H. Wandke, S. Kain & D. Struve (Hrsg.): Mensch & Computer 2009: Grenzenlos frei!?
München: Oldenbourg Verlag, 2009, S. 23-32

Benutzungsorientierte Entwicklung barrierefreier Benutzungsschnittstellen

Sabina Jeschke[1], Olivier Pfeiffer[2] & Helmut Vieritz[3]

Zentrum für Lern- und Wissensmanagement[1]; RWTH Aachen, Multimediazentrum für Lehre und Forschung, TU Berlin[2]; Rechenzentrum Universität Stuttgart[3]

Zusammenfassung

Konzeption und Design von barrierefreien[1] Benutzungsschnittstellen stellen komplexe Herausforderungen an Softwarearchitekten und Entwickler. Neben individuellen Beeinträchtigungen eines Menschen (z.B. Behinderungen des Nutzers) können Störungen durch Umgebungseinflüsse (z.B. Lärm, helles Licht) sowie Einschränkungen, die sich durch die Technologie der Benutzungsschnittstelle (z.B. kleiner Bildschirm, fehlende Maus oder Tastatur) ergeben, die Benutzung erschweren. Der hier vorgestellte Ansatz untersucht und demonstriert, dass modell- und benutzungsorientierten Vorgehensweisen zur einfacheren Entwicklung zugänglicher Benutzungsschnittstellen geeignet sind. Ausgangspunkt ist dabei die Bedarfsanalyse aus Sicht des Benutzers und die Modellierung der Arbeitsprozesse, die die Applikation unterstützen soll. Mit Hilfe der Modellierung der Mensch-Maschine Kommunikation können multimodale Bedienkonzepte in den Schnittstellenentwurf integriert werden.

1 Einleitung

Gesetzliche Vorgaben und technische Richtlinien für die barrierefreie Gestaltung von Webangeboten existieren seit geraumer Zeit und konnten dennoch einer allgemeinen Barrierefreiheit noch nicht zum Durchbruch verhelfen. Die Entwicklung zugänglicher Internet-Anwendungen gilt als aufwändig. Zwar existieren die Empfehlungen des World Wide Web Konsortiums (W3C) – u.a. *Web Content Accessibility Guidelines* (WCAG 1999) – bereits seit 1999 sowie zahlreiche weitere Empfehlungen (z.B. IN EN ISO 9241-171 2008) und populäre Auszeichnungen wie der Biene-Award demonstrieren immer wieder, dass barrierefreies Webdesign ohne Abstriche möglich ist. Dennoch sind zugängliche Webseiten in der

---

[1] *Barrierefrei*, *zugänglich* bzw. *accessible* sind in diesem Beitrag synonym zu verstehen und bezeichnen die freie Zugänglichkeit von Informationen für jeden Benutzer. Insbesondere bezieht sich das auf den freien Zugang zu Informationen für Menschen mit Behinderung in computerbasierten Anwendungen.

Minderzahl und die derzeitige Situation kann nicht allein mit dem Unwillen der Verantwortlichen erklärt werden. Die Erleichterung der Entwicklung barrierefreier Informationstechnologien ist deshalb – neben der Forschung an geeigneten assistiven Technologien[2] – ein wichtiges Forschungsfeld.

Die Herausforderung liegt in der Komplexität der Anforderungen barrierefreier Webangebote. Hinzu kommt, dass diese Anforderungen bereits in den frühen Phasen der Anwendungsentwicklung beachtet werden müssen, denn obwohl Barrierefreiheit eine Qualität der Benutzungsschnittstelle ist, stellt sie Anforderungen an das Gesamtkonzept der Applikation. Ohne detaillierte Fachkenntnisse ist dies nicht zu bewältigen.

Ein Beispiel: Für die barrierefreie Integration von Grafiken und Abbildungen in die Anwendung müssen unter anderem textbasierte Informationen über Zweck und Funktion der Bilder angeboten werden, die assistive Technologien wie bspw. Screenreader nutzen können. Wird diese Anforderung nicht bereits im Datenmodell komplexerer Anwendungen beachtet und fehlen die erforderlichen Datenobjekte, dann kann sie im Design der Benutzungsschnittstelle auch nicht umgesetzt werden.

