VDE-Kongress "Innovationen für Menschen" , Volume 1, pp. 193-198, 18.-20. Oktober, Berlin, VDE Verlag GmbH, Berlin, ISBN 3-8007-2827-3

Expertensystem zur kritierienorientierten Bewertung der Gebrauchsfähigkeit von Dialogsystemen

Nico Hamacher, Karl-Friedrich Kraiss
Lehrstuhl für Technische Informatik, RWTH Aachen

Kurzfassung

Wegen immer kürzeren Entwicklungszyklen muss die Usability-Bewertung der Benutzungsoberfläche möglichst reibungslos in den Entwicklungsprozess integriert werden. Dabei sind vor allem automatische Bewertungen hilfreich. Dieser Beitrag beschreibt eine Möglichkeit der Usability-Bewertung von Benutzungsoberflächen von Dialogsystemen mit Hilfe von Guidelines. Die Bewertung erfolgt dabei automatisch unter Verwendung eines Expertensystems. Eine normierte Beschreibung des zu bewertenden Dialogsystems stellt die Faktenbasis dar. Die Regeln entsprechen den zum Einsatz kommenden Guidelines. Neben der Überprüfung von Gestaltgrundsätzen lassen sich auch Bedienvorgänge simulieren. Ontologien geben dazu den Orientierungsrahmen während Lern- und Erinnerungsprozesse ebenfalls Bestandteil der Simulation sind.

1 Einleitung

Bei der Entwicklung interaktiver Systeme spielen die Begriffe Ergonomie und Gebrauchstauglichkeit (im Folgenden Usability genannt) eine immer wichtigere Rolle [11]. Durch eine stetige Funktionszunahme finden sich heutzutage in immer mehr Geräten des alltäglichen Gebrauchs bildschirmgestützte Bediensysteme mit einem Graphical User Interface (GUI), z.B. in Fahrerinformationssystemen, Videorekordern oder in modernen Waschmaschinen. Dadurch steigen die Anforderungen an Messmethoden und –möglichkeiten zur Usability-Bewertung. Bei der Systementwicklung gehören Guidelines[1] zum Standard, deren Einhaltung überprüft werden muss. Idealerweise sollte eine solche Überprüfung parallel zur Entwicklung erfolgen und durch automatischen Einsatz möglichst wenig Ressourcen (z.B. Zeit, Geld) benötigen.
Im Folgenden werden die Möglichkeiten zur Usability Bewertung mit dem Schwerpunkt auf Guideline-Methoden erläutert. Anschließend folgt die Vorstellung der expertensystemgestützten Guideline-Bewertung interaktiver Geräte, die von einem Anwendungsbeispiel begleitet wird.

2 Möglichkeiten der Usability Bewertung

Das Gebiet der Usability-Bewertung ist sehr umfangreich und beinhaltet sehr unterschiedliche Anwendungsgebiete. Zur besseren Übersicht werden im Folgenden der Aufbau von Dialogsystemen, unterschiedlichen Erscheinungsformen, Bewertungsmöglichkeiten sowie eine Übersicht über aktuelle Werkzeuge aufgeführt.

2.1 Komponenten eines Dialogsystems

Ein Dialogsystem lässt sich nach dem Seeheim-Modell in folgende Komponenten unterteilen (Abbildung 1) [13]. Die Präsentationskomponente (GUI) dient der Informationsein- und -ausgabe und stellt somit die Schnittstelle zum Benutzer dar. Die Aufgabe der Dialogkontrolle ist, die Benutzeraktionen zu erkennen und entsprechend dem Kontext mit Darstellung von Informationen bzw. Aktivierung von Funktionen zu reagieren. Der Zugriff auf die eigentliche Funktionalität der Anwendung ist über die Anwendungsschnittstelle gegeben. Diese Struktur ist heute die Grundlage der meisten dialoggesteuerten Systeme.

Bild 1: Komponenten eines Dialogsystems nach dem Seeheim-Modell.

Eine Usability-Bewertung umfasst die Präsentationskomponente und die Dialogkontrolle, da Sie direkten Einfluss auf das Erscheinungsbild des Gerätes für den Benutzer haben.

[1] Der Begriff *Guidelines* wird hier auch synonym für Styleguides bzw. Richtlinien verwendet.

