

Dynamische Generierung kontextadaptiver Benutzungsschnittstellen durch Modellinterpretation

Steffen Lohmann, J. Wolfgang Kaltz, Jürgen Ziegler

Institut für Informatik und Interaktive Systeme
Universität Duisburg-Essen
Lotharstrasse 65
47057 Duisburg
{lohmann, kaltz, ziegler}@interactivesystems.info

Abstract: Bei der modellbasierten Entwicklung von interaktiven Systemen ist die effiziente und homogene Integration von Benutzungsschnittstellen, die sich optimal in die jeweilige Interaktionssituation einfügen, mit zahlreichen Herausforderungen verbunden. Dieser Beitrag untersucht, inwieweit Kontextwissen methodisch fundiert in die Modellierung und den Prozess der automatischen Generierung von Benutzungsschnittstellen einbezogen werden kann, um diese an die kontextuellen Gegebenheiten anzupassen und die Benutzerinteraktion zu unterstützen. Dabei werden drei Teilbereiche unterschieden: die Auswahl situativ geeigneter Systemfunktionen, die Erstellung von an den Kontext angepassten Benutzungsschnittstellen für diese Funktionen und deren Parametrisierung. Ein ontologiebasiertes Modellierungsvorgehen und ein Framework für kontextadaptive Webanwendungen liefern die Grundlage für den hier beschriebenen Ansatz.

1 Einleitung

Ein zentrales Anliegen der modellbasierten Entwicklung von interaktiven Systemen ist die effiziente und konsistente Überführung von abstrakten, systembeschreibenden Modellen in lauffähigen Programmcode. Im Bereich der Webanwendungsentwicklung verfolgen mehrere Ansätze ein systematisches Vorgehen, bei dem ein konzeptuelles Modell den Ausgangspunkt für die Modellierung weiterer Facetten des Systems bildet, wie Navigations-, Präsentations- oder Prozessaspekten. Die Entwicklung kontextadaptiver Webanwendungen verlangt darüber hinaus weitere Modellierungsaktivitäten. Als viel versprechend in Hinblick auf Konsistenz und Effizienz erachten wir es, wenn aus diesen Modellen nicht nur (semi-)automatisch Programmcode generiert wird, sondern wenn sie vielmehr einen inhärenten Bestandteil der Systemarchitektur darstellen. Das von uns entwickelte CATWALK-Framework folgt diesem Designparadigma, indem Modelle die Systemgrundlage bilden und diese zur Laufzeit automatisch interpretiert werden, im Speziellen zur Generierung kontextadaptiver Webanwendungen (vgl. [Ka05]).

Das CATWALK-Framework und die zugrunde liegende Modellstruktur bilden den Ausgangspunkt für diesen Beitrag und werden einführend erläutert. Auf dieser Grundlage untersuchen wir anschließend, wie Systemfunktionen in Abhängigkeit ihrer situativen Relevanz ausgewählt werden können und wie sich für die Funktionen automatisch grafische Benutzungsschnittstellen generieren lassen, die durch Einbeziehung von Kontextwissen die Interaktion des Benutzers unterstützen. Zunächst jedoch führen wir verwandte Ansätze an und verdeutlichen die Motivation des hier vorgestellten Ansatzes.

2 Ontologiebasierte Modellierung und modellinterpretierende Architektur für kontextadaptive Webanwendungen

2.1 Verwandte Ansätze im Bereich der generativen Modellinterpretation

In letzter Zeit wurden verschiedene Ansätze präsentiert, die die automatische Generierung von Webanwendungen aus abstrakten Modellen zum Ziel haben. An dieser Stelle wollen wir kurz auf zwei Ansätze exemplarisch eingehen, die sich in diesem Zusammenhang mit dem Thema Kontextadaptivität auseinander gesetzt haben.

Im Umfeld der Web Modeling Language (WebML) [Ce00] entstand das CASE-Tool WebRatio [Ma05]. In WebML werden Entity-Relationship-Diagramme zur Modellierung verwendet. WebRatio integriert einen Codegenerator, der aus den WebML-Modellen JSP- bzw. ASP-Templates generiert. Auf Möglichkeiten der Berücksichtigung von Kontext in WebML wurde eingegangen [CDM03], jedoch in nur sehr eingeschränkter Form auf Ebene der Betrachtung ganzer Webseiten und nicht Teilbereichen davon. Wie Kontextwissen bei der Auswahl von situativ geeigneten Systemfunktionen und der Generierung entsprechender Benutzungsschnittstellen herangezogen werden kann, wird im Zusammenhang mit WebML nicht diskutiert.

