Ausarbeitung

Milen Koychev

Multikanalfähigkeit von Dialogsystemen

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Milen Koychev

Multikanalfähigkeit von Dialogsystemen

Ausarbeitung im Rahmen der Vorlesung Anwendungen 2
im Studiengang Informatik
am Studiendepartment Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuer : Diplom. Inform. Birgit Wendholt
Zweitbetreuer: Prof. Dr. rer.nat. Kai von Luck

Abgegeben am 22. Februar 2007

Milen Koychev

Thema der Ausarbeitung

Multikanalfähigkeit von Dialogsystemen

Stichworte

Multikanalfähigkeit, Dialogsysteme, Softwareergonomie, Softwareengineering, Softwarearchitektur

Kurzzusammenfassung

Die heutzutage existierenden bzw. sich schnell entwickelnden Technologien im Bereich der Hardware- und Softwareherstellung ermöglichen den Menschen, Dienste verschiedener Softwaresysteme auf unterschiedlichen Hardwareplattformen in Anspruch zu nehmen. Dies fordert eine Optimierung des Softwareherstellungsprozesses, indem die Software nicht mehr für eine einzelne Endplattform entwickelt wird, sonder für einen multiplattform- bzw. multikanalfähigen Einsatz konzipiert werden muss. Diese Arbeit stellt Verfahren zur Sicherung der Multikanalfähigkeit von Dialoganwendungen vor. Im Anschluss werden die vorgestellten Verfahren mit deren Vor- bzw. Nachteilen verglichen und bewertet.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einführung und Motivation

Die heutzutage existierenden bzw. sich schnell entwickelnden Technologien im Bereich der Hardware- und Softwareherstellung ermöglichen den Menschen, Dienste verschiedener Softwaresysteme auf unterschiedlichen Hardwareplattformen in Anspruch zu nehmen. Dabei sind solche Systeme so konzipiert worden, dass diese eine Mensch-Maschine-Interaktion erfordern, indem der Mensch durch geeignete Eingaben bzw. Systemausgaben den Ablauf kontrollieren und steuern kann.

In vielen Fällen unterscheiden sich die Eigenschaften (wie z.B. Rechenkapazität, Eingabe- und Ausgabemöglichkeiten usw.) der Hardwareplattformen von einander sehr stark. Um dem Menschen Interaktion mit demselben Dienst bzw. Softwareprodukt auf unterschiedlichen Geräten zu ermöglichen, müssen Teile des entsprechenden Softwaresystems für jede Plattform optimiert bzw. neu entwickelt werden. Solche Systemänderungen werden oft nicht direkt vom Benutzer des Systems wahrgenommen und bleiben somit „transparent“. Eine andere Situation tritt auf, wenn es sich um Modifikationen der Präsentationssicht einer Anwendung handelt. Diese wirken sich direkt auf die Mensch-Maschine-Interaktionen aus und können die Bedienbarkeit sowie die Komplexität der Software beeinflussen.

Um trotzt der großen Unterschiede zwischen den Hardwareplattformen die Bedienbarkeit der Software auf unterschiedlichen Geräten zu garantieren sowie die Komplexität der Software und des Softwareherstellungsprozesses zu begrenzen, sind spezielle Verfahren zur dynamischen Optimierung der Präsentation entwickelt worden.

Ziel dieser Ausarbeitung ist, einen Überblick der Verfahren zur dynamischen Optimierung der Präsentationssicht einer Dialoganwendung zu geben. Dabei werden diese Verfahren mit deren Stärken, Schwächen und Einsatzmöglichkeiten vorgestellt und im Anschluss analysiert bzw. bewertet. Die Ergebnisse dieser Arbeit werden als Grundlage für weitere Forschungsarbeiten (die Ausarbeitung im Rahmen der Seminar-Ringvorlesung sowie die im Sommersemester 2007 geplanten Masterarbeit an der Hochschule für Angewandte Wissenschaften Hamburg) auf dem Gebiet der Multikanalfähigkeit von Dialoganwendungen dienen.