Die Barrierefreiheit der Oberfläche ist dementsprechend als Anforderung an das Gesamtkonzept der Applikation zu sehen und von Beginn an zu beachten. Die übliche Konsequenz ist, dass das notwendige Expertenwissen bereits früh im Entwicklungsprozess involviert wird. Eine Lösung, die nur in größeren Projekten umgesetzt werden kann.

Hier wird ein alternativer Lösungsansatz verfolgt. Dazu wird erstens der Konzeptionsprozess der Anwendung mit der Benutzungsschnittstelle selbst begonnen, und es wird zweitens eine Vorgehensweise für die Entwicklung gewählt, die das Gesamtkonzept der Applikation in den Vordergrund stellt. Für die erste Idee hat sich bereits der Begriff *benutzer-* bzw. *benutzungsorientierte Entwicklung*[3] (DIN EN ISO 13407 1999) etabliert. Eine geeignete Vorgehensweise für die zweite Idee ist die *modellzentrierte* bzw. *-getriebene Entwicklung*. Die Kombination aus beidem – die benutzer- bzw. benutzungszentrierte Modellierung (z.B. Reuther 2003) – ist ein geeignetes Vorgehen für die gestellten Ziele.

Herkömmlicherweise stehen am Beginn der Applikationsentwicklung Funktionalität und Architektur der Applikation im Fokus und die Benutzungsschnittstelle wird dann an diesen Funktionalitäten ausgerichtet. Ohne besondere Beachtung der Anforderungen der Benutzer entstehen so schnell unübersichtliche Benutzungsschnittstellen. Dieses an den Sichtweisen der Entwickler und Softwarearchitekten orientierte Vorgehen wird mit zunehmenden Applikationsfunktionalitäten problematischer. Heutzutage umfasst Konzeption und Design der Applikationsoberfläche oft bereits den überwiegenden Anteil an Arbeitszeit in der Software-

---

[2] Der Begriff *Assistive Technology* bezeichnet im Englischen Technologien, die Menschen mit Behinderungen in der alltäglichen Lebensführung unterstützen. Im Bereich der IT-Technologien zählen dazu unter anderem Screenreader, Braillezeilen, Bildschirmlupen, Speechsynthesizer, Spracheingabelösungen, alternative Keyboards und Zeigegeräte (WCAG 1999).

[3] Bei der benutzer- bzw. benutzungsorientierten Methode ist der Entwicklungsprozess auf die Anforderungen der Benutzer hin ausgerichtet, mit dem Ziel, das Design benutzungsfreundlicher Produkte zu unterstützen.

entwicklung und der Schwerpunkt der Projektarbeit verschiebt sich von der Entwicklung weg von den Funktionalitäten hin zur Gestaltung ihrer Bedienung. Die logische Konsequenz ist die Forderung, den gesamten Entwicklungsprozess bei den Anforderungen des Benutzers zu beginnen und daraus geeignete Bedienkonzepte abzuleiten.