In der UML-basierten Web Engineering (UWE)-Methode [Ko01] wird adaptiven Aspekten mit eigenen Benutzer- und Adaptionsmodellen Rechnung getragen. UWE verwendet als Modellierungssprache UML; die Modelle werden in XMI gespeichert. Für die Generierung von Webanwendungen aus den UWE-Modellen wurde das Webentwicklungs-Framework Apache Cocoon erweitert [KK02]. Jedoch bleiben Benutzer- und Adaptionsmodell bisher bei der Generierung unberücksichtigt. Ferner können die erzeugten Java-Klassen und XSLT-Stylesheets nicht direkt ausgeführt werden, sondern müssen vorerst manuell vervollständigt werden.

Allgemein ist festzustellen, dass sich Ansätze der modellbasierten Webanwendungsentwicklung bei der Betrachtung des Interaktionskontextes häufig nur auf Teilaspekte konzentrieren und adaptives Systemverhalten nur zu einem geringen Grad von den methodischen Modellierungsvorgehen berücksichtigt wird (vgl. auch [Ka03]). Wenn Kontext in Betracht gezogen wird, wirkt dieser im Allgemeinen nur sehr eingeschränkt auf den Prozess der Anwendungsgenerierung. Es interessiert zumeist in erster Linie, inwieweit die Inhalte oder Navigation einer Webanwendung an den Kontext angepasst werden

können. Der modellbasierten, automatischen Auswahl von situativ geeigneten Systemfunktionen und der dynamischen Generierung kontextadaptiver Benutzungsschnittstellen für solche Funktionen wurde bisher nur wenig Aufmerksamkeit zuteil.

2.2 Ontologiebasierte Modellierung

Unser Ansatz setzt auf das dem WISE-Projekt [WI06] zugrunde liegende Vorgehen der ontologiebasierten Modellierung auf. Im Gegensatz zu anderen Entwicklungsvorgehen, bietet die ontologiebasierte Softwareentwicklung weitergehende semantische Ausdrucksmöglichkeiten bei Modellierung und Modellaustausch (vgl. [He04]), was insbesondere mit Blick auf die interoperable Integration von Kontextwissen in verteilten Systemen Vorteile verspricht. Die Grundlage unseres Ansatzes bildet eine Modellablage, die aus folgenden Modellen besteht (vgl. Abb. 1):

- Eine Domänenontologie, die Konzepte und Konzeptrelationen der Anwendungsdomäne beschreibt und auf Ressourcen referenziert, die von der Anwendung verwendet werden.
- Mehrere Kontextontologien, die Konzepte und Konzeptrelationen des Interaktionskontextes beschreiben, die für adaptives Systemverhalten von Relevanz sind.
- Ein Kontextrelationenmodell, das die kontextuellen Einflüsse definiert, z.B. mittels Relationen zwischen Konzepten der Domänenontologie und Konzepten der Kontextontologien.
- Ein Navigations-, ein Sichten- und ein Präsentationsmodell je mit Adaptionsspezifikationen, in denen Regeln für adaptives Systemverhalten auf Basis der Ontologiekonzepte und Kontextrelationen definiert sind.

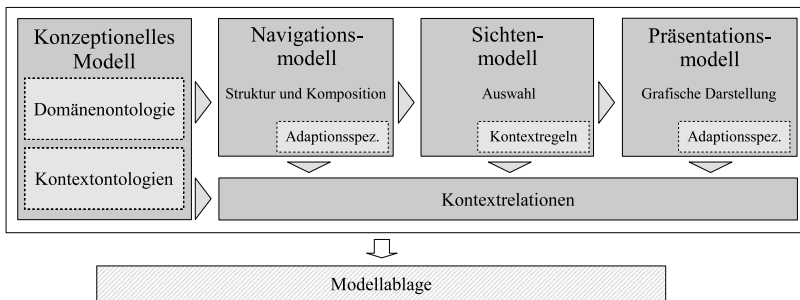


Abbildung 1: Kontextmodellierung in der WISE Methodologie

2.3 CATWALK-Framework

CATWALK ist ein komponentenorientiertes Framework, das die Modelle zur Laufzeit interpretiert, um hieraus eine kontextadaptive Webanwendung zu generieren. Abb. 2 zeigt die Architektur von CATWALK, das auf Apache Cocoon aufsetzt. Jede Client-Anfrage wird in einer Cocoon-Pipeline von Komponenten, die für die Anwendungsgenerierung zuständig sind, schrittweise verarbeitet. Die Komponenten

