A. M. Heinecke, H. Paul (Hrsg.): Mensch & Computer 2006: Mensch und Computer im Struktur*Wandel*.
München, Oldenbourg Verlag, 2006, S. 43-53

Modellierung aufgabenangemessener Abläufe im Web

Gerd Szwillus

Universität Paderborn, Institut für Informatik

Zusammenfassung

Moderne Webseiten bieten typischerweise zahlreiche Möglichkeiten der Interaktion. Neben Einkaufs- und Bestellvorgängen gibt es interaktive Webanwendungen für viele Zwecke, wie zum Beispiel zum Abwickeln von Auktionen, zum Verwalten von Digitalfotos oder zur Berechnung von Finanzierungsmodellen. Derartige Abläufe im Web anzubieten, erfordert eine angemessene Entwurfsmethodik, die den zu beschreibenden Sachverhalten und Abhängigkeiten gerecht wird, andererseits aber auch die besondere Nutzungssituation im Web berücksichtigt. Wir schlagen ein Modellierungskonzept vor, das sich an der klassischen Aufgabenmodellierung orientiert und dieses Konzept derart erweitert, so dass die Veränderungen durch die Aufgabenerledigung in ihrer Umgebung explizit einbezogen werden. Auf diese Weise können Ablaufstrukturen für das Web entwickelt, bewertet und optimiert werden, ohne dass der aufwendige Schritt zur visuellen Gestaltung nötig wird.

1 Einleitung

In den Anfängen des WWW war die Präsentation von Information die Hauptaufgabe von Webauftritten. Diese rein datenzentrierte Sicht ist allerdings zwischenzeitlich eher zur Ausnahme geworden: in der Regel bieten Webseiten zahlreiche Möglichkeiten der Interaktion, was zur Implementation von Abläufen aller Art intensiv genutzt wird. Da man üblicherweise große Benutzerzahlen erreichen will, müssen die angebotenen Abläufe möglichst einfach zu benutzen sein und trotzdem viele Möglichkeiten bieten, um Dienstleistungen erfolgreich ins Web verlagern zu können. Das Problem einer aufgabengerechten Gestaltung von Benutzungsschnittstellen, wie vielfach bereits für Desktop-Systeme gefordert, stellt sich hier unter verschärften Bedingungen neu.

Im mittlerweile als „visuelles Medium“ geltenden Web werden Abläufe aller Art dem Besucher eines Webauftritts letztlich als Abfolge zumeist grafisch aufwendig gestalteter Seiten entgegentreten. Bekanntermaßen muss erheblicher Aufwand für deren Entwicklung getrieben werden, bei der die visuellen Fragen vielfach in den Vordergrund drängen, bzw. dieser Entwurfsaspekt als erster und somit am wichtigsten erscheinender Aspekt behandelt wird. An

manchen Abläufen, die im WWW eingesetzt und den Benutzern zugemutet werden, erkennt man, dass die Entwicklung des Ablaufs selbst hierbei eher vernachlässigt wird.

Wir schlagen in diesem Papier ein Modellierungskonzept vor, das sich an der klassischen Aufgabenmodellierung orientiert. Hierbei steht die Sichtweise des agierenden Menschen im Vordergrund. Allerdings wird mit einbezogen, welche Veränderungen die Aufgabenerledigung in ihrer Umgebung bewirkt. Diese Vorgehensweise erlaubt eine präzise Beschreibung der zu entwickelnden Abläufe aus Sicht der Erkenntnis- und Entscheidungsprozesse des Besuchers. Damit können die Ablaufstrukturen entwickelt, bewertet und optimiert werden, ohne dass der aufwendige Schritt zur visuellen Gestaltung nötig wird.

2 Aufgabenerledigung im Web

Zur Beschreibung von Aufgabenerledigung durch den Menschen haben sich Aufgabenmodelle (Mori et al. 2002), (Uhr 2003), (van Welie et al. 1998) in verschiedenen Entwurfssituationen bewährt. Diese drücken die Aufteilung von Aufgaben in Teilaufgaben, sowie typische zeitliche Relationen zwischen Aufgaben aus. Auf diese Denkweise wollen wir uns bei unserer Modellierungstechnik abstützen, allerdings zeigt sich schnell, dass diese Strukturen alleine nicht hinreichen, um zu aussagekräftigen Modellen zu kommen. Dialogabläufe im Web verwenden intensiv den Austausch von Informationen zwischen dem Besucher und der Website, was sich in Aufgabenmodellen nur unzureichend widerspiegelt. Um dies zu verdeutlichen, betrachten wir im Folgenden einen Beispielablauf im Web und präsentieren anschließend ein für derartige Abläufe geeignetes Modellierungskonzept.

