ist hier durch eine Invariante in der Annotation zum Konferenzbeitrag modelliert.

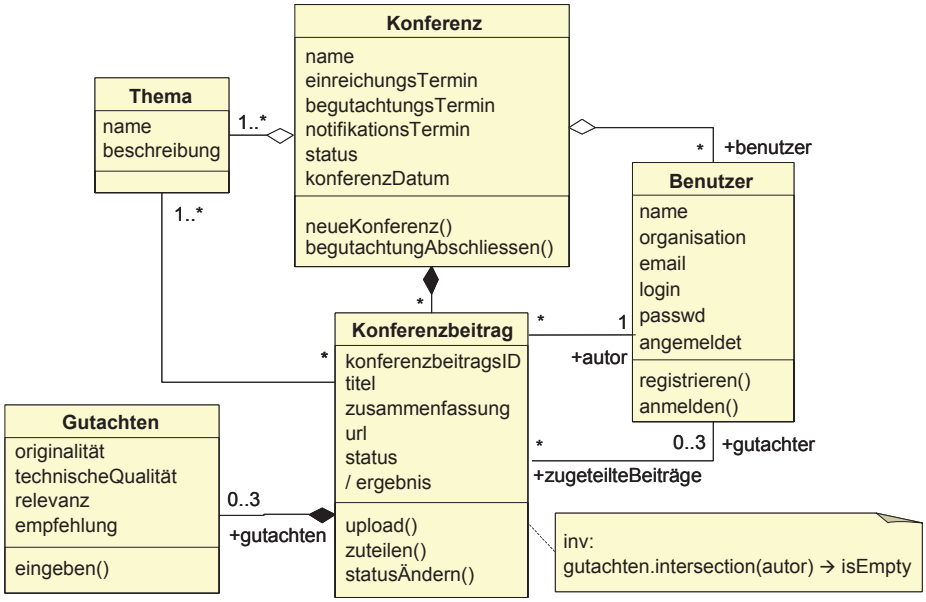

Abb. 3.5 *Klassendiagramm des Begutachtungssystems*

Dieses Klassenmodell dient in Folge als Basis für die Modellierung des Hypertexts und der Präsentation der Beispielanwendung.

Zusätzlich zum statischen Klassendiagramm wird für dieses Beispiel in Abb. 3.6 ein Zustandsdiagramm für einen Konferenzbeitrag dargestellt. Es zeigt, dass ein eingereichter Konferenzbeitrag nach Ende der Einreichfrist bis zu 3 Gutachtern zur Begutachtung zugeteilt wird. Ist der Grenzwert bei der Beurteilung nicht erreicht, wird der Konferenzbeitrag abgelehnt, ansonsten wird der Konferenzbeitrag angenommen, die Autoren per E-Mail benachrichtigt und der Konferenzbeitrag nach Einreichen der Endversion gedruckt.

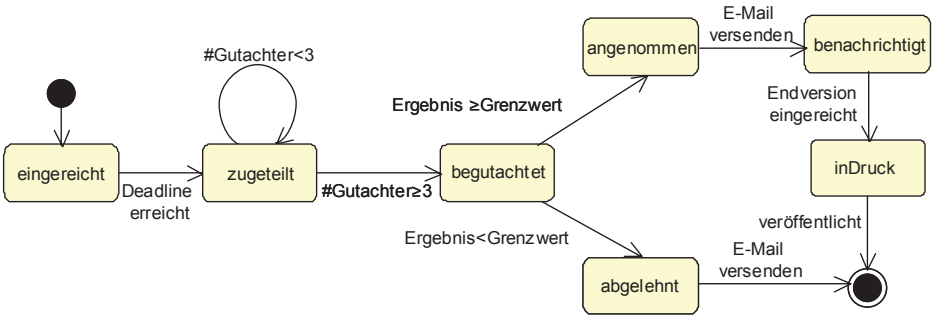

***Abb. 3.6** Zustandsdiagramm für einen Konferenzbeitrag*

3.6. Hypertextmodellierung

Die Nicht-Linearität von Hypertext ist eine der wichtigsten Eigenschaften, die bei der Modellierung von Web-Anwendungen zu berücksichtigen ist. Diese muss sorgfältig konzipiert werden, um Desorientierung und kognitive Überlastung des Benutzers durch geeignete Zugriffsstrukturen, d.h. Navigationsmöglichkeiten zu vermeiden (siehe auch Kapitel 11).

3.6.1. Ziele

Ziel der Hypertextmodellierung – auch als Navigationsmodellierung bekannt – ist es, die Navigierbarkeit, d.h. die dem Benutzer zur Verfügung stehenden Navigationspfade durch den Inhalt der Web-Anwendung zu spezifizieren. Das *Ergebnis* der Hypertextmodellierung ist einerseits das *Hypertextstrukturmodell*, auch *Navigationsstrukturmodell* genannt, das die Struktur des Hypertexts definiert, d.h. welche Klassen des Content-Modells mittels Navigation besucht werden können. Andererseits wird die Verfeinerung des Hypertextstrukturmodells um Zugriffselemente in Form eines *Zugriffsmodells* spezifiziert.

Die Hypertextmodellierung fokussiert auf die strukturellen Aspekte des Hypertexts und der Zugriffselemente. Das Verhalten der Web-Anwendung durch Navigation wird sehr selten explizit repräsentiert, weil es dem Entwickler sehr geringe Zusatzinformation zum schon erstellten Hypertextstrukturmodell bietet. Eine Ausnahme ist die Methode OOHDM [ScGR02], die einen Ansatz zur Modellierung von Szenarien anbietet, um aus einem Anforderungsmodell das Hypertextmodell zu konstruieren.

3.6.2. Konzepte der Hypertextstrukturmodellierung

Im Gegensatz zur Content-Ebene, bei der fast ausschließlich auf ER-Diagramme oder Klassendiagramme zur Repräsentation zurückgegriffen wird, werden für die Hypertextstrukturmodellierung häufig spezialisierte Notationen verwendet. Die Modellierung der Hypertextstruktur basiert dabei auf den Konzepten des Hypertexts, d.h. Knoten (auch Seiten oder Dokumente genannt) und Verweise zwischen den Knoten (auch Links genannt).

Ausgangspunkt bei der Erstellung des Hypertextstrukturmodells ist zumeist das Content-Modell, in dem entschieden wird, welche Klassen und Objekte des Content-Modells im Hypertext als Knotentypen und Knoten verfügbar sein sollen. Einige Methoden definieren Transformationsvorschriften für die Ableitung von Links auf Basis von Beziehungen auf der Content-Ebene, wobei zusätzliche Links als explizite Entwurfsentscheidungen hinzugefügt werden können. Andere Methoden modellieren die Hypertextstruktur unabhängig vom Content-Modell. Auf Basis der Anforderungen, die in den Anwendungsfällen beschrieben sind, können mehrere Hypertextstrukturmodelle modelliert werden, die somit unterschiedliche Hypertextsichten, auch navigationale Sichten genannt, auf den Content definieren. Auf diese Art und Weise können im Sinne einer Personalisierung zum Beispiel für die Berücksichtigung von Zugriffsrechten unterschiedlicher Benutzerrollen verschiedene Hypertextstrukturmodelle erstellt werden (siehe dazu auch Unterkapitel 3.8).