Diese Ausarbeitung bezieht sich im Wesentlichen auf die im Literaturverzeichnis angegebenen Quellen sowie auf die praktischen Erfahrungen während des Projektes „Flughafen“ im Studiengang Informatik-Master im WS06/07 an der Hochschule für Angewandte Wissenschaften Hamburg.

2 Multikanalfähigkeit: Definition

In diesem Kapitel wird der Bergriff „Multikanalfähigkeit“ im Softwareumfeld erläutert. Dabei wird das Untersuchungsgebiet dieser Ausarbeitung im Kontext der Multikanalfähigkeit eingegrenzt.

2.1 Was ist Multikanalfähigkeit?

Bevor die unterschiedlichen Lösungsansätze zur Sicherung der Multikanalfähigkeit einer Anwendung vorgestellt werden, wird das Thema „Multikanalfähigkeit von Dialogsystemen“ im Kontext dieser Arbeit präziser definiert. Als erstes wird der Begriff „Multikanalfähigkeit“ erläutert.

Multikanalfähigkeit im Bereich der Technik und Technologie wird heutzutage in der Fachliteratur als die Fähigkeit eines Systems bezeichnet, über unterschiedliche Kanäle Daten mit weiteren Systemen oder Teilen des Gesamtsystems auszutauschen. Dabei wird der Begriff „Kanal“ in erster Linie nicht genau definiert.

Im Bereich der Dialogsysteme werden die Kanäle durch das Smartkom-Models (s. [smartkom]) als unterschiedliche Mensch-Maschine-Kommunikationsmedien (physikalische Informationsträger) und deren Zusammensetzung bezeichnet. Durch diese Medien werden die unterschiedlichen menschlichen Sinne angesprochen (s. Abbildung 2-1).

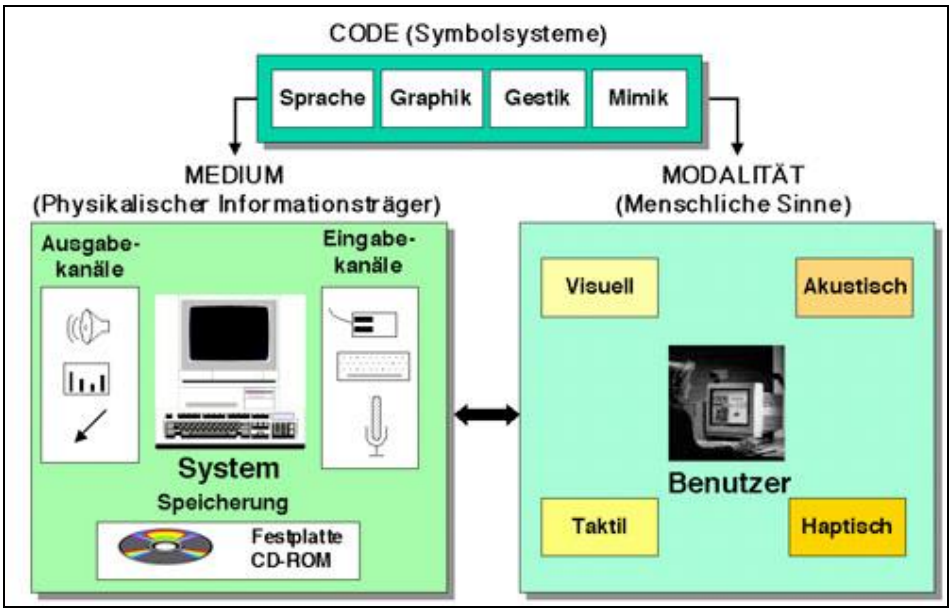

Abbildung 2-1 Smartkom-Modell [smartkom]

Ein weiterer Begriff tritt im Rahmen des Smartkom-Modells auf – der Code. Dieser stellt die eigentliche Information, die ausgetauscht werden muss, dar. Es gibt laut [smartkom] vier Informationsarten (Codes): Sprache, Graphik, Gestik und Mimik.