2.2 Usability Bewertung während des Entwicklungsprozesses

Der Entwicklungsprozess eines interaktiven Gerätes lässt sich in verschiedene Phasen unterteilen (Abbildung 2). Zu Beginn wird in der Definitionsphase ein allgemeines Systemkonzept ausgearbeitet. Auf Basis dieses Konzepts sind in der Spezifikations- und Entwurfsphase die notwendigen (Unter-)Systeme und Funktionen zu definieren. In der anschließenden Realisierungssphase erfolgt die detaillierte Realisierung der einzelnen Systemkomponenten, die in der Integrationsphase zu dem funktionsfähigen Gesamtsystem zusammengesetzt werden. Die abschließende Nutzungsphase stellt den Einsatz des Gerätes entsprechend des konzipierten Verwendungszwecks dar.

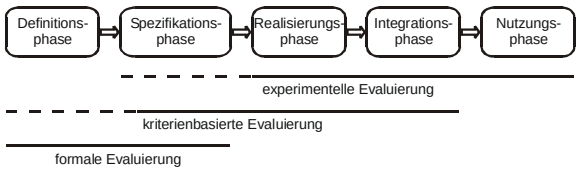

Bild 2: Phasen der Systementwicklung und Möglichkeiten der Usability-Bewertung.

Die Methoden der Usability-Bewertung während der Systementwicklung lassen sich in formale, kriterienbasiert und experimentelle Methoden einteilen. Die formale Evaluierung bewertet das System auf der Grundlage eines normativen Benutzermodells [11]. Diese Methoden können zwar sehr schnell durchgeführt werden, ihre Aussagekraft ist jedoch begrenzt. In der experimentellen Evaluierung wird ein Prototyp des Systems von Probanden bedient und anschließend bewertet. Diese Bewertungsmethode ist sehr zeit- und kostenintensiv und eignet sich daher nur bedingt zur häufigen Durchführung. Bei der kriterienbasierten Evaluierung kommt Expertenwissen in Form von Guidelines oder Richtlinien zum Einsatz, wobei diese meist lediglich in Papierform vorliegen und die Umsetzung und Interpretation häufig schwierig ist [9].

2.3 Kriterienbasierten Evaluierung von Dialogsystemen

Es lassen sich drei unterschiedliche Typen von Dialog-Applikationen unterscheiden, für die entsprechende Kriterienkataloge bzw. Guidelines existieren:

- *WIMP*[2]-Anwendungen stellen die klassischen Softwareprogramme moderner Betriebssysteme dar. Es gibt zahlreiche Guidelines für WIMP Anwendungen. Die meisten Guidelines betreffen dabei die Anordnung der Eingabeelemente (Buttons, Eingabefelder) im Dialogfenster. Häufig kümmert sich dabei das Betriebssystem selbst um die Darstellung, so dass z.B. die Farbgebung oder Schriftgrößen bereits festgelegt ist und nicht mehr vom Designer bestimmt werden muss.
- *Websites* bieten primär Informationen, die über Computernetze (Intranet, Internet) abgefragt werden können. Für Websites gibt es zahlreiche Guidelines, die die Zugriffsperformance, den syntaktischen Aufbau der HTML[3]-Syntax sowie die Usability betreffen [2]. Die Dialogkontrolle ist häufig per HTML direkt in die Syntax eingebunden. Daher liegt der Schwerpunkt der Usability-Bewertung im Syntaxcheck zur Überprüfung von Inkonsistenzen und Oberflächengestaltung.
- *Embedded Systems*[4] kommen in den letzten Jahren immer häufiger zum Einsatz. Aufgrund der verschiedenen Anwendungsgebiete der Systeme gibt es entsprechend unterschiedliche Guidelines, z.B. für Fahrerinformationssysteme [7], On-Sreen-Displays zur Bedienung von Fernsehgeräten oder Videorekordern [15], oder Mobiltelefone [16]. Da die Darstellungsfläche für Informationen bei diesen Geräten i.d.R. stark begrenzt ist, wird der Zugriff auf Funktionen per mehrstufigen Menüs geordnet, durch das der Benutzer navigieren muss. Zu diesen Geräten gibt es aufgrund der Unterschiedlichkeit der Funktionen und Einsatzmöglichkeiten noch sehr wenige akzeptierte Standards, die geräte- und funktionsübergreifend die Bedienungen vereinheitlichen.