implementieren dem Designprinzip der *Separation of Concerns* folgend je einen spezifischen Aspekt. Sie sind in Java implementiert und verwenden weitere Artefakte (wie z.B. XSLT-Stylesheets für XML-Transformationen). Auf das Modellablage wird in jedem Generierungsschritt über ein Cocoon Pseudoprotokoll zugegriffen und benötigte Teile der Modelle werden zur Laufzeit interpretiert. Eine zentrale Komponente (die *Adaptation Engine*) koordiniert die Systemadaptation, indem sie Kontextrelationen und Adaptationsspezifikationen interpretiert und mit dem jeweils aktuellen Kontextzustand (der vom *Context Manager* verwaltet wird) abgleicht. Im letzten Schritt wird vom *XSL Transformer* eine Ausgabe generiert (i.d.R. eine Webseite), die an die Geräteeigenschaften des Clients und die jeweilige kontextuelle Situation angepasst ist.

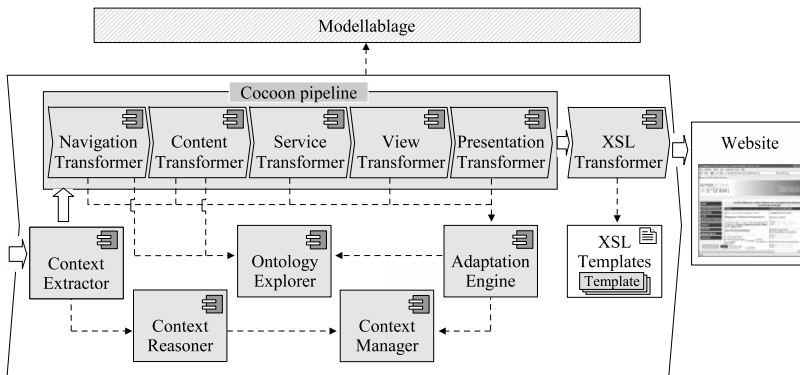


Abbildung 2: CATWALK Komponentenarchitektur

3 Generierung kontextadaptiver Benutzungsschnittstellen

Wissen über den Interaktionskontext wird in verschiedenen Teilbereichen bei der Generierung von Benutzungsschnittstellen berücksichtigt: zunächst bei der Auswahl von situativ geeigneter Funktionalität, für die in der aktuell aufgerufenen Webseite Benutzungsschnittstellen angeboten werden sollen, dann bei der Entscheidung, wo und wie die Schnittstellenelemente angezeigt werden sollen und schließlich bei der Bestimmung von Parameterwerten für die Benutzungsschnittstellen, die bereits vorausgewählt sein sollen. Möglichkeiten der Kontextadaptation in all diesen Teilbereichen werden im Folgenden näher erläutert. Zur Veranschaulichung wird ein Szenario beschrieben, in dem ein Betreiber eines Automobil-Service-Portals einen Dienst zur Reservierung von Mietfahrzeugen anbietet.

3.1 Repräsentation von Systemfunktionen

Die CATWALK-Architektur verfolgt einen serviceorientierten Ansatz, bei dem von der Webanwendung bereitgestellte Funktionalität in Web Services gekapselt wird. Systemfunktionen werden konzeptuell in ihre einzelnen Operationen unterteilt und jede

Operation erhält einen Eintrag in der Domänenontologie. Über ontologische Relationen werden die Operationen miteinander verbunden; einer der Einträge, der sog. Funktionsrepräsentant, steht stellvertretend für die gesamte Systemfunktion. Wissen über die Abhängigkeiten zwischen den einzelnen Operationen einer Systemfunktion ist somit in der Domänenontologie repräsentiert.

Angenommen, die Reservierungsfunktion des Beispielszenarios besteht aus vier Operationen, die von zwei verschiedenen Web Services realisiert werden – einer zur Auswahl (1) des gewünschten Fahrzeugtyps und (2) dessen Ausstattung, ein anderer für (3) Buchung und (4) Bezahlvorgang. Für jede Operation wird dann ein Ontologieeintrag erstellt, der auf die entsprechende WSDL-Beschreibung referenziert (und deren `<PortType>`- und `<operation>`-Tag). Zusätzlich können einzelne Parameter der Funktion gesondert in der Domänenontologie erfasst werden (vgl. Abschnitt 3.4).