Das Benutzungsbeispiel beschreibt die Nutzung der Fahrplanauskunft der Deutschen Bahn (Stand: März 2006). Ausgehend von der Homepage (www.bahn.de) ist ein kleines Rechteck in der linken oberen Ecke der Website (Abb. 1) dem Einstieg in die Reiseauskunft gewidmet – der Großteil der Seite besteht aus bunten Angeboten, die den Besucher einfangen sollen. Unter dem Titel „Reiseauskunft – Tickets“ kann man den Abfahrtsort, den Zielort, das Reisedatum und die Abfahrts- bzw. Ankunftszeit einer eventuellen Reise eingeben. Unterstellen wir, dass der Benutzer die Aufgabe „Planen einer Bahnreise“ verfolgt, ist es möglicherweise legitim, jetzt eine Liste möglicher Zugverbindungen zu erwarten. Stattdessen wird der Besucher auf der Folgeseite mit einer großen Zahl von Einstell- und Eingabemöglichkeiten konfrontiert. Zumindest der unerfahrene Benutzer wird diese Seite von oben nach unten durchlesen. Dabei wird er zunächst erkennen, dass seine zuvor eingegebenen Reisedaten übernommen wurden. Er kann nun, wenn er möchte, die Reisedaten für die Rückreise eintragen. Darunter wird er nach der Anzahl der Reisenden, den vorhandenen BahnCards, sowie der Wagenklasse gefragt. Ihm wird als Verkehrsmittel „Standardsuche“ angeboten und er muss entscheiden, ob er „schnelle Verbindungen bevorzugen“ will und ob er ein Fahrrad mitnehmen möchte oder nicht. Danach stößt der Benutzer auf den Button „Verbindung suchen“, was ihm dann eine neue Seite mit einer Liste der Zugverbindungen einschließlich Preisangaben zur gewünschten Fahrzeit verschafft.

Abbildung 1: Fahrplanauskunft

Anhand dieser Liste kann der Besucher nun entscheiden, *welche* Hinfahrt er antreten will. Für diese Bewertungsaufgabe kann er Details über die Verbindungen anfordern und die Liste auf frühere oder spätere Verbindungen erweitern. Hat er genau eine dieser Verbindungen ausgesucht, kann er die Rückfahrt hinzufügen. Dazu wird ihm eine Terminierung der Rückfahrt vorgeschlagen, die er modifizieren und danach die Verbindungen zur Rückfahrt anfordern kann. Diese Liste bewertet der Benutzer analog zur Hinfahrt und kann die Gesamtfahrkarte anschließend auch buchen.

Dieser Ablauf korrespondiert wahrscheinlich nicht mit den Erwartungen der meisten Besucher, die als wesentliche Information nach Angabe der Reisedaten (Wann von wo nach wo?) direkt Information über die Reiseverbindungen erwarten; dass man vor Erhalt der Information über mögliche Rückfahrtverbindungen zunächst *genau eine* Verbindung für die Hinfahrt aussuchen muss ist ein Stolperstein für viele Besucher.

3 Modulare Aufgaben-Objekt-Modellierung

In diesem Kapitel stellen wir zunächst klassische Aufgabenmodelle als Basis unserer anschließend diskutierten Modellierungstechnik vor. Wir sprechen von *Modularer Aufgaben-Objekt-Modellierung*, da die Beschreibung der beeinflussten Objekte und die modulare Gliederung der Modelle die wesentlichen Erweiterungen darstellen.