Die grundlegende Idee der modellzentrierten Entwicklung – dass zuerst Modelle der Anwendung entwickelt werden, aus denen danach durch einen Modellcompiler halb- bzw. vollautomatisch der Anwendungscode erzeugt wird – verschiebt den Schwerpunkt der Arbeit des Entwicklers. Wird durch einen benutzungsorientierten Ansatz die Mensch-Maschine-Schnittstelle in diesen Prozess integriert und an die erste Stelle gesetzt, ergibt sich die Möglichkeit, die Gestaltung zugänglicher Anwendungen zu vereinfachen. Ausgangspunkt des Designs sind dann die Interessen des Benutzers und die Analyse der Arbeitsprozesse, die die zukünftige Applikation unterstützen soll. Die Modellierung erfolgt unabhängig von den technischen Gegebenheiten des verwendeten Systems und den zu seiner Bedienung notwendigen physischen Operationen. Werden neben den allgemeinen Anforderungen der Benutzer auch spezifische beachtet, die sich aus vorliegenden sensorischen, motorischen oder auch technischen Einschränkungen heraus ergeben, können schwerwiegende Designfehler bzgl. der Barrierefreiheit und Zugänglichkeit von Beginn an vermieden werden. Voraussetzung dafür ist die Integration der Anforderungen barrierefreier Anwendungen in den Modellierungsprozess. Entsprechende Ansätze in der Forschung sind bisher selten, zeigen aber das Potenzial dieser Vorgehensweise (z.B. Goble et. al. 2005). Durch die entsprechende Werkzeugunterstützung werden Softwarearchitekten und Entwickler entlastet und die Barrierefreiheit der Benutzungsschnittstelle ergibt sich als Qualität eines integrierten Entwicklungsprozesses.

Der Gewinn dieses Vorgehensmodells sind zum einen die allgemeinen Vorteile der modellzentrierten Entwicklung – Trennung von Funktionalität und Technik, Wiederverwendung etc. – und zum anderen eine adäquate Herangehensweise an die beschriebene Problemstellung und die sich daraus ergebende Vereinfachung der Entwicklung barrierefreier Benutzungsschnittstellen.

2 Das Vorgehensmodell

Ausgangspunkt des benutzungszentrierten Designs sind nach Normans *Activity-Centered Design* (Norman 1989; Norman 1998) die Ziele und Aktivitäten des Benutzers. In Form von Anwendungsfällen lassen sich dann in grober Perspektive typische Nutzungssituationen beschreiben. Als Notationssprache werden Anwendungsfalldiagramme der *Unifying Modeling Language* in ihrer zweiten Version (UML 2) genutzt. Für die grafische Notation ist derzeit UML 2 das Mittel der Wahl. Da eine rein grafische Notation visuell nicht zugänglich wäre, ist eine korrespondierende textbasierte Notation notwendig, die von UML in Form der *Human-Usable Textual Notation* (HUTN 2004) unterstützt wird. Die Vorteile von UML sind die breite Verwendung in der Softwareentwicklung, die flexible Anpassungsmöglichkeiten (z.B. Stereotypen oder Profile) und die textbasierte Notation.

Beispielhaft wird hier das Vorgehen an Hand der Modellierung der Benutzungsschnittstelle einer einfachen eLearning-Plattform demonstriert. Typischerweise haben die Benutzer der eLearning-Plattform folgende Ziele: Der Student will sich zu einer bestimmten Vorlesung informieren sowie sie besuchen und der Dozent will eine bestimmte Vorlesung erstellen sowie verwalten. Abbildung 1 zeigt das Anwendungsfalldiagramm mit fünf Anwendungsfällen.

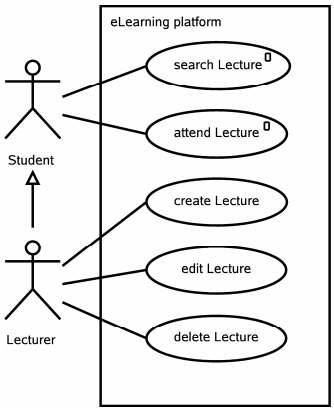

Abbildung 1: Anwendungsfalldiagramm für die eLearning-Plattform

2.1 Aufgabenanalyse und Mensch-Maschine Dialog

Die Analyse und Modellierung der Arbeitsprozesse[4] (Aufgabenanalyse) dient der Detaillierung der Anwendungsfälle. Für die Analyse besonders geeignet sind Strukturlegepläne (Reuther 2003) – ergänzt um strukturierte Interviews und teilnehmende Beobachtung. Für die Notation eignen sich die Aktivitätsdiagramme der UML 2. Die Aufgabenanalyse untersucht und beschreibt die Arbeitsabläufe, die der Benutzer mit Hilfe der Anwendung umsetzen will. Die Arbeitsprozesse lassen sich auf diversen hierarchischen Niveaus in Einzelschritte zerlegen. Die Namensgebung für die Teilprozesse variiert und orientiert sich hier am Modell der UML 2: Use Case (Task), Activity und Action (siehe Abbildung 2). Tasks lassen sich in Aktivitäten und weiterhin in Aktionen zerlegen. Die unterste Ebene der physischen Operationen wird dabei abstrahiert und es wird in der Beschreibung eine weitgehende Unabhängigkeit von den Ein- und Ausgabegeräten erreicht.