3.2 Auswahl situativ geeigneter Systemfunktionen

Im Navigationsmodell werden Relationen zwischen Elementen der Domänenontologie definiert und so die Navigationsstruktur der Webanwendung erstellt. Einträge von Funktionsrepräsentanten fließen – wie andere Domänenelemente auch – in die Modellierung der Navigationsstruktur mit ein. Zusätzlich lässt sich durch die Definition von Kontextrelationen zwischen einem Eintrag eines Funktionsrepräsentanten und Konzepten der Kontextontologien die Erreichbarkeit einer Systemfunktion in Abhängigkeit zur kontextuellen Situation festlegen. Adaptionsspezifikationen im Navigationsmodell definieren das adaptive Systemverhalten, das ausgeführt werden soll, wenn über Kontextrelationen verbundene Konzepte der Kontextontologien aktiviert sind. Ob für eine Systemfunktion in der aktuellen Ausgabe (Webseite) eine Benutzungsschnittstelle angeboten wird, wird dann nicht mehr ausschließlich durch die Position des Anwenders in der Navigationsstruktur determiniert, sondern zusätzlich durch die jeweiligen kontextuellen Gegebenheiten beeinflusst. Dies bedeutet, dass zwei verschiedenen Benutzern nicht notwendigerweise dieselbe Funktionalität angeboten wird, nur weil sie einen identischen Navigationsweg durch die Webanwendung gewählt haben.

Nehmen wir an, ein Nutzer hat die Startseite des Automobil-Service-Portals aufgerufen. Im Navigationsmodell wurde eine Relation zwischen den Domäneneinträgen der Startseite und des Funktionsrepräsentanten der Reservierungsfunktion modelliert. Zusätzlich wurde im Navigationsmodell für diese Relation eine Adaptionsspezifikation festgelegt, die definiert, dass die Reservierungsfunktion angezeigt werden soll, wenn sie über Kontextrelationen verfügt und die verbundenen Kontextkonzepte aktiviert sind. Schließlich wurde im Kontextrelationenmodell eine Relation zwischen dem Eintrag des Funktionsrepräsentanten für die Reservierungsfunktion und dem Konzept „Besitzt Auto“ der Benutzerkontextontologie modelliert. Das Konzept „Besitzt Auto“ soll aktiviert sein, wenn der Benutzer über ein Auto verfügt und deaktiviert, falls nicht. Aufgrund dieser Modellierungen wird einem Besucher des Portals, der kein Auto besitzt, die Reservierungsfunktion direkt auf der Startseite angeboten, wohingegen Autobesitzer diese Funktion beispielsweise ausschließlich per Navigation erreichen.

3.3 Generierung von Benutzungsschnittstellen

Nachdem im ersten Schritt bestimmt wurde, für welche Systemfunktionen in der aktuellen Ausgabe Benutzungsschnittstellen angeboten werden sollen, wird im nächsten Schritt ermittelt, wie diese dem Anwender präsentiert werden. In Abhängigkeit der Relevanz, die Systemfunktionen in der jeweiligen Situation besitzen und der kontextuellen Gegebenheiten (wie z.B. Einschränkungen bei den Darstellungsmöglichkeiten des Client-Geräts), bieten sich verschiedene Varianten für die Präsentation der Benutzungsschnittstellen an.

Die CATWALK-Architektur sieht hierfür die Definition verschiedener GUI Design Patterns vor. Jedes Design Pattern besteht aus einer XSLT-Datei und optional einem CSS-Stylesheet und wird durch einen Eintrag in der Domänenontologie repräsentiert, der auf die URIs dieser Dateien referenziert. In einer ähnlichen Vorgehensweise wie auch bei der Navigationsmodellierung werden im Präsentationsmodell Relationen zwischen Design Pattern-Einträgen und anderen Elementen der Domänenontologie (z.B. dem Funktionsrepräsentanten einer Systemfunktion) modelliert sowie im Kontextrelationenmodell Relationen zwischen Design Pattern-Einträgen und Konzepten der Kontextontologien. Adaptionsspezifikationen im Präsentationsmodell regeln, unter welchen Bedingungen ein Design Pattern ausgewählt wird. Zur Laufzeit werden die für die jeweilige Operation der Systemfunktion nötigen Teile der im Domänenmodell referenzierten WSDL-Beschreibung eingelesen. In Abhängigkeit der kontextuellen Situation wird dann ein entsprechendes Design Pattern angewandt, das dynamisch die Benutzungsschnittstelle für die Systemfunktion erzeugt.