Zudem gibt es zahlreiche Guidelines, die unabhängig von der Anwendung prinzipielle Anforderungen an GUI-Systeme definieren. Diese Guidelines sind jedoch meist in ihrer Formulierung so allgemein gehalten, dass eine Interpretation und konkrete Anwendung schwer fällt [9]. Ein Beispiel dafür ist die ISO 9241 [5], die die Effektivität als Maß der Gebrauchstauglichkeit durch „Die Genauigkeit und Vollständigkeit, mit der ein Benutzer ein bestimmtes Ziel erreicht“ lediglich sehr allgemein beschreibt.
Für eine automatische Bewertung muss eine Operationalisierung von Guidelines erfolgen, die jedoch nur für besonders konkrete und detaillierte Guidelines Erfolg versprechend ist. Es gibt mittlerweile einige Ansätze einer solchen Operationalisierung. Zu nennen ist vor allem die Arbeit von Farenc, die insgesamt 205 Guideline-Regeln benennt, die sich zur automatischen Bewertung eignen [8].

[2] *WIMP* (Windows, Icons, Menus, Pointers) ist ein Synonym für fenstergestützte Programme.

[3] *HTML* (Hyper Text Markup Language) ist die Sprache, in der Websites überwiegend programmiert sind.

[4] *Embedded Systems* sind Computersysteme, die in einem technischen System „eingebettet“ sind. Der Begriff ist sehr weit definiert; in dieser Arbeit soll der Begriff nur für solche Systeme stehen, die über eine Mensch-Maschine-Schnittstelle verfügen.

2.4 Stand der automatisierten kriterienorientierten Bewertung

Es gibt einige Werkzeuge, die eine kriterienbasierte Usability-Bewertung automatisch erlauben, dazu gehören z.B. AIDE [17] und Sherlock [14] (für WIMP-Anwedungen) sowie A-Promt [1] und Bobby [3] (für Websites). Diese Werkzeuge ermöglichen jedoch lediglich eine statische Analyse durch hart in den Programmquelltext einkodierte Regeln. Der grundlegende Aufbau, der diesen Werkzeugen zu Grunde liegt, ist in Abbildung 3 dargestellt. Nach der Extraktion der GUI-Daten aus der Anwendung werden diese in unterschiedlichen Prozeduren analysiert. Dabei korrespondiert eine Prozedur i.d.R. jeweils mit einer Guideline. Das Ergebnis dieser Bewertung steht anschließend als Report zur Verfügung.

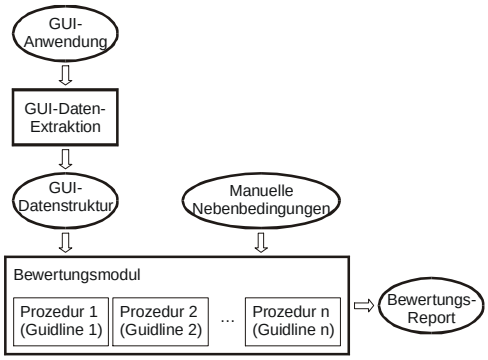

Bild 3: Softwarearchitektur bestehender automatischer Bewertungs-Werkzeuge

Der Nachteil dieser Vorgehensweise liegt im statischen Aufbau sowie der eingeschränkten Bearbeitungsmöglichkeiten der Regeln. Zwar lassen sich manuell Nebenbedingungen für die einzelnen Prozeduren festlegen, eine Anpassung bestehender oder Erstellung neuer Regeln ist jedoch nur durch eine Änderung des Programmcodes möglich.
Die Anwendung dieser Werkzeuge liegt bislang lediglich in der Gestaltüberprüfung der GUI. Ebenso existieren bislang nur sehr wenige Werkzeuge zur automatischen Bewertung der Dialogkontrolle. Dazu gehört z.B. das auf GOMS basierende TREVIS, bei dem jedoch ebenfalls die Methoden fest in den Sourcecode einprogrammiert sind [11].