3.1 Aufgabenmodellierung als Grundlage

In der Aufgabenmodellierung sind die *Concurrent Task Trees (CCT)* (Mori et al. 2002) und die zugehörige Entwicklungsumgebung CTTE weitgehend als Standard etabliert. CTT bietet hierarchische Aufgabenmodellierung und unterstützt dabei auch verschiedene Rollen (Typen von Akteuren). Aufgabenobjekte im obigen Sinne können in Dialogboxen spezifiziert werden, werden innerhalb des Modells allerdings dann nicht weiter verwendet. CTT bietet die Möglichkeit zu spezifizieren, dass neben einer rein zeitlichen Abhängigkeit auch ein Informationstransfer stattfindet, dies ist aber strukturell auf benachbarte Aufgaben beschränkt und die weitergegebene Information ist nicht näher beschreibbar. Zudem unterstützt CTT nicht das Setzen und Abfragen zusätzlicher Bedingungen. Auch die bekannte Benennung von Aufgabenmitteln und Aufgabengegenständen stellt zwar den Bezug zwischen Aufgaben und veränderter Umgebung her, bleibt aber zu unpräzise in der Beschreibung der Wirkung. In eigenen Arbeiten (*Tombola* (Uhr 2003), *Tamoa* (Zeiger & Mistrzyk 2005)) wurden Objekte

und Bedingungen explizit in die Modellierung eingebracht, allerdings bestehen für den angestrebten Einsatzzweck verschiedene Defizite bezüglich eines Rollenmodells oder der Modularisierbarkeit. Auch verschiedene Webmodellierungsansätze haben auf die wachsende Bedeutung von Abläufen bei der Beschreibung von Webauftritten reagiert. Besonders weit getrieben wurde dies innerhalb von UWE (Koch & Kraus 2002), indem ein explizites Prozessmodell integriert wurde (Knapp et al. 2004). Auch das ursprünglich stark datenbank- und damit inhaltsorientierte WebML (Ceri 2002) enthält Sprachmittel zum Beschreiben von Prozessen.

Als Grundlage für die Modellierung von Bedienabläufen im Web stützen wir uns auf drei Grundkonzepte klassischer Aufgabenmodelle ab: *Hierarchisierung* der Aufgaben, Festlegung von *temporalen Relationen* zwischen Aufgaben und die Zuordnung von *Rollen*. Rollen repräsentieren Akteure, die das Aufgabenmodell „ausführen" und dabei verschiedene Verantwortlichkeiten (Rechte und Pflichten) haben. Im Falle des Bahnverbindungs-Dialogs eines Benutzers mit einer Website existieren zwei Rollen, „Besucher" und „System" – in anderen Anwendungsfällen können die Rollen der Bediener erheblich differenzierter ausgeprägt sein. In den Abbildungen der Modelle (s.u.) stellen wir Aufgaben der Rolle „System" in grau mit weißer Schrift, Aufgaben des menschlichen Bedieners in der Rolle „Besucher" in hellgrau und Aufgaben, die von beiden Rollen gemeinsam durchgeführt werden, in weiß dar.

3.2 Das Modellkonzept

Unsere Modellerweiterungen gegenüber klassischer Aufgabenmodellierung betreffen vor allem die Objekte, die von der Aufgabendurchführung betroffen sind, die so genannten Aufgabenobjekte. Diese Objekte und die darauf stattfindenden Manipulationen werden in das Konzept eingebunden; anschließend wird die Modularisierung kurz behandelt.

Aufgabenobjekte

Aufgabenmodellierung stellt eine Beschreibungssprache für Aufgabenerledigung dar, die auf hohem Abstraktionsniveau angesiedelt ist. Daher soll auch der Objektbegriff entsprechend abstrakt eingesetzt werden – wir sprechen hier von so genannten *unscharfen Objekten*. Das soll deutlich machen, dass Objekte einerseits als klar von ihrer Umgebung abgegrenzt erkennbare „Dinge" sind, andererseits aber Details der Beschreibung oder die innere Struktur nicht angegeben werden. Es findet keine Typisierung oder anderweitige Charakterisierung oder Verfeinerung statt. Beispielweise ist „Abfahrtszeit" ein Objekt des Bahnbeispiels. Es wird benannt, aber nicht in Form oder Syntax näher spezifiziert. Ein anderes intern wesentlich komplexeres Objekt des Beispiels ist „Zugverbindungen für die Rückfahrt". Zwar wird nichts über die innere Struktur der Objekte gesagt, aber es wird ermöglicht festzulegen, wie Objekte den Ablauf des Aufgabenmodells beeinflussen.

Objekteinbindung: Die Rucksackmetapher

Aufgabenmodelle beschreiben die Ausführung von Aufgaben durch menschliche Bediener eines Systems. In der Modellierung abstrahieren wir von dieser Aufteilung derart, dass wir in beiden Fällen von Akteuren sprechen, die die Aufgaben ausführen, jeweils charakterisiert

durch verschiedene Rollen, nämlich „System“ und „Benutzer“. Während die Akteure das Modell durchlaufen, manipulieren sie die unscharfen Objekte (s. Objektoperationen).

