Eine komponentenorientierte Architektur für die kontext-sensitive Adaption von Web-Anwendungen

J. Wolfgang Kaltz, Steffen Lohmann, Jürgen Ziegler[1]

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

Abstract: In diesem Beitrag wird ein Context Engineering-Ansatz vorgestellt, der die Integration von Methoden der ontologiebasierten Kontextmodellierung bei der Entwicklung von Web-Anwendungen systematisiert und anhand einer Systemarchitektur erläutert, wie geeignete Kontextkomponenten durch die Verwendung dieser Modelle Adaptionen durchführen, die den Benutzer bei der Suche nach relevanten Informationen und Diensten unterstützen.

1 Einleitung

Um die zunehmende Informationsflut, der die Benutzer interaktiver Systeme ausgesetzt sind, zu begrenzen, wird immer häufiger der Einsatz von Systemkomponenten vorgeschlagen, die Informationen nach Relevanzgesichtspunkten automatisiert bewerten und dem Benutzer strukturiert und gezielt darbieten. Eine Möglichkeit besteht darin, die Inhalte und Dienste sowie Navigationsstrukturen in Abhängigkeit vom Interaktionskontext anzupassen. Eine solche Kontextadaptivität von Systemen ist gegenwärtig Gegenstand zahlreicher Forschungsarbeiten (z.B. [De04], [Sc02]), doch mangelt es bislang an einer einheitlichen, umfassenden Context Engineering-Methodik, die die generelle Berücksichtigung von Kontext im Entwicklungszyklus von Systemen integriert.

Im Folgenden stellen wir eine Systemarchitektur vor, die Kontextmodellierungsmethoden in einem Web Engineering-Prozess [Mu01] systematisch verwendet. Wir erläutern zunächst, wie Kontext in der Modellierungsphase von Web-Anwendungen berücksichtigt werden kann und zeigen anschließend, auf welche Weise Kontextkomponenten zur Laufzeit des Systems auf diese Kontextmodellierungen zugreifen und sie für die Informationsauswahl und zur Bestimmung von Systemadaptionen verwenden.

---

[1] Diese Arbeit wird unterstützt durch das Bundesministerium für Bildung und Forschung (BMBF) unter der Projektnummer 01ISC30F (Projekt WISE).

Grundlage hierfür bildet eine Vorgehensweise, die wir derzeit im Projekt WISE entwickeln. Die vorgestellte Architektur berücksichtigt besonders den Einsatz von Verfahren und Technologien, die einen hohen Grad an Generalisierbarkeit, Interoperabilität und Potenzial zur Wiederverwendung besitzen.

2 Kontextmodellierung im Web Engineering-Prozess

2.1 Domänenontologie und Navigationsstruktur

Die hier vorgestellte Vorgehensweise zur Modellierung von Kontextbeziehungen erweitert den Ansatz der ontologiebasierten Navigation [CR00] und verwendet im Speziellen eine Unterteilung, die von Wissen & Ziegler [WZ03] für die Modellierungsphasen innerhalb des Web Engineering-Prozesses vorgeschlagen wurde. Diese unterscheidet zwischen konzeptionellem, Navigations-, Präsentations- und Sichtenmodell.

Den Ausgangspunkt für alle weiteren Modellierungen bildet eine Domänenontologie, in der der Informationsbestand repräsentiert ist. Sie ist der zentrale Teil des konzeptionellen Modells und ihre Erstellung ist der notwendige erste Schritt in der Modellierungsphase des Web Engineering-Prozesses, gegebenenfalls unter Rückgriff auf eine bereits vorhandene Domänenontologie. Die Entwicklung einer solchen Ontologie erfolgt in der Regel manuell, eventuell unterstützt durch automatisierte Analysen von Ressourcenkorpora. Die Themenstruktur des konzeptionellen Modells bildet somit ein semantisches Netz über die Ressourcen des Informationsbestands der Web-Anwendung.