Für das Beispiel des Konferenzbegutachtungssystems sind Hypertextsichten für folgende Benutzerrollen nötig: Autor, Gutachter und Mitglieder des Programmkomitees. Abbildung 3.7 stellt das Hypertextstrukturmodell für die Sicht der Mitglieder des Programmkomitees dar. Diese Mitglieder können alle eingereichten Konferenzbeiträge sehen. Weiters haben sie Zugriff auf die Liste der angenommenen und abgelehnten Konferenzbeiträge sowie auf das Profil der Gutachter. In Abb. 3.7 werden in Anlehnung an die Modellierungsmethode UWE Klassen, die Knoten im Hypertextstrukturmodell repräsentieren, durch das UML-Stereotyp «navigation class» gekennzeichnet, um sie von Content-Klassen zu unterscheiden. Links wiederum werden mit einer als «navigation link» stereotypisierten gerichteten Assoziation modelliert.

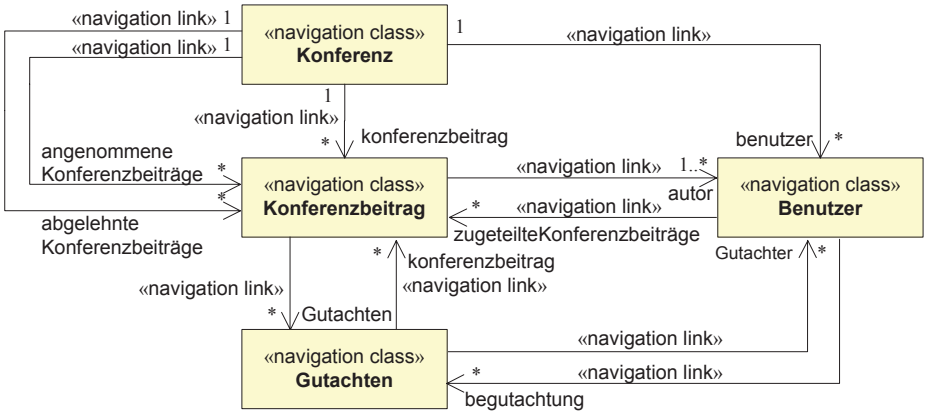

Abb. 3.7 *Hypertextstrukturmodell des Begutachtungssystems aus Sicht eines Mitglieds des Programmkomitees*

In der Literatur werden verschiedene Arten von Verweise genauer unterschieden, um die Semantik des Hypertextstrukturmodells weiter zu verfeinern. So unterscheidet beispielsweise die Methode HDM [GaMP95]:

- *Strukturelle Verweise*, die Elemente des selben Knotens verbinden, z.B. von Gutachten zu den Detailbewertungen.
- *Perspektivische Verweise*, die verschiedene Sichten eines Knotens in Beziehung setzen, z.B. PostScript-Version und PDF-Version eines Konferenzbeitrags.
- *Anwendungsverweise,* die unterschiedliche Knoten abhängig von der Anwendung in Beziehung setzen, z.B. Link zum »Best Paper«.

Andere Klassifikationen basieren auf dem möglichen Transport von Information beim Navigieren. So unterscheidet die Methode WebML [CFB+03]:

- *Kontextuelle Verweise*, die Information transportieren, z.B. die Nummer eines Gutachters, um von einem Gutachter zu den von ihm erstellten Gutachten zu navigieren.
- *Nicht-kontextuelle Verweise*, die keine Information bei der Navigation transportieren, z.B. von einem Gutachten zu dem zugrunde liegenden Bewertungsschema.

In Hinblick auf die Verteilung von Knoten der Hypertextebene auf Seiten der Präsentationsebene (siehe Unterkapitel 3.7) unterscheidet die Methode WebML zusätzlich:

- *Intra-Seiten Verweise*, wenn Quelle und Ziel eines Verweises zur selben Seite gehören, z.B. wenn ein Link direkte Navigation zur Zusammenfassung

des Konferenzbeitrags ermöglicht, die weiter unten auf der Seite angezeigt wird.

- *Inter-Seiten Verweise*, wenn Quelle und Ziel sich auf verschiedenen Seiten befinden, z.B. wenn Detailangaben zu den Autoren auf einer anderen Seite als der Konferenzbeitrag zu finden sind.

Basierend auf den funktionalen Anforderungen von Web-Anwendungen, definiert die Modellierungsmethode UWE [KoKr02] folgende unterschiedliche Verweise:

- *Navigationsverweise*, die zur Navigation zwischen Knoten dienen, z.B. Verweise zwischen Konferenzbeiträgen und deren Autoren.
- *Prozessverweise*, die auf den Startknoten eines Ablaufs verweisen, z.B. zum Beginn der Gutachtenerstellung.
- *Externe Verweise*, die auf einen nicht direkt zur Anwendung gehörenden Knoten verweisen, z.B. auf die durch eine Dachorganisation aufgestellten Formatierungsrichtlinien, die nicht direkt im Konferenzbegutachtungssystem gespeichert sind.

Die Modellierungsmethode OO-H [GoCP01] definiert die vier folgenden Verweise:

- *R-Links* verweisen auf einen Ausgangspunkt der Navigation, um einer funktionalen Anforderung des Systems gerecht zu werden, z.B. Gutachten eingeben.
- *T-Links* verweisen zwischen internen Knoten, z.B. von einem Autor zu seinen Konferenzbeiträgen.
- *E-Links* verweisen auf einen externen Knoten, z.B. auf externe Formatierungsrichtlinien.
- *S-Links* verweisen auf Dienste, z.B. auf eine externe Suchmaschine.

3.6.3. Konzepte der Zugriffsmodellierung

Das Hypertextstrukturmodell alleine ist aber nicht ausreichend für die Beschreibung, wie Knoten durch Navigation erreicht werden können. Es bedarf einer detaillierten Modellierung von *Zugriffsstrukturen*, die eine Reihe von Navigations- und Orientierungshilfen für den Zugriff auf einen Knoten einführen. Dabei werden wiederkehrende Zugriffsstrukturen identifiziert, die in Entwurfsmuster, sogenannte »Hypermedia-Patterns« oder »Navigations-Patterns« zusammengefasst sind (siehe u.a. [GeCo00], [LyRS99], [RoSL98]).

Gerade durch die Verwendung dieser erprobten Navigationsmuster kann die Qualität der Hypertextmodellierung entscheidend verbessert werden.