Der menschliche Akteur empfindet die Bedienfolge an einer Website als zusammenhängendes Geschehen – technisch spricht man auch vom Begriff einer *Sitzung*. Im Laufe einer Sitzung tauschen das System und der Bediener Informationen aus, etwa spezifiziert der Benutzer der Bahn-Webseite einen Abfahrtsort und eine Abfahrtszeit, diese Daten werden vom System aufgenommen, für Berechnungen genutzt und ihm auch wieder gezeigt. Für den Benutzer baut sich eine Modellvorstellung von „gemeinsamem Wissen“ zwischen ihm und dem System auf, das im Laufe einer Sitzung zu- und abnimmt. Als Metapher für den Ort, in dem das Sitzungswissen aufgehoben wird, verwenden wir die Metapher eines *Rucksacks*, den der Benutzer von einem Arbeitsschritt zum nächsten trägt. Die Interaktion mit der Webseite beginnt mit einem fast leeren Rucksack – das System „weiß“ praktisch nichts über den Besucher. Im Laufe der Zeit kommen dann „Dinge“, Informationen, d.h. Objekte, hinzu, werden geändert, überprüft und auch wieder entnommen. Allerdings wird der Rucksack initial nicht als völlig leer angenommen: Er enthält für jeden Akteur das Objekt „Rolle“, in dem angegeben wird, welche Rolle der Akteur aktuell einnimmt. Ein anonymer Besucher der Website, der sich in keiner Weise besonders ausgezeichnet hat (etwa durch Registrierung oder Existenz eines *Cookies* auf seiner Festplatte) wird durch die Rolle „Besucher“ charakterisiert. Durch die nachfolgend beschriebenen Objektmanipulationen können alle Objekte des Rucksacks – auch die Rollenzuordnung – verändert werden.

Objektmanipulationen

Im Folgenden werden die in dem Modellkonzept zur Verfügung gestellten Objektmanipulationen diskutiert. Aus pragmatischen Gründen wurde hier die übliche Darstellung von Aufgaben in Aufgabenmodellen als mit Linien verbundenen Rechtecken (s. etwa Abb. 11) um grafische und einfache textuelle Elemente ergänzt. Die Festlegung der endgültigen Darstellung wird Gegenstand einer nachfolgenden Tool-Entwicklung sein; ein Abgleich mit existierenden Modellen – zum Beispiel UML – ist vorgesehen.

Abbildung 2

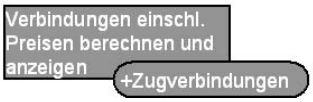

Abbildung 3

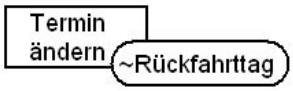

Abbildung 4

Sowohl der menschliche Akteur als auch das System stellen Objekte für die Aufgabenerledigung zur Verfügung. Dies wird in einem klassischen Aufgabenmodell in den Beschreibungen der Aufgaben ausgedrückt, etwa wenn der Besucher die Aufgabe „Zielort festlegen“ durchführt: Der Besucher stellt dem Prozess das Objekt „Zielort“ zur Verfügung. In unserer Modellierung wird dies ausgedrückt durch die Klausel „+Zielort“, notiert in einem Oval auf dem Rand der Aufgabe, die diese Wirkung zeigt (Abb. 2). Analog stellt das System später im Prozess etwa das Objekt „Zugverbindungen für die Hinfahrt“ bereit, wenn die Aufgabe „Verbindungen einschl. Preisen berechnen und anzeigen“ durchgeführt wird. Durch Zeichnen der Objektmanipulation über den rechten Rand des Aufgabekastens (Abb. 3) wird aus-

gedrückt, dass *nach* Erledigung der Aufgabe dieses Objekt zur Verfügung steht. Analog zum Bereitstellen können Objekte auch durch eine Aufgabe verändert werden, markiert durch eine Tilde (~). Wird zum Beispiel der Tag der Rückfahrt in der Aufgabe „Termin ändern" modifiziert, dann kann dies durch „~Rückfahrttag" notiert werden (Abb. 4). In einer spezielleren Form kann man die Werteänderung eines Objektes ggf. auch direkt notieren, falls dies benötigt wird. Diese Art der Modifikation wird als „Objekt = neuer Wert" notiert, etwa wie in „Fahrradmitnahme = Nein", wenn der Benutzer kein Fahrrad auf die Bahnfahrt mitnehmen möchte. Schließlich können Objekte auch wieder der Sitzung entzogen werden – notiert durch ein Minuszeichen (-).