[4] Unter *Arbeitsprozessen* oder *Workflow* werden hier die Tätigkeiten des Benutzers zur Bedienung der Applikation verstanden.

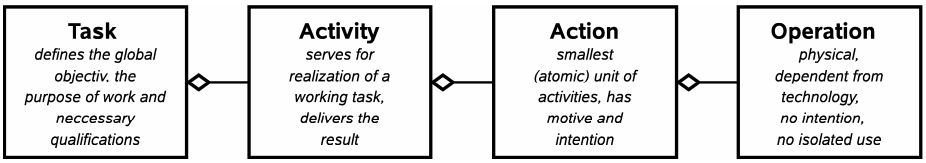

Abbildung 2: Hierarchische Zerlegung von Arbeitsprozessen in UML-Notation

Die konkreten Arbeitsabläufe variieren von Benutzer zu Benutzer und der Mensch-Maschine-Dialog sollte idealerweise die unterschiedlichen Abläufe unterstützen. In die Analyse und Modellierung muss mit einfließen, wie sich die Arbeitsabläufe bei Vorliegen bestimmter Behinderungen des Benutzers gestalten und ob dadurch spezifische Anforderungen entstehen. Da die relevanten Analyseebenen Task, Activity und Action unabhängig von den physischen Operationen des Benutzers beschrieben werden, fallen dort die behinderungsspezifischen Differenzen gering aus. Spezielle Untersuchungen zu dieser Problematik stehen jedoch ganz allgemein noch aus.

Im Bereich des Useware Engineerings[5] fand der von Reuther vorgestellte useML-Ansatz (*Use*ware *M*arkup *L*anguage, Reuther 2003)in den letzten Jahren große Resonanz, da er ein Problem der Aufgabenanalyse löste: ihre Komplexität. Durch die Verwendung von fünf Basiselementen in useML – den sogenannten elementaren Benutzungsobjekten – lassen sich Analyse und Modellierung bedeutend vereinfachen (siehe Abbildung 3). Ausgehend von der XML-basierten Notation in useML verwendet der hier vorgestellte Ansatz UML 2 als Notationssprache, da es komplexere Zusammenhänge anschaulicher darstellen kann und derzeit die Lingua franca in der Softwaremodellierung ist.

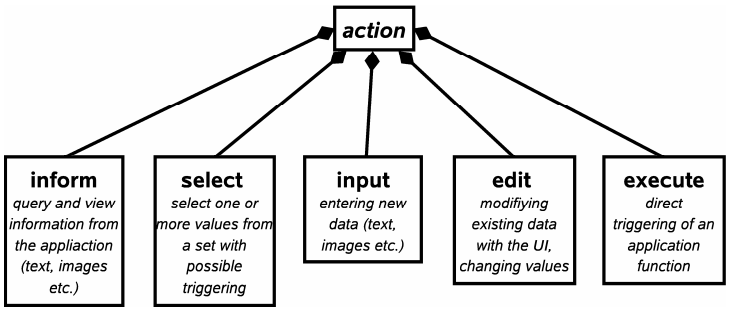

Abbildung 3: Elementare Benutzungsobjekte in useML

Dazu zählen die Elemente informieren (inform), auswählen (select), eingeben (input), editieren (edit) und auslösen (execute). Die durch die Einfachheit entstehenden Restriktionen sind

[5] *Useware* umfasst alle zur Bedienung eines technischen Systems notwendigen Hard- und Softwarekomponenten und wurde 1998 von GfA, VDI/VDE-GMA, GI und VDE-ITG als neuer Sammelbegriff vorgeschlagen.

auflösbar (Meixner & Görlich 2008). Abbildung 4 zeigt das UML-Aktivitätsdiagramm für den Anwendungsfall „create Lecture“ der eLearning-Plattform.