Wenn man im Beispiel des Konferenzbegutachtungssystems von einem Gutachter zu einem ihm zugeteilten Konferenzbeitrag gelangen will, ist es notwendig, diesen spezifischen Konferenzbeitrag bei der Navigation zu identifizieren. Das kann zum Beispiel über eine Liste der Konferenzbeiträge realisiert werden, in der man den entsprechenden Konferenzbeitrag selektieren kann. Eine solche Auswahlliste zur Unterstützung der Navigation ist auch als »Index« bekannt. Der Index ist eine Zugriffsstruktur, die die Auswahl einer Instanz einer Klasse erlaubt. Ein »Menü« hingegen ermöglicht den Zugriff auf Knoten unterschiedlicher Klassen oder auf Untermenüs. Weitere Zugriffsstrukturen sind »Guided-Tour« und »Query«. Eine Guided-Tour erlaubt ein sequentielles Durchlaufen einer Reihe von Knoten. Eine Query dient der Suche nach Knoten und erlaubt somit den direkten Zugriff auf diese. In den meisten Modellierungsmethoden sind die hauptsächlich angewendeten Hypermedia-Patterns als eigene Modellelemente verfügbar. Spezielle Entwurfsmuster sind auch der Verweis »Home«, der auf eine Startseite einer Anwendung hinweist, und »Landmark«, der auf eine von allen Seiten erreichbare Seite verweist.

Diese Zugriffsstrukturen werden dem Hypertextstrukturmodell teilweise in automatisierter Form hinzugefügt. Zum Beispiel können Indexe überall automatisch hinzugefügt werden, wo auf eine Menge (> 1) von Objekten einer Klasse zugegriffen werden soll.

Abbildung 3.8 stellt ein vereinfachtes Zugriffsmodell der Beispielanwendung für die in der Hypertextstrukturmodellierung gezeigten Sicht der Mitglieder des Programmkomitees dar. Diese Mitglieder haben Zugriff auf die eingereichten Konferenzbeiträge, Gutachten und Profile der Benutzer des Systems. Zugriff auf die Konferenzbeiträge ist über die eindeutige Nummer der Konferenzbeiträge möglich oder es kann nach den Titeln der Konferenzbeiträge gesucht werden. In UWE werde die Zugriffsstrukturen Menü, Index, Query und Guided-Tour durch die entsprechenden UML-Stereotypen «menu» (z.B. Menü Konferenz), «index» (z.B. BegutachtungsStatus), «query» (z.B. SucheKonferenzbeitragNachTitel) und «guided-tour» genauer spezifiziert.

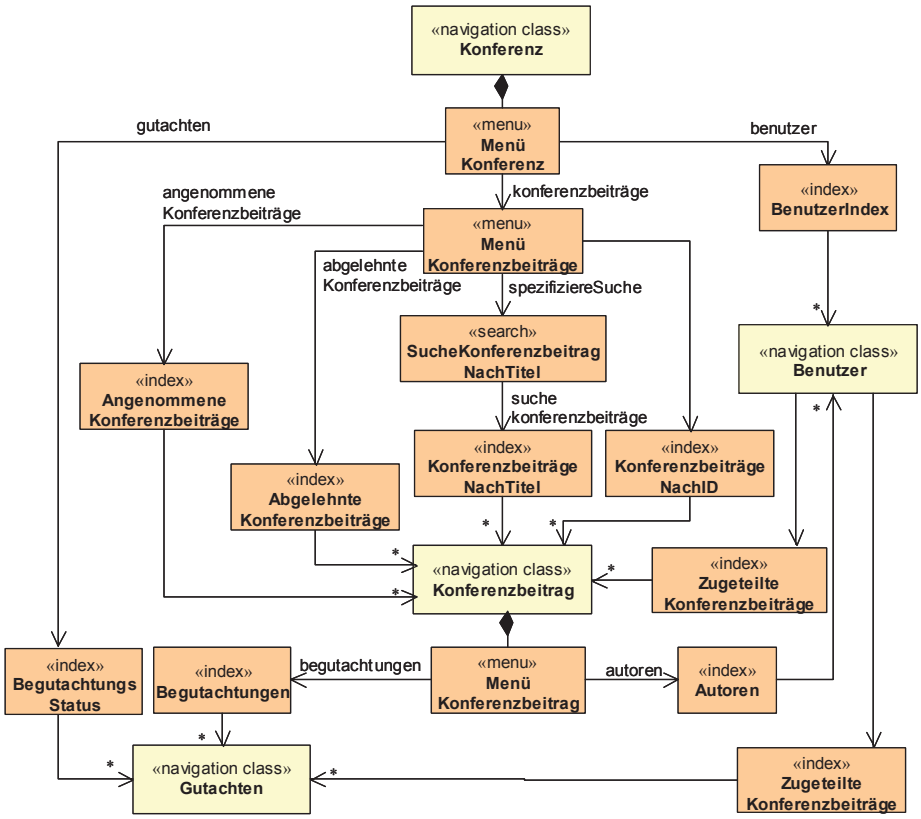

Abb. 3.8 *Vereinfachtes Zugriffsmodell des Hypertextstrukturmodells aus Abb. 3.7*

3.6.4. Schnittstellen zur Content-Modellierung

Die Hypertextmodellierung ist somit zwischen der Content-Modellierung und der Präsentationsmodellierung eingebettet. Je nach der zu Grunde liegenden Modellierungsmethode weist das Hypertextmodell eine starke Abhängigkeit vom Content-Modell auf. Es existiert dabei sowohl eine Abhängigkeit auf Typebene, zum Beispiel welche Klassen des Content-Modells welche Knotentypen des Hypertextmodells bilden, als auch eine Instanzabhängigkeit, d.h. welche Mengen von Objekten des Content-Modells welche Mengen von Knoten des Hypertextmodells bilden. Nicht alle Methoden beschreiben genau, welche Abhängigkeiten zwischen dem Content-Modell und dem Hypertextmodell bestehen. Manche Methoden wiederum spezifizieren eine direkte Abhängigkeit des Hypertexts vom Content, in dem Knoten als Sichten im Sinne von »Datenbank-Views« auf das Content-Modell gebildet werden.

3.7. Präsentationsmodellierung

Analog zum traditionellen Software Engineering beschäftigt sich die Präsentationsmodellierung mit der Benutzerschnittstelle und somit mit dem »*look and feel*« der Web-Anwendung. Im Gegensatz zu traditionellen Anwendungen ist das zentrale Element der Präsentation bei Web-Anwendungen die Seite als Einheit der Visualisierung.

3.7.1. Ziele

Ziel der Präsentationsmodellierung ist es, die Struktur und das Verhalten der Benutzerschnittstelle so zu gestalten, dass die Einfachheit der Interaktion mit der Web-Anwendung im Sinne einer Selbsterklärbarkeit gewährleistet ist und der Kommunikations- und Repräsentationsaufgabe der Web-Anwendung Rechnung getragen wird. *Ergebnis* der Präsentationsmodellierung ist einerseits ein durchgängiges Präsentationskonzept durch die Modellierung wiederkehrender Elemente der Seiten, wie zum Beispiel Kopf- und Fußzeilen. Andererseits ist individuell für jede Seite, die dem Benutzer angeboten werden soll, deren Zusammensetzung sowie die Gestaltung der darin enthaltenen Felder, Texte, Bilder, Formulare usw. erforderlich. Neben dem strukturellen Aufbau der Seiten beschreibt das Präsentationsmodell auch die verhaltensorientierten Aspekte der Benutzerschnittstelle, wie zum Beispiel die Aktivierung von Funktionen der Anwendungslogik durch das Anklicken einer Schaltfläche. Auf Grund der vielfältigen Navigationsmöglichkeiten und des damit verbundenen Risikos des Orientierungsverlusts ist darauf zu achten, dem Benutzer auf der Präsentationsebene entsprechende Orientierungshilfen zu geben, z.B. eine Anzeige des bisherigen Navigationspfads.