Die Rolle, die ein Objekt bereitstellt, ist der „Besitzer" dieses Objekts und hat dann auch alle Rechte an diesem Objekt: Sie kann das Objekt bereitstellen, modifizieren und wieder zurückziehen – im Normalfall können alle anderen Rollen das Objekt nur lesen. Das Konzept erlaubt aber auch die Spezifikation beliebiger anderer Fälle von Zugriffsrechten. Dadurch ist es schon auf der abstrakten Ebene der hier beschriebenen Modellierung möglich, sehr präzise die Möglichkeiten der Interaktion für verschiedene Benutzergruppen anzugeben. Spezielle Regeln gelten für das Objekt „Rolle" selbst: Dieses darf von niemandem bereitgestellt oder zurückgezogen werden, verändern darf es nur das System.

Abbildung 5

Abbildung 6

Objektbedingungen

Zur Beeinflussung des Ablaufs des Aufgabenmodells können die im Rucksack vorhandenen Objekte auf das Erfülltsein von Bedingungen geprüft werden. Meist werden Bedingungen als notwendig für das *Starten* einer Aufgabe formuliert, dann am linken Rand einer Aufgabe, sind aber auch als Abschlussbedingungen erlaubt. In der einfachsten Form wird die Existenz eines Objektes im Rucksack erfragt, etwa „Hat der Benutzer einen Rückfahrttag festgelegt?" durch die Abfrage der Existenz des Objektes „Rückfahrttag" bei der Aufgabe „Vorher gewählte Daten anzeigen" (Abb. 5). Auch die Nicht-Existenz, markiert durch ein vorangestelltes Nicht-Zeichen (¬), kann erfragt werden und Vorbedingung für eine Aufgabe sein (Abb. 6).

Abbildung 7

Abbildung 8

Darüber hinaus kann auch geprüft werden, ob ein Objekt eine bestimmte Bedingung erfüllt. Der Test selber wird nur informell spezifiziert, sollte aber so festgelegt sein, dass er für einen menschlichen Leser stets ein wohldefiniertes Ergebnis liefert. So kann etwa die Bedingung „Hat sich der Benutzer für genau eine der angebotenen Hinfahrten entschieden?“ dadurch formuliert werden, dass das Objekt „AuswahlHinfahrten“ eine Menge mit genau einem Element ist: „AuswahlHinfahrten einelementig?“ (s. Abb. 7). Dass die Webseite der Bahn fordert, dass dies der Fall sein muss, ehe die Rückfahrt betrachtet werden kann, lässt sich daher als entsprechende Vorbedingung notieren. Auch die Wertangaben für neue Objekte lassen sich für Bedingungen nutzen, etwa mit einer Bedingung wie „Fahrradmitnahme == nein?“. Es können auch mehrere Bedingungen notiert und nacheinander geprüft werden, wie für die Aufgabe „Hinfahrt festlegen“ gezeigt (s. Abb. 8).

Modularisierung

Keine Website existiert losgelöst von ihrer Umgebung: Wer eine Bahnfahrt bucht, braucht oft auch ein Hotel, einen Mietwagen, einen Stadtplan oder ein Theaterprogramm. Die hochgradige Verknüpfbarkeit von Inhalten im Web ist gerade seine Stärke. Daher treten in Webabläufen aus Benutzersicht vielfache Wechsel von Websites auf. Natürlich müssen Abläufe zunächst innerhalb einer Website modelliert werden können – aber Übergänge zu anderen Websites gehören auch dazu. Ein solcher Übergang bedingt den Beginn einer neuen „leeren“ Sitzung, da die Zielsite i.A. keine Daten über den neuen Besucher hat. In der Sprechweise unserer Modellierung legt der Benutzer seinen Rucksack vor Verlassen der Ursprungssitzung ab und erhält einen neuen Rucksack, in dem lediglich die Rollenangabe „Besucher“ enthalten ist. Nach Rückkehr von der Zielsite hat er Erkenntnisse gewonnen (etwa welches Hotel er gebucht hat) und kann dies in die Ursprungssitzung einfließen lassen. Er nimmt also den Rucksack wieder auf und stellt darin zusätzlich das Objekt „Hotel“ bereit. Wird der Übergang zu einer anderen Website aber auch vom System unterstützt, können durchaus direkt Daten der einen Sitzung in die neue Sitzung fließen. Dies kann formuliert werden, indem der Übergang zum Zielmodell mit Parametern (s. Abb. 9) versehen wird. Diese Angabe führt im Beispiel dazu, dass das Objekt „Zielort“ in den Rucksack für die neue Sitzung und nach Rückkehr das Objekt „Hotel“ aus der neuen Sitzung übernommen wird (s. Abb. 10).