Aus den Konzepten und Inhalten der Domänenontologie werden in der Navigationsmodellierungsphase sowohl Navigationsstrukturen als auch Informationen zur Komposition einzelner Website-Elemente zu einer Gesamtseite erzeugt (vgl. [WZ03]).

2.2 Kontextmodellierung

Ziel der Kontextmodellierung ist es, Adaptionsleistungen in der resultierenden Web-Anwendung auf verschiedenen Ebenen zu unterstützen: Konzeptauswahl, Content Erzeugung, Navigation, Sichten und Darstellung. In einer ersten Phase werden durch die Analyse relevanter Kontextaspekte Kontextontologien erstellt, die in verschiedene Teilontologien gegliedert sind. Für ihre Erstellung können vordefinierte Taxonomien verwendet und gegebenenfalls erweitert werden. Anschließend werden adaptionsrelevante Konzepte der Domänenontologie mit Konzepten der Kontextontologien in Beziehung gesetzt und so die Domänenontologie um Kontextualisierungsrelationen erweitert. Eine geeignete Taxonomie der Kontextaspekte sowie verschiedene Relationstypen und Mechanismen der Kontextualisierung beschreiben wir in [ZLK05].

In der anschließenden Navigationsmodellierungphase wird auf die Kontextontologien zurückgegriffen, um die Verwendung geeigneter Navigationsstrukturen und Kompositionen für unterschiedliche Kontexte zu definieren. Das resultierende Navigationsmodell wird im nächsten Schritt um das Sichtenmodell erweitert. Dieses beschreibt die Sichtbarkeit der einzelnen Elemente der Website-Komposition in Abhängigkeit zur jeweiligen Kontextsituation. Auch für diese Kontextualisierung wird das Kontextmodell herangezogen.

Der letzte Modellierungsschritt schließlich betrifft die Kontextualisierung der grafischen Darstellung der Website. An dieser Stelle wird erneut auf das konzeptionelle Kontextmodell Bezug genommen, um verschiedene grafische Darstellungsvarianten sowohl der Inhalte, Dienste und Navigation als auch des gesamten Erscheinungsbilds der Website in Abhängigkeit des Kontextzustands zu definieren.

Die erarbeiteten Modelle aller vier Phasen werden im WISE-Repository in strukturierter Form gemäß dem WISE-Schema abgelegt (siehe Abbildung 1).