Nicht alle Methoden zur Modellierung von Web-Anwendungen stellen technologieunabhängige Konzepte für die Präsentationsmodellierung zur Verfügung, sondern greifen auf technologiebasierte Konzepte zurück, z.B. beschrieben durch Stylesheet-Sprachen wie XSL [PiKr03].

Die bei Web-Anwendungen ebenfalls wichtige dekorative Ausgestaltung der Benutzerschnittstelle in Form des graphischen Layout wird oftmals von Graphik-Designern anhand von Skizzen oder durch die werkzeuggestützte Realisierung prototypischer Seiten konzipiert [This03]. Diese Aufgabe des Web-Design wird nicht als Bestandteil der Präsentationsmodellierung betrachtet.

3.7.2. Konzepte

Zur Modellierung der Präsentation werden Modelle erstellt, die die Verteilung der Knoten der Hypertext-Ebene auf Seiten der Präsentation beschreiben. Dabei werden Modellelemente auf drei hierarchischen Ebenen beschrieben:

- *Präsentationsseite:* beschreibt eine dem Benutzer präsentierte Seite, als Einheit der Visualisierung. Diese kann aus verschiedenen Präsentationseinheiten zusammengesetzt sein.
- *Präsentationseinheit:* dient der Gruppierung zusammengehöriger Benutzerschnittstellenelemente, die ein logisches Fragment der Seite darstellen, und präsentiert einen Knoten des Hypertextmodells.
- *Präsentationselement:* ist der elementare Baustein des Präsentationsmodells. Präsentationselemente repräsentieren die einzelnen Informationen eines Knoten und können Text, Bilder, Audio, usw. umfassen.

Die Komposition der Präsentationsseiten kann wie in Abb. 3.9 gezeigt auf Basis der geschachtelten UML-Darstellung der »Komposition« visualisiert werden. Dabei werden Präsentationsseiten und Präsentationseinheiten durch mit «page»-und «presentation unit» stereotypisierte Klassen dargestellt. Die unterschiedlichen Arten von Präsentationselementen werden ebenfalls durch entsprechende UML-Stereotype gekennzeichnet. Abbildung 3.9 zeigt zwei Präsentationsseiten des Konferenzbegutachtungssystems. Ein Konferenzbeitrag wird auf der Seite »KonferenzbeitragSeite« mit den entsprechenden Feldern sowie einem Link auf die Vollversion und zur Anzeige der Autoren positioniert. Weiters kann der Prozess zur Eingabe eines Gutachtens mithilfe der mit «button» gekennzeichneten Schaltfläche gestartet werden. Die Seite »AutorenSeite« beinhaltet zwei Präsentationseinheiten, die Liste der Autoren und die Detailinformationen eines Autors.

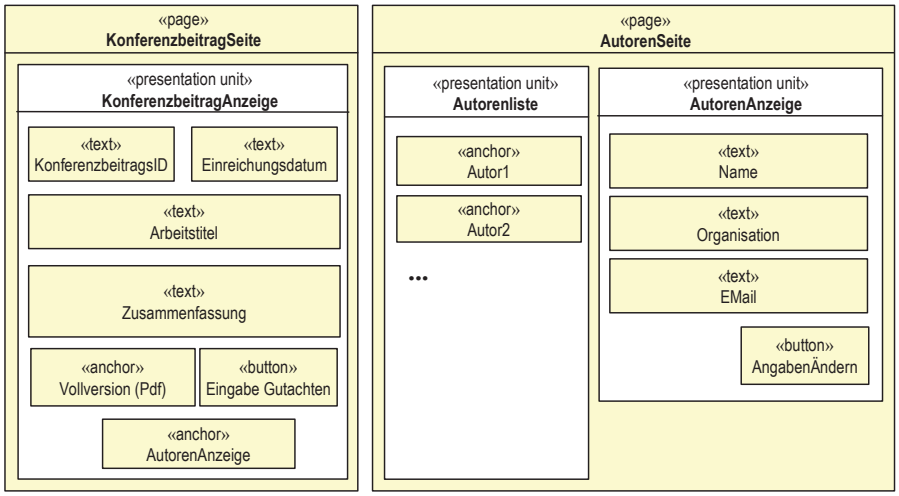

Abb. 3.9 *Präsentationsseiten im Begutachtungssystem*

Verhaltensaspekte der Benutzerschnittstelle, wie zum Beispiel die Benutzerinteraktion eines Gutachters, um zu seinen zugeteilten

Konferenzbeiträgen zu gelangen, lassen sich wie in Abb. 3.10 gezeigt durch entsprechende Verhaltensdiagramme modellieren. In der Regel findet die Interaktion des Benutzers mit der Web-Anwendung nicht nur in der Präsentationsebene ihren Niederschlag, sondern wird je nach Art der Interaktion entsprechend an die Hypertextebene und Content-Ebene weitergereicht. Wie aus dem Sequenzdiagramm in Abb. 3.10 ersichtlich, aktiviert der Gutachter über die Konferenz-Homepage und die Navigationsleiste die Navigation zum Index der zugeteilten Konferenzbeiträge. Diese Information wird wiederum aus den Konferenzbeiträgen der Content-Ebene, dargestellt durch ein Multiobjekt, zusammengestellt. Über die daraufhin angezeigte Liste der zugeteilten Konferenzbeiträge kann der Benutzer wiederum einen Konferenzbeitrag durch Navigation selektieren, der im Detail angezeigt wird.

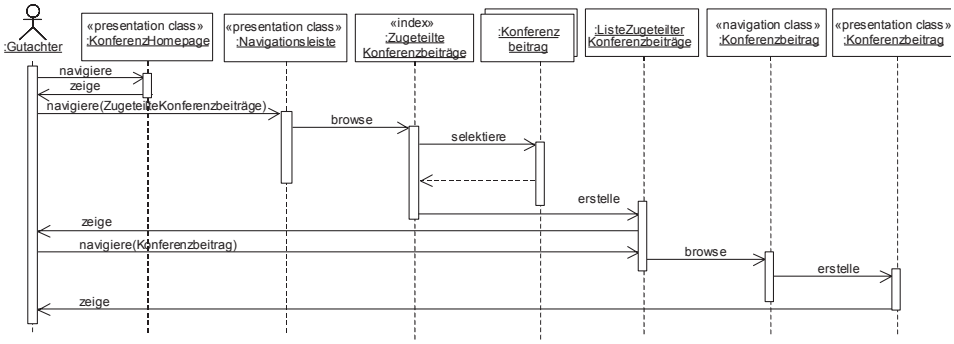

Abb. 3.10 *Szenario aus dem Präsentationsmodell des Begutachtungssystems (vereinfacht)*

3.7.3. Schnittstellen zur Hypertextmodellierung

Analog zur Abbildung des Content-Modells auf das Hypertextmodell ist auch die Abbildung der Hypertextelemente auf die Elemente der Präsentation zu spezifizieren. Dabei wird zu meist davon ausgegangen, dass alle Instanzen eines Knotens auch auf der Präsentationsebene angezeigt werden.