3 Systemarchitektur

Um die in Kapitel 2 vorgestellte Möglichkeit der automatischen Guideline-Bewertung zu erweitern, wird in diesem Kapitel ein Konzept vorgestellt, das eine Bewertung mit Hilfe eines Expertensystems realisiert (Abbildung 4). Das Konzept erweitert die Systeme von Ericsson [6] und Beirekdar [2].

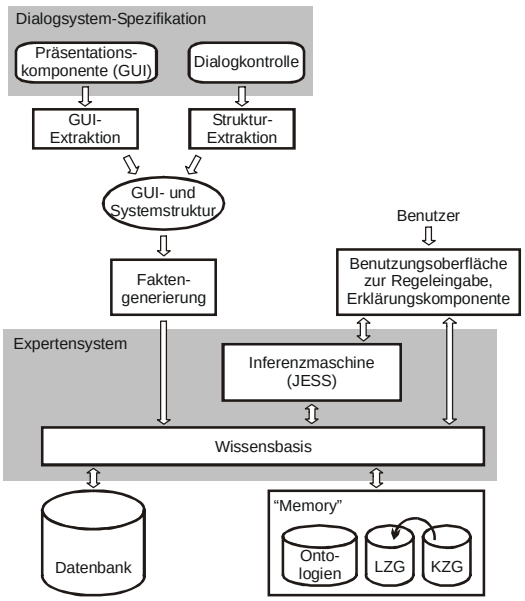

Bild 4: Architektur der Guideline-Bewertung mit einem Experten-System

Eine Beschreibung des Dialogssystems, z.B. aus technischen Spezifikationen, die in der Spezifikationsphase definiert wurden, liegt als Bewertungsbasis zu Grunde. Zur Veranschaulichung der Möglichkeiten steht das Benutzerinterface eines Autoradios mit CD zur Verfügung (Abbildung 5). Das Autoradio verfügt über die gängigen Funktionen. Aktuell ist ein Radiosender eingestellt. Das Gerät besitzt ein Drehrad zur Lautstärkeeinstellung (links), 11 Buttons sowie ein Navigationskreuz (rechts). Ein zentrales Display stellt Informationen zur Verfügung.

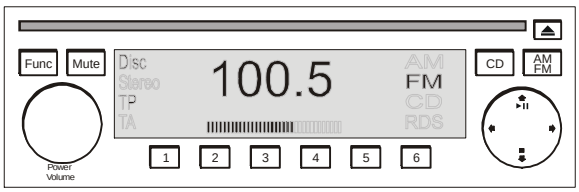

Bild 5: GUI eines Autoradios zur beispielhaften Bewertung.

Die Darstellung der GUI lässt sich dabei angelehnt an die CUA[5]-Norm beschreiben. Diese Norm schreibt eine objektorientierte Dekomposition der grafischen Objekte vor. Dabei werden die einzelnen Elemente durch Attribute sowie Beziehungen zu weiteren Elementen, aus denen sie aufgebaut sind, beschrieben. Als Beispiel sei die Beschreibung des Knopfs *Func* (s. Abbildung 5) dargestellt:

```
But_Func{  Type: button
           x: 43, y: 20,
           layer: 31,
```

[5] *CUA* (Common User Application) bezeichnet den Standard der IBM für die Benutzeroberfläche von Anwendungen auf unterschiedlichen Hardware-Architekturen [12].

```
            width: 40, height: 24,
            shape: rectangle,
            color: silver,
            label: Lab_Func,
            …                    }
Lab_Func{text: 'Func',
            font: Arial,
            size:  6,
            color: black,
            …                    }
```

Die GUI ist aus mehreren grafischen Layern zusammengesetzt (0 entspricht dem hintersten Layer). Dadurch kann z.B. beschrieben werden, welche Beschriftungen zu welchen Elementen gehören (Überdeckung eines Elementes durch Schrift) bzw. ob ein Element durch Überdeckung gar nicht sichtbar ist.
Die GUI-Elemente sind darüber hinaus in die Klassen Eingabeelemente (z.B. Buttons), Displays (z.B. LEDs, LCDs) und sonstige Elemente (z.B. Linien, Logos) eingeteilt.