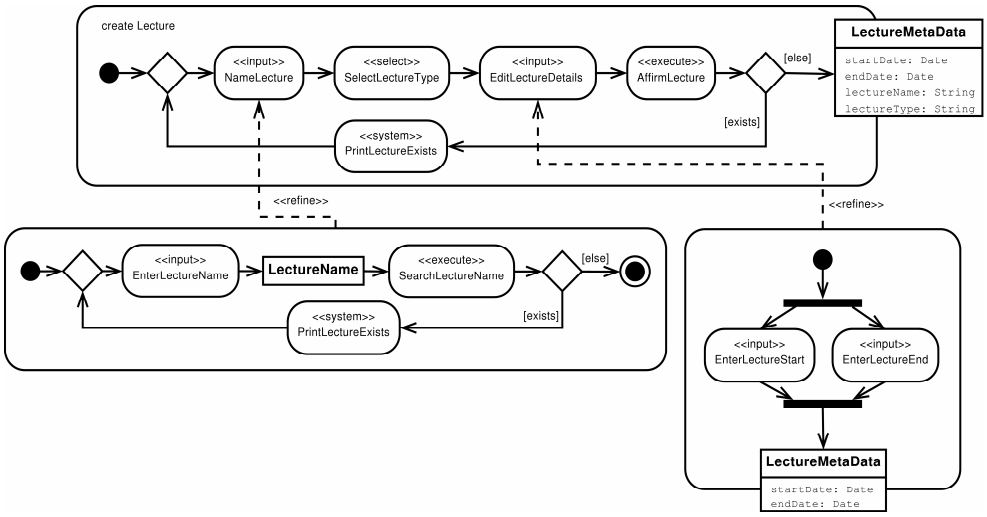

Abbildung 4: Aktivitätsdiagramm für den Anwendungsfall „create Lecture"

Der Dozent sucht, ob es eine bestimmte Vorlesung bereits gibt. Anderenfalls werden Metainformationen zur Vorlesung (Titel und Zeitraum) erzeugt. Die fünf elementaren Benutzungsobjekte werden als UML-Stereotypen markiert. So wird eine einfache Klassifizierung der Benutzeraktionen integriert, die die spätere Auswahl und Zuordnung von Interfacekomponenten erleichtert. Aktivitäten sind eine Zusammenfassung von Aktionen und werden über die *refine*-Assoziation in die Gesamtdarstellung eingebunden. Wie bereits erwähnt, wird davon ausgegangen, dass der Ablauf der Aktivitäten unabhängig von einer sensorischen oder motorischen Behinderung des Benutzers – hier der Dozent – ist. Unabhängig davon ob ein Bildschirm oder ein Screenreader bzw. Maus, Tastatur oder Spracherkennung genutzt werden, gestaltet sich der Arbeitsablauf auf die dargestellte Weise.

Nach Luyten (Luyten 2004) sind für benutzungszentrierte Ansätze das Task-, Dialog- und das Präsentationsmodell essentiell. Dialog- und Taskmodell werden vom useML-Ansatz nicht mehr unterstützt und werden hier ergänzend entwickelt. Das Dialogmodell beschreibt die Interaktion zwischen Benutzer und Benutzungsschnittstelle und vermittelt zwischen dem Task- und dem Präsentationsmodell. In Abbildung 4 sind die Reaktionen der Applikation mit dem Stereotyp *system* kenntlich gemacht. Detaillierter lassen sie sich mit Hilfe von Zustandsdiagrammen (state charts) in UML beschreiben und modellieren (Traetteberg 2002). Zur Unterstützung der Entwicklung barrierefreier Anwendungen sind multimodale Dialogmodelle besonders geeignet. Sie beschreiben die Mensch-Maschine Kommunikation unabhängig von den technischen und physischen Bedingungen der Benutzung und sind auf die Verwendung verschiedener Ein- und Ausgabegeräte hin ausgerichtet.