Wie bereits erwähnt, sind die durch einen Benutzer ausgelösten Aktionen nicht notwendiger Weise auf die Präsentationsebene beschränkt. Daher sind zusätzlich die Entsprechungen sowohl auf der Content-Ebene in Form von Methoden auf Objekten der Anwendungslogik und auf der Hypertext-Ebene in Form von Navigation zu berücksichtigen.

3.8. Kontextualitätsmodellierung

Die zunehmend an Bedeutung gewinnende Kategorie von ubiquitären Web-Anwendungen erfordert die Berücksichtigung von Kontextinformation und

einer entsprechenden Anpassung der Anwendung bereits bei der Modellierung (s.a. Kapitel 1). Vorarbeiten dazu stammen aus dem Bereich der Personalisierung ([Kobs01], [Brus01]) und dem Mobile Computing ([EiVP01], [WaSc01]).

Modellierung von Kontextualität ist allerdings noch ein sehr junges Gebiet, sodass existierende Methoden zur Modellierung von Web-Anwendungen zumeist keine umfassende Kontextualitätsmodellierung zur Verfügung stellen ([ChSP98], [Frat99], [KaRS00]).

3.8.1. Ziele

Ziel der Kontextualitätsmodellierung ist die explizite Darstellung der Kontextinformation sowie der daraus abzuleitenden Anpassungen. Dies erfolgt je nach Art der Modellierungsmethode unterschiedlich - nicht immer ist das *Ergebnis* dabei ein explizites Kontextualitätsmodell. Zumeist ist die Modellierung der Kontextualität mit den Content-, Hypertext- und Präsentationsmodellen eng verbunden.

Die Berücksichtigung von Kontextualität ist von Wartung und Re-Engineering explizit zu unterscheiden. Kontextualitätsmodellierung berücksichtigt Kontextinformation, die zum Modellierungszeitpunkt vorhergesagt werden kann und beim Ausführen einer Web-Anwendung in verschiedenen Ausprägungen auftreten kann. Im Gegensatz dazu ist die Anpassung auf Grund von Veränderungen in den organisatorischen oder technologischen Rahmenbedingungen Aufgabe der Wartung und des Re-Engineering.

3.8.2. Konzepte

Kontextualität verlangt die Auseinandersetzung mit der zu berücksichtigenden Nutzungssituation der Web-Anwendung, also vor allem mit der Frage »was« angepasst werden soll und zu »welchem Zeitpunkt«. Um die Web-Anwendung personalisieren zu können, müssen die Benutzer sowie deren Präferenzen in einem sogenannten Benutzermodell oder Benutzerprofil modelliert und verwaltet werden. Für die Anpassungen aus dem Bereich des Mobile Computing müssen zum Beispiel Endgeräteprofile, Lokationsinformation, Bandbreite der Übertragung, usw. berücksichtigt werden. Diese Informationen werden in einem Kontextmodell dargestellt in Form eines Klassendiagramms zusammengefasst. Zur Laufzeit können sie unterschiedliche Ausprägungen haben, die sich über die Zeit ändern (z.B. Benutzer ändern ihre Präferenzen, die Anwendung wird an verschiedenen Orten »konsumiert«). Dies wiederum ist Anlass zur Anpassung der Web-Anwendung. In Hinblick auf das Abstraktionsniveau der Kontextinformation kann zwischen *physischem* und *logischem Kontext*

unterschieden werden. Der physische Kontext ergibt sich aus der jeweiligen Nutzungssituation (zum Beispiel der Login-Name des Benutzers oder die GSM-Zelle, in der sich ein mobiler Benutzer gerade aufhält). Der logische Kontext stellt zusätzliches Kontextwissen zur Verfügung (zum Beispiel Arbeitsadresse versus Heimadresse, Arbeitszeit versus Freizeit). Diese Kontextinformation kann auch über externe Quellen in die Web-Anwendung eingebracht werden. Ein Beispiel einer solchen externen Quelle, die Kontextinformation für die genauere Bestimmung des Lokationskontexts zur Verfügung stellt, sind Geographische Informationssysteme (GIS). Es existieren erste Ansätze zur Entwicklung von universellen Komponenten, die unterschiedliche Arten physischer und logischer Kontextinformation zur Verfügung stellen kann, wie das *ContextToolkit* [Abow99] oder das *NEXUS Projekt* [GLMR01].

Die Modellierung der Anpassung an den Kontext kann auf zwei grundsätzliche Arten erfolgen. Es kann *»ergebnisorientiert«* vorgegangen werden, indem verschiedene Modelle bzw. Modellvarianten in Hinblick auf die unterschiedlichen Ausprägungen der zu berücksichtigenden Kontextinformation erstellt werden. Man spricht in diesem Zusammenhang von *statischer Adaptierung*. Die in Abb. 3.7 beispielhaft gezeigte Hypertextmodellierung beschreibt bereits eine, auf den Kontext der Benutzerrolle »Programmkomiteemitglied« statisch adaptierte Hypertextstruktur. Der Nachteil der statischen Adaptierung ist das exponentielle Wachstum der zu berücksichtigenden Modellvarianten. Im Gegensatz dazu wird bei der *dynamischen Adaptierung* das Content-, Hypertext- und Präsentationsmodell um kontextabhängige Transformationsregeln ergänzt, die die zur Laufzeit zu erstellenden Varianten beschreiben. Beispielsweise können solche dynamischen Transformationsregeln das Hinzufügen oder Weglassen von Modellelementen oder Filtern von Instanzen spezifizieren, um eine personalisierte Liste der für den Benutzer interessanten Konferenzbeiträge zu erstellen. Inwieweit dynamische Adaptierung oder statische Adaptierung zu bevorzugen sind, hängt vom Anwendungsfall ab. Vorteil der dynamischen Adaptierung ist, dass die kombinatorische Explosion von Modellvarianten vermieden wird. Nachteilig ist hingegen, dass das Ergebnis, also die dem Kontext angepasste Variation des Modells, nicht direkt vorliegt, sondern sozusagen »zur Laufzeit« erstellt wird, und somit das Modellverständnis darunter leidet. Eine detaillierte Behandlung des State-of-art der Kontextualitätsmodellierung kann in [KPRS03] nachgelesen werden.

Beispiele dynamischer Adaptierung der Hypertext- und Präsentationsebene für das Konferenzbegutachtungssystem werden in Abb. 3.11 und Abb. 3.12 gezeigt. Die Adaptierungsinformation wird mithilfe von mit »contextuality« stereotypisierten Annotationen an die anzupassende Klasse angefügt. Die hier informell beschriebenen Regeln können, mittels einer formalen Sprache, zum Beispiel der Object Constraint Language (OCL) [OMG01], in weiteren

Verfeinerungsschritten genauer spezifiziert werden. Abbildung 3.11 zeigt eine Personalisierung der Hypertextstruktur, sodass nur die für den Benutzer interessanten Konferenzbeiträge navigierbar sind. Die Elemente der Zugriffsstruktur »InteressanteKonferenzbeiträge« werden dabei aufgrund der persönlichen Interessensgebiete durch Transformationsregeln dynamisch angepasst.