Die internen Systemzustände und –übergänge der Dialogkontrolle sind häufig als Zustandsautomaten festgelegt. Dabei kommen Formalismen wie Statecharts oder SDL[6] zum Einsatz. Diese Formate lassen sich ebenfalls automatisch auslesen. Eine beispielhafte Modellierung der Dialogkontrolle als Statechart ist in Abbildung 6 dargestellt. Die Aktionen entsprechen dabei Tastendrücken („cd_in“ entspricht dem Einlegen einer CD).

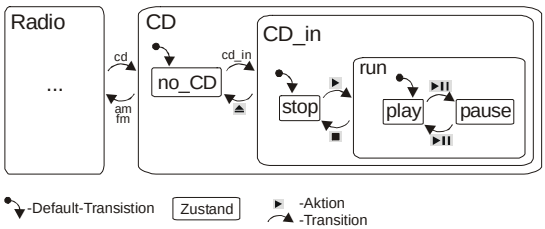

Bild 6: Statechart der CD-Bedienung des Autoradios.

Ein Programmmodul generiert aus den GUI- und Strukturdaten automatisch Fakten für das Expertensystem. Die Informationen werden lediglich in ein Format übersetzt, das die Inferenzmaschine lesen und bearbeiten kann, die Struktur und der Aufbau bleiben gleich.

Eine Benutzungsoberfläche erlaubt die Eingabe und das Editieren der zu formulierenden Guidelines als Regeln (s. Abbildung 4). Dabei kommt die deklarative Sprache LISP zum Einsatz. Regeln haben in LISP prinzipiell einen Wenn-Dann-(Prämisse-Konklusion)-Aufbau. Die Prämisse stellt Anforderungen an Fakten. Falls Fakten existieren, die diese Anforderungen erfüllen, wird die Konklusion aktiviert. Diese kann Funktionsaufrufe enthalten bzw. Fakten modifizieren, neu generieren oder entfernen.

[6] SDL (Specification and Description Language) erlaubt die Festlegung aller Funktionsparameter eines Gerätes in elektronischer Form

Aufgrund der Komplexität von LISP ist es sinnvoll, die Programmierung der Regeln zu vereinfachen. Dazu schränkt die Benutzungsoberfläche die Regeleingabe insoweit ein, dass in der Konklusion lediglich auf Fakten zugegriffen werden kann, die in der Faktenbasis tatsächlich vorhanden sind.
Dieses Vorgehen lehnt sich an die GDL (Guideline Description Language [2]) an. Die GDL erlaubt die Festlegung von Regeln durch die Definition zu bewertender Element (GUI- oder Strukturelemente), mit erlaubten Wertebereichen.

Zur Bewertung der Gestaltung genügen Regeln, die das GUI-Format z.B. auf die Farbgebungen der Elemente einzeln oder zueinander, Ausdehnungen, Schriftgrößen oder die Überdeckung von Elemente überprüfen. Somit lassen sich einfach Designfehler im Layout bzw. der Spezifikation erkennen. Als Beispiel dient die Regel, die überprüft, ob ein Label (Variable `a`) die gleiche Farbe (Variable `c`) wie ein dahinter liegendes Element (`b`) hat. Die Überprüfung der Überlappung übernimmt die Funktion `over`:

```
(defrule check_labelcolor
 (label(id ?a) (layer ?l)) (color ?c))
 (elem (id ?b) (layer (<?l)) (color ?c))
 (over (?a ?b))
=>
 (out "Label " ?a " has same color "
 "than background and cannot be read!")
)
```

Zur Simulation einfacher Lernprozesse dient das „Memory“-Modul. Dort können entsprechend markierte GUI- und Strukturelemente gelernt und erinnert werden. Das Kurzeitgedächtnis (KZG) ist als FIFO-Cache mit einer Kapazität von 7 Einheiten angelegt, was gängigen Angaben über die Kapazität des KZG entspricht [18]. Bei mehrfachen Wiederholungen werden die Elemente in Abhängigkeit des Wiedeholungsintervalls aus dem KZG in das Langzeitgedächtnis (LZG) übernommen.