Jede Informationsart kann über unterschiedliche Medien ausgetauscht werden. Eine Sprachinformation kann über ein Sprachmedium (z.B. Lautsprecher) oder über ein Visualisierungsmedium (z.B. Bildschirm) ausgegeben werden. Weiterhin kann solche Information über Tastatur oder Mikrofon eingegeben werden. Ebenso könnte ein Bildschirm gleichzeitig für verschiedne Codes eingesetzt werden z.B. für Sprachausgabe in Form eines Textes und Darstellung einer grafischen Information in Form eines Bildes.

Multikanalfähigkeit von Dialogsystemen aufgrund eines solchen umfangreichen Modells zu untersuchen, wird ziemlich interessant sein. Dennoch würde eine solche Untersuchung das Rahmen dieser Arbeit sprengen. Um trotzt der großen Komplexität des Modells erste Erfahrungen auf diesem Gebiet in Richtung Dialogsysteme machen zu können, wird das Smakrtkom-Modell vereinfacht, indem nur ein Teil des Modells in Betracht gezogen wird.

Multikanalfähigkeit bei Dialogsystemen wird als Benutzereingaben, Benutzerausgabengaben sowie weitere Mensch-Maschine-Interaktionen auf unterschiedlichen Gerätetypen definiert. Das Bildschirmmedium spielt in dieser Betrachtung eine zentrale Rolle. Als Kanäle werden die unterschiedlichen Bildschirmgrößen bei den auf dem Markt vorhandenen Geräten (wie z.B. PDA, Smartphone, Ultramobile PCs. Systeme mit Ausgaben auf Powerwalls) und deren Eingabemöglichkeiten betrachtet.

2.2 Fazit

Im Rahmen dieser Ausarbeitung wird das Smartkom-Modell vereinfacht, indem nur ein Aspekt (Bildschirmmedium als Interaktionskanal) der Multikanalfähigkeit in den Vordergrund gestellt wird. Dennoch verlangt ein solches Modell von den Softwareentwicklern, bestimmte Vorkehrungen zu treffen, um die Multikanalfähigkeit der Software zu sichern bzw. zu erzielen. Welche Vorkehrungen genau das sind, wird in dem nächsten Kapitel diskutiert.

3 Multikanalfähigkeit: Realisierung

In diesem Kapitel werden die unterschiedlichen Verfahren zur Sicherung der Multikanalfähigkeit der Softwareanwendungen vorgestellt. Es werden die unterschiedlichen Anforderungen an die multikanalfähigen Anwendungen bezüglich deren Einsatzgebietes vorgestellt. Dabei wird das Gebiet der Dialogsysteme ausführlich beschrieben.

3.1 Publishing

Mit der Erscheinung der ersten mobilen Geräte auf dem Markt wurden die ersten Anforderungen auf dem Gebiet der Multikanalfähigkeit definiert. Der einfachste Anwendungsfall, in dem Multikanalfähigkeit gefordert ist, ist Publishing. In diesem Szenario handelt es sich darum, Textinhalte auf unterschiedlichen Gerätetypen zu publiziert bzw. dem Benutzer zur Verfügung zu stellen. Es findet eine sehr begrenzte Mensch-Maschine-Interaktion statt. Der Benutzer muss nur im Stande sein, durch den gesamten Textinhalt zu navigieren.

Die ersten Versuche, eine begrenzte Multikanalfähigkeit zu gewährleisten, haben sich darauf begrenzt, den Text für die unterschiedlichen mobilen Gerätetypen aufzubereiten. Dabei wird der Text in Seiten so aufgeteilt, dass die Seitengröße der Bildschirmgröße des Endgerätes entspricht, auf dem der Textinhalt konsumiert wird. Das Verfahren erzielt nicht immer für den Menschen optimale Ergebnisse. Oft werden zusammenhängende Textteile über mehre Seiten verteil, was die Benutzerakzeptanz dieses Verfahrens in Frage stellt.