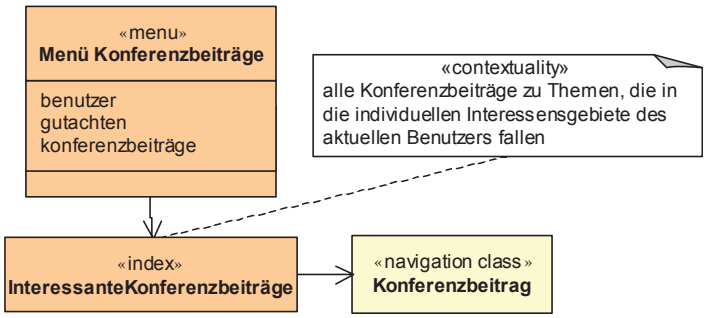

Abb. 3.11 *Dynamische Adaptierung eines Index im Hypertextmodell*

Das Beispiel in Abb. 3.12 zeigt, wie Elemente des Präsentationsmodells mit Transformationsregeln angepasst werden können. Konkret soll die Schaltfläche »Eingabe Gutachten« nur für Benutzer in der Rolle »Gutachter« sichtbar sein.

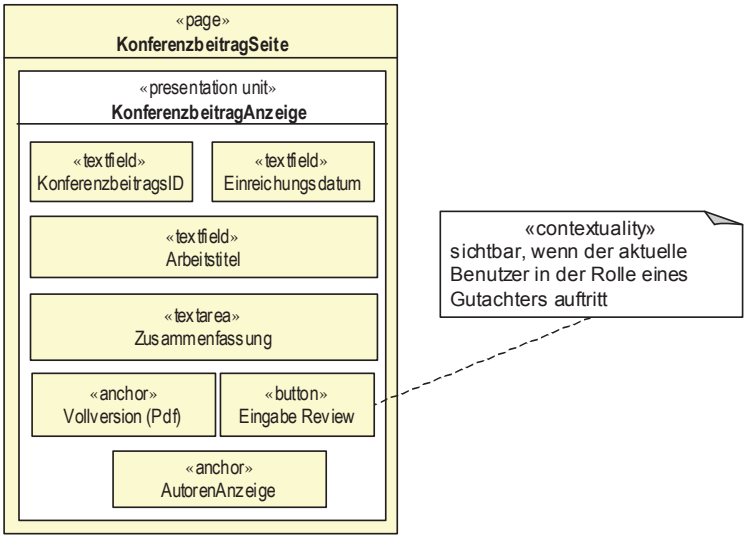

Abb. 3.12 *Dynamische Adaptierung einer Seite im Präsentationsmodell*

3.8.3. Schnittstellen zur Content-, Hypertext- und Präsentationsmodellierung

Wie in Abb. 3.2 bereits gezeigt, kann Kontextualität Einfluss auf alle Ebenen der Modellierung von Web-Anwendungen haben. Änderungen können dabei entweder lokal auf eine Ebene beschränkt sein oder müssen auf mehreren Ebenen durchgeführt werden. Eine Trennung des Kontextualitätsmodells von den Modellen der Content-, Hypertext- und Präsentationsebene wäre in Hinblick auf Änderbarkeit, Flexibilität und Kapselung empfehlenswert, ist allerdings in den existierenden Methoden oftmals nicht vorgesehen. Manchmal ist diese Trennung auch nur schwer möglich. So sind der Benutzer einer Web-Anwendung und seine Präferenzen Gegenstand der Kontextmodellierung, in der Regel ist der Benutzer aber auch Gegenstand der Content-Modellierung.

3.9. Methoden und Werkzeuge zur Modellierung von Web-Anwendungen

Grundlegendes Ziel aller Modellierungsmethoden ist es, auf die Anforderungen der Entwicklung von Web-Anwendungen zugeschnittene Konzepte in Form einer Modellierungssprache zur Verfügung zu stellen [ReSc00]. Viele Methoden definieren zudem auch ein Vorgehensmodell und werden von einem Werkzeug unterstützt, das die erzeugten Modelle (semi-)automatisch in eine Implementierung überführt (siehe Abb. 3.14).

3.9.1. Modellierungsmethoden im Überblick

Die Methoden zur Modellierung von Web-Anwendungen basieren auf traditionellen Modellen, wie dem *Entity-Relationship Modell,* oder entwickeln eine objektorientierte Modellierungssprache wie die *Object Modelling Technique (OMT)* oder die *Unified Modelling Language (UML)* weiter. Methoden neueren Datums bauen dabei zumeist auf den Stärken zuvor entwickelter Methoden auf. Es lassen sich daher, wie in Abb. 3.13 gezeigt, vier Generationen von Methoden zur Modellierung von Web-Anwendungen identifizieren.

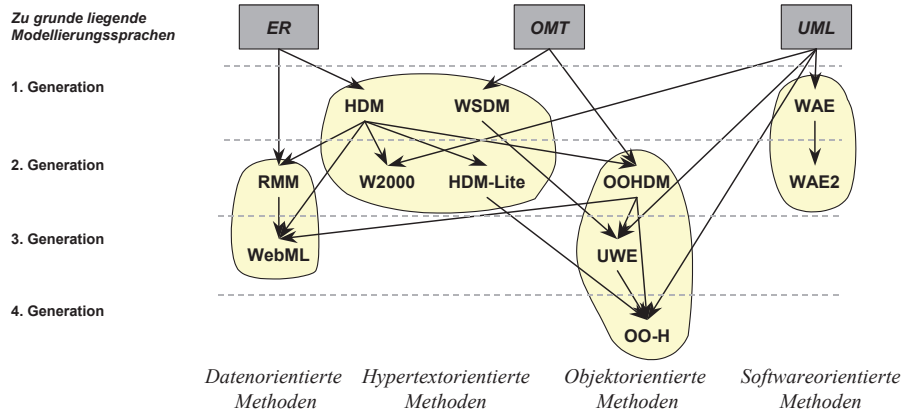

Abb. 3.13 *Historische Entwicklung der Methoden zur Modellierung von Web-Anwendungen*

Die Methoden zur Modellierung von Web-Anwendungen folgen auf Grund der Abstammung und ihres Fokus unterschiedlichen Paradigmen:

- *Datenorientierte Methoden* entspringen dem Bereich der Datenbanksysteme und basieren hauptsächlich auf dem ER-Modell. Dieses wird um spezifische Konzepte für die Modellierung der Hypertext-Ebene erweitert. Primärer Fokus dieser Methoden ist dabei die Modellierung von Datenbank-getriebenen Web-Anwendungen. Beispiele für datenorientierte Methoden sind die *Relationship Management Methodology (RMM)* [IsKK98] und die *Web Modelling Language (WebML)* [CFB+03].

- *Hypertextorientierte Methoden* stellen den Hypertext-Charakter von Web-Anwendungen in den Mittelpunkt und haben sich vor allem aus dem Bereich der Hypertextsysteme entwickelt [LoHa99]. Vertreter dieser Gruppe sind das *Hypertext Design Model (HDM)* [GaMP95], das zu *W2000* weiter entwickelt wurde [BaGP01], *HDM-lite* [FrPa98] und die *Web Site Design Method (WSDM)* [DeDe00].