Funktionen sind grundsätzlich mit Begriffen (bzw. Icons) beschriftet. Bei menügestützten Systemen sind diese Begriffe in Menüs gruppiert und mit einem Sammelbegriff versehen. Bei der Bedienung muss auf dem GUI klar ersichtlich sein, auf welcher Menüebene sich der Benutzer befindet und welche Funktionen ihm zur Verfügung stehen. Diese Menüsysteme setzen voraus, dass dem Benutzer die Zuordnungen von Sammelbegriffen zu den entsprechenden Funktionsbezeichnungen klar sind. Dieses Kontextwissen wird dem Expertensystem in Form von Begriffsontologien zur Verfügung, gestellt. Ontologien klassifizieren Begriffe und ordnen diese hierarchisch an (Abbildung 7). So gehört der Begriff „Abspielen“ im Beispiel in Abbildung 7 zu „Bedienung“, das wiederum „CD-Spieler“ zugeordnet ist.
Startpunkt einer Bedienung ist der *idle*-Zustand der Ontologie, Zielpunkt eine Funktion mit einer spezifi-

schen Beschriftung (z.B. „Abspielen“ einer CD). Durch den Aufbau der Ontologie ergibt sich eine Sequenz (in Abbildung 7 durch die umrahmten Begriffe markiert) der Begriffe, die den Zielbegriff immer stärker einschränken. Bei der Auswahl passender Bedienmöglichkeiten ist so eine Orientierung nach dem gegebenen Begriffspfad möglich.

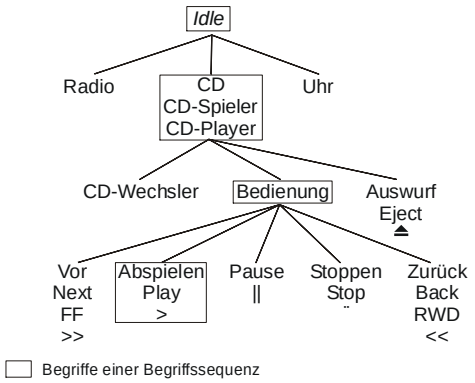

Bild 7: Ontologie am Beispiel der Bedienung eines CD-Spielers in einem KFZ.

Mit Hilfe des „Memory“-Moduls ist es möglich, einfache mentale Lernprozesse eines Benutzers zu simulieren. Durch eine Vorbelegung des LZG mit bereits gelernten Elementen sowie Eingabe unterschiedlich umfangreicher Ontologien können Bedienungen mit unterschiedlichen Wissensständen (z.B. Laie vs. Experte) simuliert werden.

Eine Bewertung nach der Konsistenz, Konformität und Angemessenheit der Displayausgaben ist durch die Simulation einer Bedienaufgabe möglich. Eine Simulation besteht aus mehreren hintereinander ausgeführten Schritten. Dazu gehört die Durchführungen von Bedienaktionen hinsichtlich des aktuellen Zustands sowie Untersuchungen des Displays auf den aktuellen Zustand des Gerätes hin. Die Folgerungsprozesse sowie die Ergebnisse der Simulation werden in der Erklärungskomponente dargestellt (s. Abbildung 4).

Als Beispielaufgabe soll der Benutzer das Autoradio vom Radiobetrieb zum CD-Betrieb umschalten und den 2. Titel der CD anspielen. Hierzu muss die Ontologie zu Beginn mit den entsprechenden Kontextbegriffen vorbelegt sein (s. Abbildung 7), nach denen sich die Simulation bei der Auswahl der Bedienschritte orientiert. Weiterhin müssen Regeln zur Überprüfung der eingesetzten Guidelines formuliert werden. Ein Benutzer verfügt darüber hinaus über Metawissen bezüglich der Bedienung. Dazu gehört z.B. das Wissen, welche Aktionen bei der Betätigung von Eingabeelementen erwartet werden (z.B.: Drehen eines Drehknopfes im Uhrzeigersinn => Inkrementierung eines Wertes; Bewegung einer Auswahl/eines Zeigers nach rechts, nach unten). Dieses Wissen muss ebenfalls als Regeln definiert sein.