Eine provisorische Umsetzung für das Mietwagen-Szenario zeigt Abbildung 3. Es sei angenommen, die Reservierungsfunktion wird in mehreren Schritten dargeboten. Im ersten Schritt hat der Anwender bereits den Fahrzeugtyp ‚PKW Mittelklasse‘ ausgewählt; im zweiten Schritt soll er nun das genaue Mittelklassemodell, die Fahrzeugausstattung sowie Abhol- und Abgabedatum, -zeit und -ort angeben. Im Beispielszenario greift der Nutzer über einen Desktop PC auf die Webanwendung zu – das entsprechende Konzept in der Gerätekontextontologie ist aktiviert. Aufgrund der definierten Kontextrelationen und der Adaptationsspezifikation des Präsentationsmodells wird ein Design Pattern gewählt, das sich zur Verwendung für Desktop PCs eignet. In gleicher Weise ließen sich für andere Client-Geräte wie PDAs oder Mobiltelefone weitere Design Patterns und entsprechende Kontextrelationen und Adaptionsspezifikationen definieren. Auch können unterschiedliche Design Patterns für verschiedene Nutzer angewandt werden (z.B. für sehbehinderte Anwender ein CSS-Stylesheet, das besser sichtbare Schnitstellenelemente erzeugt) oder je nach situativer Relevanz einer Systemfunktion auf verschiedene Design Patterns zurückgegriffen werden.

Die Benutzungsschnittstelle für den zweiten Schritt wird aus der WSDL-Beschreibung der entsprechenden Web Service-Operation(en) erzeugt. Abhängig vom Datentyp und der Anzahl an Auswahlalternativen rendert das XSLT-Stylesheet für jeden Parameter ein passendes XHTML-Formularelement zur Auswahl bzw. Eingabe des Rückgabewerts (Wechselwirkungen zwischen Parametern sollen an dieser Stelle

unberücksichtigt bleiben). Der Name der Parameter wird als Bezeichner verwendet, erlaubte Rückgabewerte werden aus den XML-Schema-Definitionen abgeleitet.

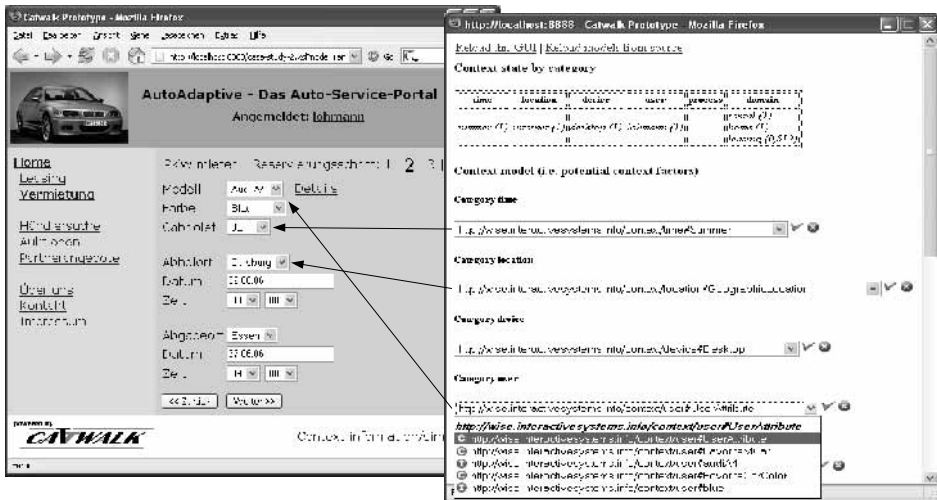


Abbildung 3: Unter Einbeziehung von Kontextwissen generierte und parametrisierte Benutzungsschnittstelle

3.4 Parametrisierung der Benutzungsschnittstellen

Um dem Anwender eine initiale Hilfestellung bei der Interaktion mit Systemfunktionen anzubieten, kann Kontextwissen darüber hinaus zur Parametrisierung der generierten Benutzungsschnittstellen herangezogen werden. Hierfür werden im Kontextrelationsmodell Relationen zwischen Parametern von Systemfunktionen der Domänenontologie und Konzepten der Kontextontologien modelliert. Im abgebildeten Beispiel wurde der Parameter ‚Modell‘ mit dem vom Benutzer favorisierten Mittelklassewagen und der Parameter ‚Farbe‘ mit der Lieblingsautofarbe des Benutzers gemappt sowie der Parameter ‚Cabriolet‘ mit der Jahreszeit Sommer und der ‚Abholort‘ mit dem aktuellen Aufenthaltsort des Benutzers. In Abhängigkeit der jeweiligen kontextuellen Situation sind diese Konzepte unterschiedlich instanziiert und führen zu verschiedenen Werten für die Parameter. Insofern ein ermittelter Wert zu den erlaubten Rückgabewerten zählt, wird er in der Benutzungsschnittstelle als Defaultwert gesetzt (z.B. durch Vorselektion im Drop-Down-Listenfeld). Ist ein gemapptes Kontextkonzept nicht instanziiert (weil es beispielsweise nicht ermittelt wurde) oder weicht die Instanz von den erlaubten Rückgabewerten ab, geschieht keine Vorauswahl.