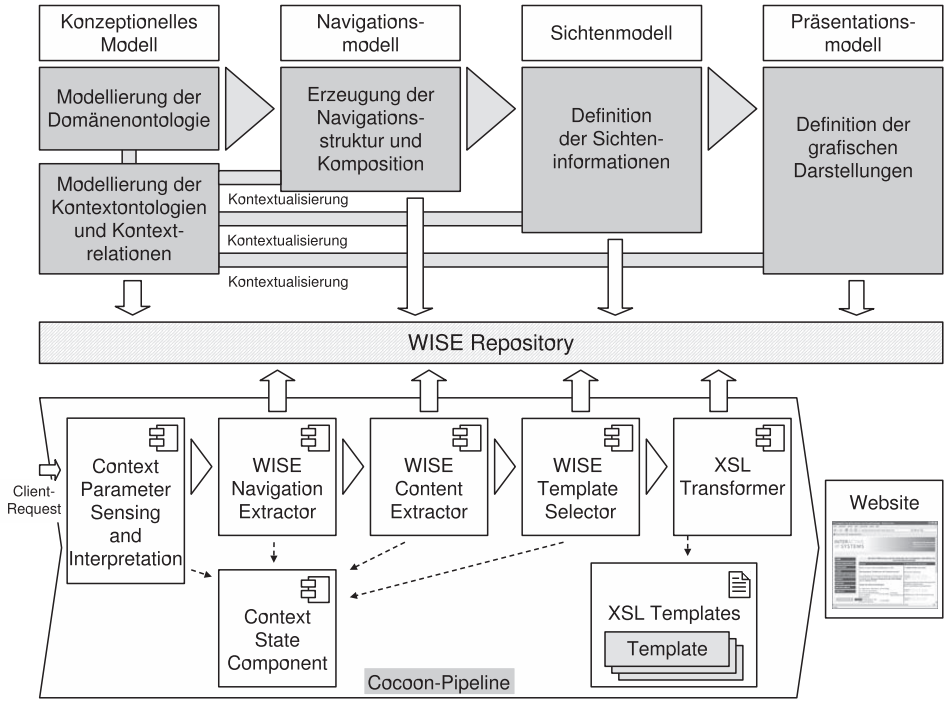

Abbildung 1: Die modellierten Kontextrelationen bilden die Grundlage für die Adaptionen des Systems

3 Softwarearchitektur und geeignete Technologien

Die im Folgenden erläuterte Softwarearchitektur verwendet die im WISE-Repository abgelegten Modelle, um geeignete Adaptionen zur Laufzeit des Systems entsprechend dem Interaktionskontext vorzunehmen. Sie bildet die Grundlage für einen Prototypen, den wir gegenwärtig im WISE-Projekt entwickeln.

Um einen hohen Grad hinsichtlich Generalisierbarkeit und Wiederverwendbarkeit der implementierten Softwarearchitektur zu erreichen, gestalten wir diese komponentenorientiert und benutzen durchgängig offene Webstandards in Kombination mit Java-Komponenten. Informationsrepräsentation und Manipulation basieren auf XML, die Steuerungslogik übernimmt das XML Web Publishing Framework Apache Cocoon [MA02].

Jede Kontextadaption stellt einen Response auf einen Client-Request dar, der in der sog. Cocoon-Pipeline verarbeitet wird. Verschiedene Komponenten übernehmen hierbei unterschiedliche Adaptionsfunktionen, die jeweils einen bestimmten Adaptionsaspekt berücksichtigen.

Zunächst muss der Kontextzustand über Sensing-Mechanismen dem System bekannt gemacht werden. In Web-Systemen stehen hierfür in der Regel lediglich Informationen zur Verfügung, die entweder über den aktuellen Request des Clients das System erreichen oder in vergangenen Requests gesendet wurden. Das gesamte Kontextwissen resultiert aus expliziten Informationen des Clients und gegebenenfalls weiteren, über Reasoning-Verfahren erschlossenen, impliziten Informationen. Die Informationen über den aktuell aktiven Kontext werden von der *Context State Component* verwaltet, die bei jedem Request Kontextinformationen erhält, diese kategorisiert und das bisher gesammelte Wissen bezüglich des Kontextzustands hierum erweitert.

Dieser Kontextinformationen bedient sich zunächst der *WISE Navigation Extractor*, dessen Aufgabe es ist, eine geeignete Kompositions- und Navigationsstruktur der Website in Form eines XML-Stroms zu erzeugen. Hierfür greift er auf die im WISE-Repository abgelegten Navigations- und Sichtenmodelle zu und wendet Navigationsadaptionen auf Grundlage der Betrachtung dieser Modelle und der ermittelten Kontextinformationen an. Den resultieren XML-Strom leitet er an den *WISE Content Extractor* weiter, der erneut auf die *Context State Component* und das *WISE-Repository* zugreift. Diesmal werden die im konzeptionellen Modell erstellten Kontextrelationen betrachtet und hierüber die Relevanz des jeweiligen Contents ermittelt. Relevanter Content wird anschließend mit dem XML-Strom, der die Navigationsstruktur-Informationen enthält, aggregiert. Der *WISE Template Selector* entscheidet anschließend anhand der modellierten kontextspezifischen grafischen Darstellungsvarianten und der Kontextinformationen der *Context State Component*, welche Templates für die Darstellung verwendet werden.