2.2 Das Abstrakte Präsentationsmodell

Der nächste Schritt in der Modellierung ist die Beschreibung des *Abstrakten Präsentationsmodells* (APM) der Benutzungsschnittstelle. Aktivitäten und Zustände werden auf abstrakte Interfaceobjekte abgebildet und deren logische Eigenschaften beschrieben. Für die Notation eignen sich Klassendiagramme der UML 2, die einzelne Interfaceobjekte als Klassen abbilden. Die für den Benutzer wichtigen Informationen über Bedeutung und Funktion der Interfacekomponenten werden als logische Eigenschaften semantisch vollständig dokumentiert. Beispielsweise werden die Metainformationen von Interfaceobjekten (Name, Titel etc.) beschrieben. Eventhandler beschreiben die Ereignisse, auf die die Benutzungsschnittstelle reagiert. Sie spielen eine wichtige Rolle bei der Gestaltung der Mensch-Maschine Kommunikation, z.B. bei der Verwendung von Javaskript in HTML-Dokumenten. Geräte- und technologieunabhängige Eventhandler unterstützen die Gestaltung einer barrierefreien Kommunikation.

Abbildung 5 zeigt das APM für die eLearning-Plattform. Die Struktur der Interfacekomponenten wird dargestellt.

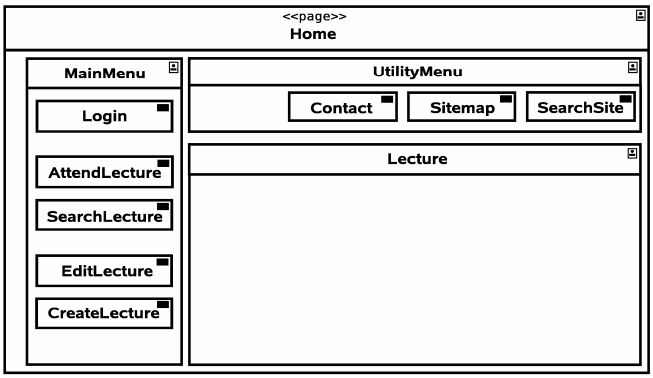

Abbildung 5: Abstraktes Präsentationsmodell für die eLearning-Plattform

Die Notation entspricht dem UWE-Ansatz (*UML-based Web Engineering*) (Knapp et al. 2007) aus dem Bereich des Web Engineerings. Der Stereotyp *page* bezeichnet den Gesamtcontainer einer Website. Die Methoden der Objekte lassen sich über die zugeordneten Eventhandler aufrufen. Die Anforderungen der Barrierefreiheit schließen dabei andere Anforderungen wie z.B. die der Usability nicht aus.

Abbildung 6 zeigt das detaillierte Klassendiagramm für die Interface-Komponente *SearchLecture*. Dabei handelt es sich um eine Suchfunktion für Vorlesungen mit Texteingabe. Der Stereotyp *form* weist die Komponente als Formularfeld aus.

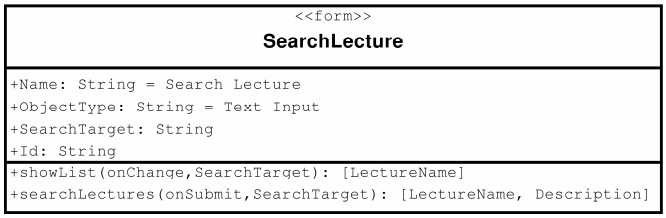

Abbildung 6: Interface-Komponente: Suchfeld für Vorlesung

Eine typische Barriere bei der Nutzung von Formularfeldern in Webseiten ist die fehlende Kennzeichnung des Feldes, die in HTML nicht direkt vorgesehen ist. Screenreader können dann dem Formularfeld keinen Namen zuweisen. Die zugängliche Lösung besteht in der Zuweisung eines Labels mit einem *label*-Element:

```
<label for="Search">Name</label><input type="text" id="Search" />
```