Abbildung 9

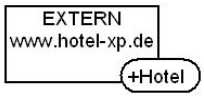

Abbildung 10

3.3 Das Beispiel

Betrachtet man den Auskunftsvorgang aus Kapitel 2, wird deutlich, dass auf der obersten Ebene der Aufgabenhierarchie sechs Aufgaben in sequentieller Reihenfolge durchzuführen sind (s. Abb. 11). „Hinreisedaten ermitteln“ entspricht dem Ausfüllen des kleinen Rechtecks links oben auf der Homepage der Bahn. Man macht Eingaben zu den Hinreisedaten – dies kann in irgendeiner Reihenfolge erfolgen – und stellt die ersten Objekte für die Sitzung bereit (s. Abb. 12). Auf der Folgeseite kann der Benutzer Tag- und Zeitangaben für die Rückfahrt machen – muss dies aber nicht. Daher ist die entsprechende Aufgabe („Rückreisedaten ermitteln“) durch ein kleines Rechteck mit dem Text „opt“ links oben als *optional* gekennzeichnet. Wenn die Aufgabe ausgeführt wird, liefert diese Aufgabe die Angaben R-Tag und R-Zeit. Anschließend muss der Benutzer Angaben zur Preisberechnung machen (s. Abb. 13), die die Zahl der Reisenden, die vorhandenen Ermäßigungen und die Wagenklasse betreffen. Diese Aufgabe wird nicht als optionale Aufgabe modelliert, sondern als Aufgabe mit mehreren optionalen Unteraufgaben. Der Hintergrund ist, dass mindestens der Neuling auf dieser Seite gezwungen wird, diese Anteile durchzulesen und (eventuell) nichts zu verändern, wenn er nur eine Verbindungsauskunft braucht. Diese Entscheidung zu treffen („Das interessiert mich jetzt nicht und es schadet nicht, es erst mal zu ignorieren“) ist kognitiver Aufwand! Übrigens muss auch der erfahrene Benutzer auf der Hut sein – mindestens muss er immer wieder prüfen, ob die Seite sich nicht geändert hat, und er diese Angaben wirklich einfach überspringen kann. Ähnlich ist die Aufgabe „Angaben zur Verbindung machen“ strukturiert.