- *Objektorientierte Methoden* basieren entweder auf OMT oder UML, wobei neuere Methoden fast ausschließlich den Standard UML bevorzugen. Methoden dieser Kategorie sind *Object-Oriented Hypermedia Design Method (OOHDM)* [ScGR02], *UML-based Web Engineering (UWE)* [KoKr02] und die *Object-Oriented Hypermedia Method (OO-H)* [GoCP01].

- *Softwareorientierte Methoden* betrachten Web-Anwendungen vor allem aus dem Blickwinkel traditioneller Softwareentwicklung und verwenden dazu Techniken, die stark an das klassische Software Engineering angelehnt sind. Ein Beispiel ist *Web Application Extension (WAE)* und dessen Weiterentwicklung *WAE2* [Cona03].

Abbildung 3.14 gibt einen genaueren Überblick über diese Methoden. Zusammenfassend wird dargestellt, welche Notation verwendet wird, ob die Methoden weiter entwickelt werden und welche Dimensionen der Modellierung (siehe Unterkapitel 3.3) unterstützt werden. Weiters wird aufgezeigt, ob für die Modellierung, aber auch für die Generierung Werkzeugunterstützung geboten wird. Fokus und Stärken der Methoden sind ebenfalls der Tabelle zu entnehmen.

Modellierungsmethode	Modellierungsparadigma	Notation	Weiterentwicklung	Kontextualitätsmodellierung						Werkzeugunterstützung		Stärken
				Anforderungsmodellier	Content-Mod.	Hyper-	Präse-		Struktur und Verhalten	Vorgehensweise	Generierung	
HDM-lite	HT	ER + eigene Notation	✗	✗	✗	✓	✓	✗	s	eigene	Generierungs-werkzeug	auto	Prozess zur Transformation der Modelle, automatische Generierung
OO-H	OO	UML + eigene	✓	✓	✓	✓	✗	pers	s + v	eigene	Modellierungs- & Generierungs-werkzeug	auto	fortschrittliches Werkzeug für automatische Generierung
OOHDM	OO	UML + eigene Notation	✓	✓	✓	✓	✓	pers	s + v	eigene	✗	✗	mächtige Konzepte für kontextuelle Navigation, Personalisierung
RMM	DB	ER + eigene Notation	✗	✗	✓	✓	✓	✗	s	eigene	Authoring-Werkzeug	semi	Hypertext-Modellierung basierend auf ER-Model, vordefinierter Prozess
UWE	OO	UML	✓	✓	✓	✓	✓	pers	s + v	RUP	Modellierungs- & Generierungs-werkzeug	semi	UML-basierte Methode, regelbasierte Personalisierung
W2000 (HDM)	HT	UML	✓	✓	✗	✓	✓	pers	s	✗	Erweiteres UML-Werkzeug	✗	benutzerzentrierte Hypertext-Modellierung
WAE2 (WAE)	SW	UML	✓	✓	✓	✗	✓	✗	s + v	RUP	Standard UML-Werkzeuge	✗	Implementierungsentwurf, Architekturentwurf
WebML	DB	ER,UML	✓	✓	✓	✓	✗	pers	s + v	eigene	Modellierungs- & Generierungs-werkzeug	auto	ausgereifte Notation, Datenbankintegration, Generierung
WSDM	HT	eigene Notation	✓	✗	✓	✓	✗	✗	s + v	eigene	✗	✗	benutzerzentrierte Vorgehensweise bei der Analyse

✓	wird unterstützt
✗	wird nicht unterstützt

pers	Personalisierung

s	Strukturmodellierung
v	Verhaltensmodellierung

RUP	Rational Unified Process
eigene	eigene Vorgehensweise

auto	automatische Generierung
semi	semi-automatische Generierung

DB	datenorientiert
HT	hypertextorientiert
OO	objektorientiert
SW	softwareorientiert

Abb. 3.14 *Überblick über Methoden der Modellierung von Web-Anwendungen*

HDM-lite wurde entwickelt mit dem Fokus, die Entwicklung und Generierung von Web-Anwendungen so weit als möglich zu automatisieren. *OO-H* ist eine der neuesten Methoden und vereint die Vorteile von WebML, OOHDM und UWE. Die Methode wird durch das Werkzeug VisualWADE (s.u.) zur modellgetriebenen automatischen Generierung ergänzt. Auf Grund des in *OOHDM* sehr ausgeprägten Konzepts des Navigationskontexts empfiehlt sich diese Methode für Web-Anwendungen, die sich durch verschiedenste Navigationszugriffe auszeichnen. OOHDM wurde in letzter Zeit in Richtung Personalisierung, Framework-Modellierung (unter dem Namen *OOHDM frame* [SREL01]), Web-Anwendungsarchitektur und Benutzerinteraktionsdiagramme erweitert und bietet somit viele innovative Konzepte. *RMM* ist eine Methode, die auf dem ER-Modell aufsetzt und einen

schrittweisen Prozess zur sukzessiven Verfeinerung der Modelle definiert. In Hinblick auf eine umfassende Unterstützung des gesamten Web-Anwendungsprozesses bietet *UWE* zusammen mit seinem Werkzeug *OpenUWE* (s.u.) einen interessanten Ansatz. Dies wird nicht zuletzt durch die UML-basierte Notation und auch durch die metamodellbasierte Generierungsunterstützung gewährleistet. *W2000*, die Weiterentwicklung von *HDM*, die viele Methoden beeinflusste, modelliert die Web-Anwendung aus einer hypertext- und benutzerzentrierten Sichtweise. Soll auch die technische Verteilung der Anwendungslogik spezifiziert werden, empfiehlt sich *WAE2*. *WebML* ist eine sehr verständliche, ausgereifte und auf das Wesentliche beschränkte Modellierungssprache für datengetriebene Web-Anwendungen. Die Methode wird durch das Werkzeug *WebRatio* (s.u.) unterstützt, das sowohl Modellierung als auch automatische Code- und Schnittstellengenerierung zu Datenbanken anbietet. *WSDM* stellt ein an Benutzeranforderungen orientiertes methodisches Vorgehen in den Mittelpunkt der Methode.

3.9.2. Werkzeugunterstützung

Aufgrund kurzer Entwicklungszyklen und der Komplexität von Web-Anwendungen empfiehlt sich der Einsatz von Werkzeugen, um nicht nur die Modellierung sondern insbesondere eine automatische Code-Generierung zu unterstützen. Beispielhaft werden *WebRatio Site Development Studio*, *VisualWADE*, sowie *OpenUWE Suite* in der Folge etwas genauer beschrieben.