Damit der Modellcompiler diesen Code erzeugen kann, benötigt er das Name-Attribut *Search Lecture* der Interface-Komponente. Das Attribut *ObjectType* deklariert die Komponente als Texteingabefeld. *SearchTarget* ist der Suchstring selbst. Die beiden angegebenen Methoden deklarieren einmal, dass bei Betätigung des Submit-Buttons die Suchfunktion der Applikation aufgerufen wird (*searchLectures*). Als Rückgabewert wird eine Liste von Vorlesungen mit kurzen Beschreibungen erwartet. Die zweite Methode *showList* dient zur Verbesserung der Usability und liefert zu jedem String im Suchfeld eine Liste passender Vorlesungen. Sie kann über AJAX-Technologien integriert werden und ergänzt die Funktionalitäten. Für den Interface-Modellierer gestaltet sich die Situation dann intuitiv, da es selbsterklärend ist, dass die Interfacekomponente ein Label bekommt und es ist für ihn transparent, dass der Modellcompiler daraus barrierefreien Quellcode erzeugt.

2.3 Konkretes Design und Layout der Benutzungsschnittstelle

Den Abschluss des Modellierungsprozesses bildet das Modell des konkreten Layouts. Die physischen Präsentationsparameter – bspw. Größe und Farbe der Objekte – werden beschrieben. Das Klassendiagramm des APM kann entsprechend erweitert werden. Das abstrakte Präsentationsmodell liefert bspw. für den grafischen Entwurf der Oberfläche die strukturellen Vorgaben. Es benennt die Objekte, die integriert werden müssen und welche Eigenschaften und Funktionalitäten sie besitzen.

Design und Layout von Webanwendungen sind in hohem Maße kreative Prozesse und lassen sich nur schwer in die abstrakte Beschreibung von Modellen einbinden. Das betrifft die Gestaltung grafischer Oberflächen ebenso wie die Produktion von akustischen Inhalten wie Sounds oder Musik. UML bietet keine Unterstützung als Notationssprache für Layoutaspekte. Es scheint auch nicht zwangsweise notwendig, das Layout selbst zu formalisieren. Vielmehr ist es notwendig, dass die verwendete Technologie es gestattet, die semantischen Informationen der Modelle mit dem Layout der Oberfläche zu verknüpfen. Zahlreiche Techno-

logien, die heute bereits im Gebrauch sind, unterstützen diese Verbindung. Zu nennen sind hier XHTML+HTML, SVG oder VoiceXML, aber auch Flash oder JavaScript. Die Zukunft wird mit der Durchsetzung von WCAG 2.0 (WCAG 2008) und WAI-Accessible Rich Internet Applications (WAI-ARIA 2009) weitere Verbesserungen bringen. Die WAI-ARIA beschreiben die Anforderungen barrierefreier, dynamisch erzeugter Webinhalte und ergänzen die WCAG.

3 Schlussteil

Der hier exemplarisch vorgestellte Ansatz demonstriert Möglichkeiten und Probleme der modell- und benutzungsorientierten Vorgehensweise für die Entwicklung barrierefreier Benutzungsschnittstellen. Die modellzentrierte Vorgehensweise bietet den Vorteil, dass die semantischen Bedeutungen und Beziehungen der Interaktionskomponenten bereits im Modell beschrieben und dieses Wissen damit explizit formuliert wird. Dadurch kann es dann auch dem Benutzer zugänglich gemacht werden. Der benutzungszentrierte Ansatz ergänzt dieses Vorgehen um eine an den Zielen und Vorstellungen des Benutzers ausgerichtete Vorgehensweise. Zur Notation bietet sich UML an, da es auch textbasiert dargestellt werden kann und somit das Vorgehensmodell selbst barrierefrei gestaltet werden kann.