Allgemein besteht die Gefahr, dass fehlerhaftes Mapping oder falsch ermittelter Kontext den Benutzer nicht bei der Interaktion unterstützen, sondern ihn verwirren. Wäre im Beispielszenario aufgrund von falsch ermitteltem Ortskontext die Stadt Essen anstelle

von Duisburg vorselektiert, würde der Benutzer eventuell fälschlicherweise davon ausgehen, dass in Duisburg keine Abholstation des Autovermieters existiert. Auch wäre die Integration einer Validitätsbewertung von Kontextinformationen sinnvoll.

4 Fazit

In diesem Beitrag haben wir einen Ansatz für die Integration von Kontextwissen in die dynamische Auswahl, Generierung und Parametrisierung situativ geeigneter Benutzungsschnittstellen vorgestellt. Der Ansatz basiert auf einem ontologiebasierten Modellierungsvorgehen und verwendet ein Framework, das die Modelle zur Laufzeit interpretiert. Im Gegensatz zu einem Großteil verwandter Ansätze wird Kontext sehr generisch betrachtet und von Anfang an in das methodische Modellierungsvorgehen integriert. Der vorgestellte Ansatz ist unabhängig von spezifischen Verfahren zur Erhebung von Kontextinformationen. Auch wenn Kontextadaptivität zunächst einen Mehraufwand bei der Modellierung voraussetzt, wurde gezeigt, wie sich durch die Berücksichtigung von Kontextwissen in verschiedenen Teilbereichen bei der Generierung von Benutzungsschnittstellen die Benutzerinteraktion unterstützen und eine verbesserte Usability erzielen lässt.

Literaturverzeichnis

- [Ce00] Ceri, S.; Fraternali, P.; Bongio, A.: Web Modeling Language (WebML): a Modelling Language for Designing Web Sites. *Computer Networks*, 33(1–6), 2000; S. 137-157.
- [CDM03] Ceri, S.; Daniel, F.; Matera, M.: Extending WebML for Modelling Multi-channel Contextaware Web Applications. In: *Proceedings of WISE - MMIS'03 Workshop*, IEEE Computer Society, 2003; S. 615-626.
- [He04] Hesse, W.; Krzensk, B.: Ontologien in der Softwaretechnik. *Proceedings des Workshops "Ontologien in der und für die Softwaretechnik" bei der Modellierung 2004 in Marburg*. GI / Univ. Marburg, 2004.
- [Ka05] Kaltz, J.W.; Lohmann, S.; Lang, E.; Hussein, T.; Ziegler, J.: Ontologiebasiertes Engineering kontextadaptiver Webanwendungen. *i-com – Zeitschrift für interaktive und kooperative Medien*, 3/2005, 2005; S. 22-30.
- [Ka03] Kappel, G.; Pröll, B.; Retschitzegger, W.; Schwinger, W.: Customisation for Ubiquitous Web Applications - a Comparison of Approaches. *International Journal of Web Engineering Technology* 1, 2003; S. 79-111.
- [KK02] Kraus, A.; Koch, N.: Generation of Web Applications from UML Models using an XML Publishing Framework. In: *Proceedings of the 6th World Conference on Integrated Design and Process Technology (IDPT)*, volume 1, 2002.
- [Ko01] Koch, N.: Software Engineering for Adaptive Hypermedia Systems: Reference Model, Modeling Techniques and Development Process. PhD thesis, Ludwig-Maximilians-Universität München, 2001.
- [Ma05] Manolescu, I.; Brambilla, M.; Ceri, S.; Comai, S.; Fraternali, P.: Model-driven Design and Deployment of Service-enabled Web Applications. *ACM Transactions on Internet Technology*, 5(3), 2005; S. 439-479.
- [WI06] WISE - Web Information and Service Engineering. Informationen zum Projekt erreichbar unter: <http://www.wise-projekt.de>