Die genannten Komponenten entsprechen dem Cocoon-Komponentenmodell und sind in Java implementiert. Beim *XSL Transformer* handelt es sich um eine Standard-Cocoon-Komponente, die auf einen eingehenden XML-Strom ein XSL-Template anwendet.

In diesem Fall erweitert er die Navigations- und Contentinformationen um die von der *WISE Template Selector Action* festgelegten Darstellungsinformationen und gibt als Resultat ein vom Kontext abhängendes Format, wie z.B. XHTML oder WML, aus.

Der Pipeline-Ansatz ermöglicht die Integration beliebiger Informationen aus externen Systemen oder auch von Web Services, da der *WISE Content Generator* seine Daten als XML-Strom empfängt. So können potentiell die unterschiedlichsten, heterogenen Informationstypen und auch Services berücksichtigt werden, sofern sie vergleichbar den textuell repräsentierten Informationen in der Domänenontologie abgebildet werden. Des Weiteren unterstützt diese Architektur sämtliche Arten von Clients, vorausgesetzt ein geeignetes Template für die Darstellungsgenerierung ist verfügbar. Auf diese Weise bietet die hier vorgeschlagene Architektur eine hohe Interoperabilität.

4 Ausblick

In diesem Beitrag wurde anhand eines Context Engineering-Ansatzes und einer darauf abgestimmten Systemarchitektur veranschaulicht, dass eine systematische Modellierung von Kontextinformationen gewinnbringend eingesetzt werden kann, sofern entsprechende Komponenten die Modelle für sinnvolle Systemadaptionen verwenden. Gegenwärtig arbeiten wir deshalb an der Konzeption von Werkzeugen, die die Kontextmodellierung unterstützen, und an der Implementierung der dargelegten Softwarearchitektur.

Literaturverzeichnis

[CR00] Crampes, M.; Ranwez, S.: Ontology-supported and ontology-driven conceptual navigation on the World Wide Web. In: Hypertext '00: Proceedings of the eleventh ACM on Hypertext and hypermedia, ACM Press, 2000; S. 191-199.

[De04] Dey, A.K.; Hamid, R.; Beckmann, C.; Li, I.; Hsu, D.: a CAPpella: programming by demonstration of context-aware applications. In: Proceedings of the 2004 conference on Human factors in computing systems, CHI'04, ACM Press, 2004; S. 33-40.

[MA02] Moczar, L.; Aston, J.: Cocoon Developer's Handbook. Sams, 2002.

[Mu01] Murugesan, S., Deshpande, Y., Hansen, S., Ginige, A.: Web engineering: A new discipline for development of web-based systems. In (Murugesan, S.; Desphande, Y., Hrsg.): Web Engineering. Managing Diversity and Complexity of Web Application Development. Springer-Verlag, Heidelberg, 2001.

[WZ03] Wissen, M.; Ziegler, J.: Ontologiebasierte Vorgehensweise zur Modellierung komponentenorientierter Web-Anwendungen. In (Ziegler, J.; Swillus, G., Hrsg.): Mensch & Computer 2003: Interaktion in Bewegung. Teubner, Stuttgart, 2003; S. 31-42.

[Sc02] Schmidt, A.; Van Laerhoven, K.; Strohbach, M.; Friday, A.; Gellersen, H.W.: Context Acquistion based on Load Sensing. In (Boriello, G.; Holmquist, L.E., Hrsg.): Ubiquitous Computing. Proceedings of UbiComp 2002, LNCS 2498, Springer-Verlag, Heidelberg, 2002; S. 333–351.

[ZLK05] Ziegler, J.; Lohmann, S., Kaltz, J.W.: Kontextmodellierung für adaptive webbasierte Systeme. In (Stary, C., Hrsg.): Mensch & Computer 2005: Kunst und Wissenschaft – Grenzüberschreitungen der interaktiven ART. Oldenbourg Verlag, München, 2005, angenommen.