Folgende Schritte zur Situationserfassung und Reaktion werden anschließend bei der Aufgabensimulation durchgeführt:
Schritt 1: Zu Anfang erfolgt die Überprüfung des aktuellen Status des Gerätes. Ein sequentielles Durchsuchen der GUI-Repräsentation ergibt, dass aktuell keine Informationen zum CD-Betrieb dargestellt werden. Die Informationen „100,5“ sowie „FM“ entsprechen vielmehr dem Radio-Betrieb.
Schritt 2: Ein Bedienschritt zum Einschalten des CD-Betriebs muss erfolgen, was das Vorhandensein eines Eingabeelement mit der Funktionsbeschriftung „CD“ oder „CD-Spieler“ (s. Ontologie in Abbildung 7) bedingt. Existiert keine solche Taste, dann bricht die Simulation ab, da das Gerät nicht in den CD-Betrieb geschaltet werden kann.
Schritt 3: Nach Betätigung der Taste „CD“ erfolgt erneut eine Überprüfung der aktuellen Displayinformationen, um den CD-Kontext zu bestätigen (s. Schritt 1).
Schritt 4: Konnte der CD-Betrieb bestätigt werden muss das Anzeigeformat auf die aktuelle Titelnummer hin untersucht werden.
Schritt 5: Falls der aktuelle Titel nicht Titel 2 ist müssen erneut Bedienschritte zum Auswählen des 2. Titels erfolgen. Dazu kommen Bedienelemente zum In- oder Dekrementieren mit entsprechender Beschriftung (+, -, <, > o.Ä.) zum Einsatz
Die Schritte 4 und 5 werden solange wiederholt, bis der 2. Titel ausgewählt ist.
Vorgänge wie das Aussuchen eines Bedienelementes zur Durchführung einer Bedienaktion, können bei mehrfacher Wiederholung übersprungen werden. Diese Vorgänge wurden beim ersten Auftreten im „Memory“-Modul gespeichert und stehen in nachfolgenden Schritten als ***gelernt*** zur Verfügung.

Derart lassen sich neben Gestaltungsregeln auch die Anzahl der benötigten bzw. gelernten Bedienschritte, die Eindeutigkeit der Beschriftungen und dargestellten Informationen sowie die Art und Anzahl der eingesetzten Regeln des Metawissens anzeigen bzw. bewerten. Voraussetzung für eine sinnvolle Bewertung sind entsprechend klar formulierte Regeln.

Die eigentliche Auswertung der (Guideline-)Regeln findet in einer Inferenzmaschine statt, welche die Regeln auf die gespeicherten Fakten anwendet. Zum Einsatz kommt dabei die *Java Expert System Shell* (JESS) [10]. JESS ist ein regelbasiertes Expertensystem, welches sich in Funktion und Bedienung an CLIPS [4] orientiert. JESS ist allerdings durch seine Implementierung in Java plattformunabhängig und bietet weitgehende Möglichkeiten der Anpassung der Inferenzmaschine.
Die Inferenzmaschine untersucht Regeln daraufhin, ob bestehenden Fakten die Prämisse erfüllen, so dass die Konklusion gültig wird. Grundsätzlich wird bei der Inferenz zwischen Vorwärts- und Rückwärtsver-

kettung unterschieden. Bei der Vorwärtsverkettung wird eine Hypothese (in der Konklusion) mit bestehenden Fakten gestützt oder widerlegt. Somit lassen sich Regelverletzungen überprüfen; der Fokus liegt auf den Fakten. Mit der Rückwärtsverkettung ist eine Prüfung möglich, ob ausgehend von einer Hypothese, z.B. „das Gerät ist gebrauchsfähig“ oder „es existiert ein Begriff ‚CD’“, stützende Prämissen vorhanden sind, der Fokus liegt auf den Regeln. Beide Mechanismen sind bei der Überprüfung von Guidelines sinnvoll, die Vorwärtsverkettung zur Überprüfung der Fakten, die Rückwärtsverkettung, um die Überprüfung von bestimmten Konventionen (Konklusionen) zu erzwingen.

5 Zusammenfassung

Das vorgestellte Konzept erlaubt die Guideline-Bewertung von Dialogsystemen. Durch die Benutzung einer speziell zur Regelprogrammierung geeigneten Programmiersprache wie LISP ist eine schnelle Eingabe und Bearbeitung von Guidelines gewährleistet. Der Einsatz von JESS erlaubt eine variable und anpassungsfähige Überprüfung von Regeln mit den aus der Systembeschreibung automatisch generierten Fakten. Die Inferenz erfolgt automatisch, so dass eine Usability-Bewertung während der Systementwicklung schnell durchgeführt werden kann. Ein weiterer Vorteil liegt in dem modularen Aufbau sowie der Wiederverwendbarkeit der Regeln.