Um die Benutzerzufriedenheit zu steigern, wurde die Textaufteilung aufgrund der Semantik der Inhalte entwickelt (s. [Pfeifer06]). Die Semantik wird in semantische Ebenen aufgeteilt und als semantische Links (ähnlich wie HTML-Links oder der WAP-Ansatz [openmobilealliance]) visualisiert. Somit hat der Benutzer die Möglichkeit, nicht nur durch den einfachen Textinhalt zu navigieren, sondern auch die Textsemantik als Navigationsmerkmal zu benutzen. Zusammenhängende Textinhalte sollen möglichst auf einer Seite dargestellt werden. Dabei werden die Grenzen dieses Verfahrens bei großen und semantisch komplizierten Textinhalten schnell erreicht. Der Benutzer muss durch „unendliche“ Hierarchien navigieren, um zum eigentlichen Inhalt zu gelangen.

3.2 Multimedia

Mit der Weiterentwicklung der Technik wurden auch die mobilen Geräte immer leistungsfähiger. Dadurch kamen die Anforderungen, nicht nur einfache Texte multikanalfähig aufzubereiten, sondern auch die multimedialen Inhalte auf den unterschiedlichen mobilen Geräten dem Benutzer zur Verfügung zu stellen.

Der einfachste Weg, Multimediainhalte für unterschiedliche mobile Geräte aufzubereiten, ist die Teile des Multimediastreams (Text, Bild und Ton) an die Eigenschaften der Endplattform anzupassen. Z.B. die Größe und die Qualität der Originalbilder sowie die Tonqualität des Streams werden in Hinsicht auf das mobile Gerät angepasst. Für Geräte mit niedrigen Bildschirmauflösungen und kleineren Bildschirmen werden die Qualität sowie die Größe der Bilder verringert. Ähnliches Verfahren wird auch für Ton-Inhalte eingesetzt. Die Verfahren für die Multikanalaufbereitung von Textinhalten, die im vorigen Kapitel vorgestellt wurden, werden im Bereich Multimedia unverändert eingesetzt.

Um Multimediainhalte noch besser an die Eigenschaften der unterschiedlichen mobilen Plattformen anpassen zu können, wurde der Standard (s. [w3c.smil]) Synchronized Multimedia Integration Language (SMIL) entwickelt. Dieser Standard teilt den Multimediastream einer Anwendung in Unterstreams auf. Jeder Unterstream wird durch seine Eigenschaften und Anforderungen an die Endplattform beschrieben. Wenn ein Benutzer auf seiner Plattform den so definierten Multimediastream konsumieren will, kann er aufgrund der SMIL-Beschreibung des Streams und entsprechend der Möglichkeiten seiner Plattform nur einige der Unterstreams in Anspruch nehmen. Solches Szenario ist denkbar, wenn z.B. das Endgerät über keinen Bildschirm verfügt, aber dennoch den Ton-Stream wiedergeben kann. In solchen Fällen kann sich der Benutzer dafür entscheiden, dass der Multimediastream auch ohne Bildinhalte für ihn relevant ist. Weiterhin kann sich ein Benutzer aufgrund der teueren oder langsamen Leitung zum Streamserver dafür entscheiden, auf einige Unterstreams zu verzichten oder diese in einer sehr niedrigen Qualität zu konsumieren. Somit erweitert der SMIL-Standard die Einsatzmöglichkeiten der „primären“ Datenkompression im Bereich der multikanalfähigen Multimediaanwendungen.