Es wurde die Eignung der Vorgehensweise demonstriert, barrierefreie Benutzungsschnittstellen zu entwickeln. Existierende Ansätze wie useML und UWE sind dazu entsprechend zu ergänzen. Das Verfahren ist insbesondere geeignet, Softwarearchitekten und Entwickler mit geringen Fachkenntnissen bzgl. der Zugänglichkeit von Benutzungsschnittstellen zu unterstützen. Des weiteren lässt sich diese Vorgehensweise auf weitere Anwendungsfelder – Benutzungsschnittstellen von Webanwendungen, von Desktop-basierten Anwendungen, von Automatisierungssystemen etc. – ausdehnen und stellt damit einen allgemeinen, geeigneten Ansatz dar, um den aktuellen Problemen der Entwicklung barrierefreier Benutzungsschnittstellen zu begegnen. Weiterhin lässt sich das Verfahren in ein nach der ISO 13407 empfohlenes partizipatives, iteratives und prototypbasiertes Vorgehen einbinden. Detaillierte Analysen dazu stehen noch aus.

Literaturverzeichnis

DIN EN ISO 13407 (1999). Benutzer-orientierte Gestaltung interaktiver Systeme. DIN Deutsches Institut für Normung. Berlin: Beuth.

DIN EN ISO 9241-171 (2008). Leitlinien für die Zugänglichkeit von Software. Ergonomie der Mensch-System-Interaktion. DIN Deutsches Institut für Normung. Berlin: Beuth.

Goble, C., Harper, S., Stevens, R. & Yesilada, Y. (2005). DANTE – Mobility Support for Visually Impaired Web Travellers. University of Manchester (UK): School of Computer Science, http://dante.man.ac.uk, Abgerufen: 1.6.2009.

HUTN (2004). Human-usable Textual Notation Specification V 1.0. Needham (MA): OMG Object Management Group, http://www.omg.org/technology/documents/formal/hutn.htm, Abgerufen: 1. 6.2009.

Knapp, A., Koch, N., Wirsing, M. & Zhang, G. (2007). UWE – Ein Ansatz zur modellgetriebenen Entwicklung von Webanwendungen. i-com, 3, 5-12.

Luyten, K. (2004). Dynamic User Interface Generation for Mobile and Embedded Systems with Model-based User Interface Development. Transnational University Limburg: School of Information Technology.

Meixner, G. & Görlich, D. (2008). Aufgabenmodellierung als Kernelement eines nutzerorientierten Entwicklungsprozesses für Bedienoberflächen. Berlin: Workshop "Verhaltensmodellierung: Best Practices und neue Erkenntnisse", Fachtagung Modellierung.

Norman, D. (1989). The Design of Everyday Things. New York: Currency Doubleday.

Norman, D. (1998). The Invisible Computer. Cambridge (MA): The MIT Press.

Reuther, A. (2003). useML – systematische Entwicklung von Maschinenbediensystemen mit XML. Universität Kaiserslautern.

Trætteberg, H. (2002). Model-based User Interface Design. Trondheim: Norwegian University of Science and Technology, Department of Computer and Information Sciences.

WAI-ARIA (2008): WAI-Accessible Rich Internet Applications. Cambridge (MA): World Wide Web Consortium, http://www.w3.org/TR/wai-aria, Abgerufen: 1.6.2009.

WCAG 1.0 (1999). Web Content Accessibility Guidelines 1.0. Cambridge (MA): World Wide Web Consortium, http://www.w3.org/TR/WCAG10, Abgerufen: 1.6.2009.

WCAG 2.0 (2008). Web Content Accessibility Guidelines 2.0. Cambridge (MA): World Wide Web Consortium, http://www.w3.org/TR/WCAG20, Abgerufen: 1.6.2009.

Kontaktinformationen

Wenn Sie Fragen zum Beitrag haben, wenden Sie sich an:

Helmut Vieritz

Abteilung Neue Medien in Forschung und Lehre
Rechenzentrum Universität Stuttgart
D-70550 Stuttgart, Allmandring 30a
Tel.: +49 (0) 711-685-63835
E-Mail: helmut.vieritz@rus.uni-stuttgart.de