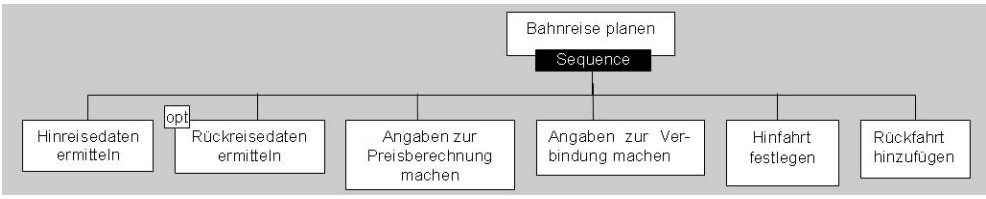

Abbildung 11: Oberste Ebene der Aufgabenhierarchie

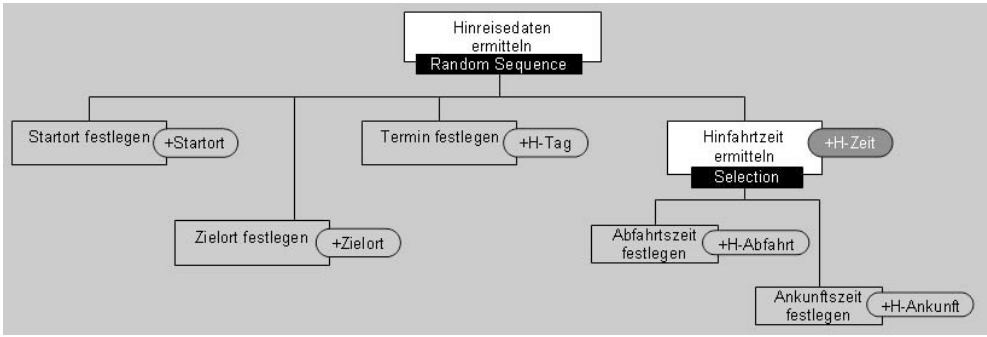

Abbildung 12: Aufgabenobjektmodell für „Hinreisedaten ermitteln“

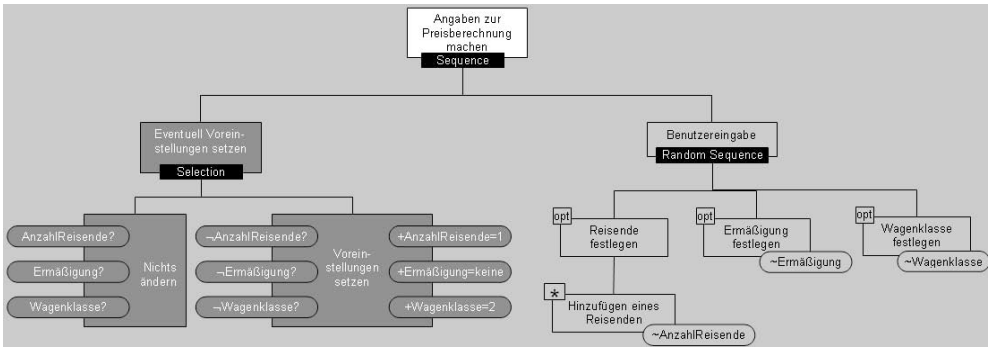

Abbildung 13: Aufgabenobjektmodell für „Angaben zur Preisberechnung machen“

Die Aufgabe „Hinfahrt festlegen“ zerlegt sich wie in Abbildung 14 gezeigt[1]. Wenn der Benutzer die Iteration (Aufgabe „Verbindung auswählen“, gekennzeichnet durch den kleinen Stern links oben Abb. 14) verlässt, enthält das Objekt H-Auswahl die von ihm ausgewählten Verbindungen für die Hinfahrt; nur wenn diese Menge einelementig ist, kann er in die Folgeaufgabe „Zur Rückfahrt wechseln“ einsteigen, ist aber ansonsten in der Iteration „gefangen“.

[1] Die Option frühere oder spätere Verbindungen anzeigen zu lassen wurde aus Platzgründen weggelassen, ist aber auch entsprechend modellierbar.

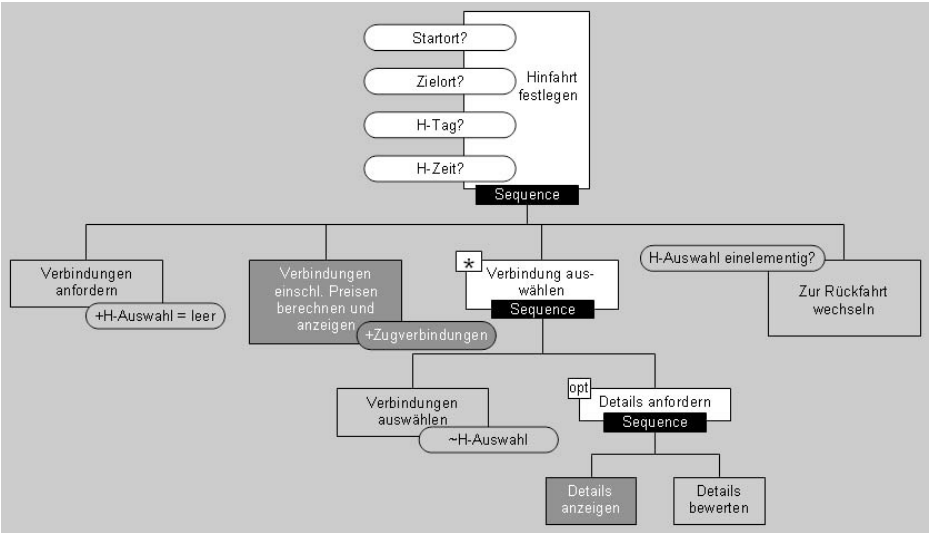

Abbildung 14: Aufgabenobjektmodell für „Hinfahrt festlegen“

Analog können nun auch die Rückfahrt und der folgende Buchungsprozess modelliert werden, was – genau wie ein Beispiel für den Einsatz des Modularitätsaspekts – hier aus Platzgründen nicht ausführlich dargestellt werden kann. Insgesamt zeigt die Betrachtung dieses Beispiels aber, dass es möglich ist, mit der Aufgaben-Objekt-Modellierung ein detailliertes Abbild der Abläufe auf einer Website aus Benutzersicht zu beschreiben. Dies geschieht ohne jegliche Festlegungen zum Aussehen der Website.