WebRatio

Das *WebRatio Site Development Studio (WebRatio*, http://www.webratio.com) ist ein modellbasiertes Entwicklungswerkzeug, das auf der Modellierungsmethode *Web Modelling Language (WebML)* aufbaut. Das Werkzeug unterstützt die ER-Notation und UML, sowie eine eigene Notation zur Hypertextmodellierung. Der Code-Generator transformiert die in XML repräsentierten Content- und Hypertextmodelle mit Hilfe von XSL in die erforderliche Datenbankdarstellung und Datenbankanbindung, und erzeugt Softwarekomponenten und unterschiedliche Ausgabeformate (HTML, WML, PDF, Microsoft Word). Für die Generierung der Präsentation der Seiten bietet WebRatio *EasyStyle* an, um annotierte Seiten automatisch in XSL Stylesheets zu transformieren. Es sind dazu keine zusätzlichen Programmieraktivitäten erforderlich. Die von WebRatio generierte Web-Anwendung läuft in einem Laufzeit-Framework basierend auf einer Menge von objektorientierten Java- oder C#-Komponenten, die anhand von XML-Beschreibungsdateien konfigurierbar sind. Die Laufzeit-Architektur basiert auf dem MVC 2 Entwurfsmuster (siehe dazu auch Kapitel 4) und ist sowohl für die *Jakarta*

Struts Open-Source Plattform, als auch für die *JSP tag* Bibliotheken und die *Microsoft.NET Plattform* geeignet.

VisualWADE

Das Werkzeug *VisualWADE* (http://www.visualwade.com) basiert auf der Methode OO-H. Mit dem Werkzeug wird die Modellierung sowie die automatische Generierung der Anwendung basierend auf XML, ASP, JSP und PHP möglich. Das Werkzeug verknüpft ein UML-Modell mit zwei weiteren Modellen: der »Navigation View«, in der hypertextuelle Aspekte der Web-Anwendung modelliert werden und der »Presentation View«, in der Interaktionselemente der Benutzerschnittstelle in Hinblick auf Struktur und Verhalten mit Hilfe einer Reihe von Templatestrukturen dargestellt werden. Dadurch wird eine Endgeräte-unabhängige Beschreibung der Benutzerschnittstelle erreicht. Ausgehend von dieser Beschreibung kann mit Hilfe von Generatoren die Web-Anwendung für verschiedene Laufzeitumgebungen und für verschiedene Endgeräte automatisch generiert werden.

OpenUWE

Die Werkzeug-Suite *OpenUWE* (http://www.pst.informatik.uni-muenchen.de/projekte/openuwe) basiert auf der Methode UWE. Das Hauptmerkmal von OpenUWE ist eine offene Architektur, die auf etablierten Standards basiert. Diese Standards werden sowohl von open source Werkzeugen, als auch von kommerziellen Werkzeugen unterstützt. Bestandteile dieser Entwicklungsumgebung – deren Entwicklung noch nicht abgeschlossen ist – sind zur Zeit das Werkzeug ArgoUWE und das Framework UWEXML bestehend aus einem Model Checker, einen Layout-Editor und einen Code-Generator für das XML Publishing Framework *Cocoon* [ZiLa02]. Die gemeinsame Datenaustauschsprache innerhalb dieser Architektur basiert auf dem erweiterbaren UWE-Metamodell.

Die Modelle der Web-Anwendung werden direkt mit dem ArgoUWE Case-Tool erstellt. ArgoUWE unterstützt nicht nur die Notation UWE sondern auch das aktive Model Checking durch die Überprüfung der für die UWE-Modelle spezifizierten OCL-Constraints. Mit Hilfe des Abbildungsmechanismus, der zwischen dem Metamodell und dem UWE UML Profil definiert ist, können die Modelle auch mit jedem anderen Standard UML-Case-Werkzeug erstellt werden. In solch einem Fall können die Modelle nachträglich durch ein Model Checker-Werkzeug überprüft werden. Der technische Modelldatenaustausch beruht auf XMI (XML Metadata Interchange).

Die Content-, Hypertext- und Präsentationsmodelle einer Web-Anwendung werden mit einem Layoutmodell ergänzt, das die physischen Layout-

Eigenschaften wie Farben, Schriftgröße oder -typ der Benutzerschnittstellenelemente der Anwendung beschreibt. Weil diese Eigenschaften in UML visuell nur unzureichend beschrieben werden können, wurde ein XML Dokumentenformat zur Beschreibung der Struktur dieses physischen Layoutmodells gewählt. Das entsprechende XML-Dokument wird vom Layout-Editor generiert und fließt zusammen mit der Darstellung des logischen Präsentationsmodells in den Code-Generator ein, der die Web-Anwendung, z.B. XML- und XSLT-Dokumente oder HTML-Seiten, letztendlich generiert.

3.10. Ausblick

Es existiert eine große Anzahl unterschiedlicher Methoden für die Modellierung von Web-Anwendungen, bei deren Weiterentwicklung allerdings eine Konvergenz der Methoden verstärkt eintreten wird. Wie weit dies allerdings – analog zur Entwicklung von UML – zu einer »Unified Web-Modelling Language« führen wird, ist zurzeit noch nicht absehbar. Unbestritten ist jedoch der Trend zur Verwendung von UML als Notationssprache. Einige Methoden haben ihre bisher nicht oder nur teilweise UML-konforme Notation bereits UML-konform gemacht und definieren ein UML-Profil für die Methode.

Welche Methoden sich dauerhaft etablieren werden, wird nicht zuletzt durch die für die Modellierungsmethode angebotene Werkzeugunterstützung und dem zugrunde legenden Metamodell [KoKr03] entscheidend beeinflusst werden. Zukünftig werden Werkzeuge nicht nur die Notation, sondern auch den Prozess der Entwicklung entsprechend systematisch unterstützen. Das setzt allerdings voraus, dass die Methoden, im Gegensatz zu bisher, klare Richtlinien und Vorgehensweisen definieren. Dabei sind sicherlich agile Vorgehensweisen [Ambl02] zu berücksichtigen, die allerdings mit der Zielsetzung der (semi-) automatischen Generierung in Einklang gebracht werden müssen.

Auch die Neuerungen im Bereich der »Model-Driven Achitecture« werden Auswirkungen auf Weiterentwicklungen der Methoden und Werkzeuge haben, da gerade Web-Anwendungen mit verschiedenen heterogenen Laufzeitplattformen und Publishing Frameworks konfrontiert sind.

Ebenfalls von Interesse für die Wiederverwendung von Modellen wird die Modellierung von Familien von Web-Anwendungen sowie Web-Anwendungsframeworks sein. In diesem Bereich wurden mit *OOHDM frame* [SREL01] bereits erste Schritte gesetzt.

Zurzeit gilt der Einbeziehung von Workflow-Konzepten in die Modellierung von Web-Anwendungen besondere Aufmerksamkeit, um den vermehrten transaktionalen und interorganisationalen Anforderungen von Web-Anwendungen gerecht zu werden [BCC+03a].

Durch den immer stärkeren Einsatz von mobilen Endgeräten für Web-Anwendungen wird die Berücksichtigung des Kontexts der Web-Anwendung bereits bei der Modellierung immer wichtiger werden. Bisher existieren allerdings noch wenige Ansätze, die Kontextualität umfassend betrachten [KPRS03].

Mit der Erhöhung der Komplexität der Modelle wird auch die Frage nach ihrer Qualität und der Sicherstellung der Qualität virulent [CoMM02].

Die Einbeziehung von Web Services in eine modellbasierte Entwicklung von Web-Anwendungen bringt neue Herausforderungen, allen voran das Zusammenspiel einer Top-down Modellierung mit einer Bottom-up Integration existierender Dienste und eine adäquate Werkzeugunterstützung dafür [BCC+03b].