6 Literatur

[1] Adaptive Technology Resource Center (University of Toronto) and Trade Center (University of Wisconsin): A-Prompt Accessibility Toolkit Project., Canada & USA (April 1999)

[2] Beirekdar, A.; Noirhomme-Fraiture, M.; Vanderdonckt, J.: KWARESMI-Knowledge-based Web Automated Evaluation with REconfigurable guidelineS optiMIzation. Proc. of 2nd Int. Conf. on Universal Access in Human-Computer Interaction UAHCI'2003, Vol. 4, Stephanidis, C. (Ed.), Lawrence Erlbaum Associates, Mahwah, 2003, pp. 1504-1508

[3] Center for Applied Special Technology (CAST): Bobby Version 2.0. HTML Web analyser, National Universal Design Laboratory, Peabody, 2002.

[4] CLIPS: A Tool for Building Expert Systems. http://www.ghg.net/clips/CLIPS.html.

[5] Deutsches Institut für Normung e.V.: ISO 9241 - Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten. Beuth Verlag GmbH, Berlin, 1998

[6] Mikael, E.: User Interface Design Support Tools and Critiquing. In: Proceedings of the 4th Annual IDA Conference on Computer and Information Science. 1994, Linköping University, Sweden.

[7] Europäische Gemeinschaft: Empfehlung der Kommission über sichere und effiziente On-board-Informations- und -Kommunikationssysteme. Europäischer Grundsatzkatalog zur Mensch-Maschine-Schnittstelle. Aktenzeichen K(1999) 4786, http://europa.eu.int/eur-lex/pri/de/oj/dat/2000/l_019/l_01920000125de00640068.pdf, 2000

[8] Farenc, C.: ERGOVAL : une méthode de structuration des règles ergonomiques permettant l'évaluation automatique d'interfaces graphiques. Thèse de Doctorat, l'Université Toulouse, 1997

[9] Farenc, C., Palanque, A.: A Generic Framework based on Ergonomics Rules for Computer Aided Design of User Interface; J. Vanderdonckt, A. Puerta (eds.), Computer-Aided Design of User Interfaces, Kluwer Academics, Dordrecht, 1999

[10] Friedman-Hill, E.-J.: Jess, The Rule Engine for the Java Platform. Version 6.1p6 (21 November 2003). Distributed Computing Systems, Sandia National Laboratories, Livermore, CA.

[11] Hamacher, N.; Kraiss, K.-F.; Marrenbach, J.: Einsatz formaler Methoden zur Evaluierung der Gebrauchsfähigkeit interaktiver Geräte; In: it + ti Informationstechnik und Technische Informatik, Vol. 44, Heft 1 (2002), pp. 49-55, Oldenbourg

[12] IBM Corp.: Object-Oriented Interface Design. Que Corporation, Carmel, USA, 1992.

[13] Pfaff, G. E. (editor): User Interface Management Systems. Springer Verlag, Berlin, 1985.

[14] Mahajan, R.; Shneiderman, B.: Visual & textual Consistency Checking Tools for Graphical User Interfaces. Technical Report CS-TR-3639 (1996), University of Maryland.

[15] Peng, C.; Cesar, P.; Vuorimaa, P.: Integration of Applications into digital Television Environment. Proceedings of the 7th International Conference on Distributed Multimedia Systems, September 26-28, 2001, Taipei, Taiwan, pp. 266-272.

[16] Roe, P. (Edt.): Guidelines-Booklet on Mobile Phones – A COST 219bis Guidebook. http://www.stakes.fi/cost219/mobiletelephone.htm, updated 2001

[17] Sears, A.: AIDE: A Step Toward Metric-based Interface Development Tools. Proceedings of the 8th annual ACM symposium on User interface and software technology, Pittsburgh PA USA, pp 101-110, 1995.

[18] Zimbardo G; Gerrig, R.: Psychologie. Springer Verlag, 7. Auflage , Berlin, 1999