4 Einsatz der Aufgaben-Objekt-Modellierung

Die beschriebene Modellierungstechnik bietet das Potenzial, bei der *Evaluation* existierender Abläufe im Web – wie hier für die Bahn-Website geschehen – zum Einsatz zu kommen. Sie bietet Abstraktion von jeglichen Präsentationseigenschaften und konzentriert sich voll auf die Entscheidungs- und Denkprozesse des Menschen im Zusammenspiel mit den Angeboten des Systems. Anhand dieser Darstellung lässt sich ablesen, welche Entscheidungen dem Benutzer in welcher Reihenfolge abverlangt werden. Auch können Usability-Regeln, wie etwa die Nielsen-Heuristiken (Nielsen 1993), insoweit überprüft werden, soweit sie sich auf Ablaufstrukturen (z.B. „Natürlicher Dialog“ oder „Minimieren von Gedächtnisbelastung“) unabhängig von der symbolischen Darstellung beziehen.

Es ist nahe liegend, dass das Konzept auch als Hilfsmittel beim *Entwurf* dienen kann. Dadurch, dass man bei der Entwicklung des Modells deutlich machen kann, welche Informationen von welchem Beteiligten verlangt werden, entsteht ein klares Bild des späteren Ablaufs ohne Gestaltungsentscheidungen vorweg zu nehmen. Insbesondere für den Entwurf neuarti-

ger, bislang nicht existenter Abläufe bietet es sich an, diese Entwurfsebene explizit in den Prozess einzuschalten.

Im Rahmen des WISE-Projektes (WISE 2006) werden zurzeit Editorwerkzeuge für eine Variante der Aufgaben-Objekt-Modellierung implementiert und in eine Web-Engineering-Umgebung integriert. Auch ein Prototyping-Werkzeug ist in Arbeit.

Literaturverzeichnis

Ceri, S. (2002): Designing Data-Intensive Web Applications. Morgan-Kaufman Publishers, December 2002.

Knapp, A; Koch, N.; Zhang, G.; Hassler, H.-M. (2004): Modeling Business Processes in Web Applications with ArgoUWE. 7th International Conference on the Unified Modeling Language (UML2004), LNCS 3273, 69-83, Springer Verlag, October 2004.

Koch, N.; Kraus, A. (2002): The expressive Power of UML-based Web Engineering. Second International Workshop on Web-oriented Software Technology (IWWOST02), D. Schwabe, O. Pastor, G. Rossi, and L. Olsina, editors, CYTED, 105-119, June 2002.

Mori G.; Paternò F.; Santoro C. (2002): CTTE: Support for Developing and Analyzing Task Models for Interactive System Design. IEEE Transactions on Software Engineering, August 2002, pp.797-813.

Nielsen, J. (1993): Usability Engineering. Academic Press International, 1993.

Uhr, H. (2003): TOMBOLA: Simulation and User-Specific Presentation of Executable Task Models, Paper. HCI International 2003, Crete, Greece.

van Welie, M.; van der Veer, G. C.; Eliëns, A. (1998): Euterpe – Tool support for analyzing cooperative environments. Ninth European Conference on Cognitive Ergonomics, pp. 25-30, 1998, Limerick, Ireland.

WISE (2006), BMBF-Projekt Web Engineering and System Engineering. http://www.wise-projekt.de/de/.

Zeiger, B.; Mistrzyk, T. (2005): Tamoa: Task Model Analyser – A Java-based Framework for the Analysis of Safety-critical Computer Systems to Reveal Safety-critical Sections in the System's Design Phase. HCI International 2005, Las Vegas, USA.

Kontaktinformationen

Universität Paderborn, Institut für Informatik, Gerd Szwillus
Fürstenallee 11, 33102 Paderborn szwillus@uni-paderborn.de Tel.: +49 5251/60-6624
Fax.: +49 5251/60-6619