3.3 Dialogsysteme

Wie schon oben beschrieben, bedeutet Multikanalfähigkeit eine Anpassung unterschiedlicher Dateninhalte an die Kanaleigenschaften. Dabei wird der Typ der zu anpassenden Daten durch den Anwendungsfall bestimmt. Typisch für den Bereich der Dialogsysteme ist eine sehre intensive Mensch-Maschine-Interaktion. Diese wird bei den heutigen Software-Dialogsystemen mittels geeigneten Eingabe- bzw. Ausgabemasken - die so genannten Dialoge durchgeführt. Aus diesem Grund bedeutet die Multikanalfähigkeit in diesem Bereich, die Aufbereitung der Dialoge einer Anwendung für den Multikanaleinsatz.

Im Weiteren werden Verfahren vorgestellt, die bei den Dialogsystemen eingesetzt werden können, um die Anwendungen mit Multikanalfähigkeit auszustatten. Dabei werden die vorgestellten Verfahren analysiert und bewertet.

3.3.1 Anpassung der Dialoggröße

Um eine Anwendung multikanalfähig aufzubauen, kann die Anpassung der Dialoggröße nur sehr begrenzt eingesetzt werden. Dennoch werden die Anforderungen an eine multikanalfähige Dialoganwendung durch die Analyse dieses Verfahrens deutlich.

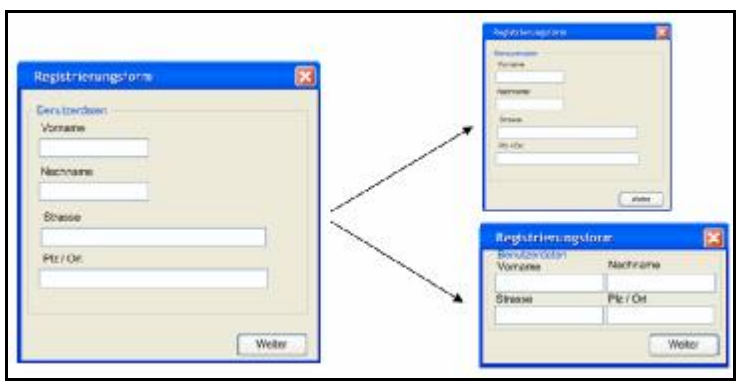

Abbildung 3-1 Anpassung der Dialoggröße

Wenn ein Dialog von einem Benutzer angefordert wird, passt das Verfahren einfach die Dialoggröße an die Kanaleigenschaften (in diesem Fall die Bildschirmgröße) des Endgerätes an, auf dem der Benutzer mit dem Gesamtsystem agiert. Dabei wird die Größe aller Dialogelementen entsprechend angepasst. Somit wird ein einfaches Skalieren des Dialoges erreicht. Dabei werden die Dialoge nur verkleinert und nie vergrößert (s. Abbildung 3-1).

Eine Optimierung dieses Verfahren besteht darin, dass nicht nur die Dialoggröße geändert wird, sondern auch die Anordnung der Dialogelemente. Somit wird eine bessere Ausnutzung der Dialogfläche angestrebt (s. Abbildung 3-1).

Vorteil dieses Verfahrens ist seine Einfachheit. Leider stößt ein solches Verfahren wegen der menschlichen Aufnahmefähigkeit sehr schnell an seine Grenzen. In vielen Fällen sind die Dialoge zu klein, um eine sinnvolle Mensch-Maschine-Interaktion zu ermöglichen. Auch durch die Optimierung wird das Verfahren nicht viel einsatzfähiger. Obwohl der Platz auf den Dialogen besser ausgenutzt wird und die Dialogelemente in einigen Fällen sich noch bedienen lassen, werden die Dialoge schnell unübersichtlich.

Das Verfahren kann sinnvoll nur bei sehr einfachen Dialogen eingesetzt werden. Bei Anwendungen, die nur sehr begrenzte Interaktion mit dem Menschen benötigen, kann dieses Verfahren wegen seiner geringen Komplexität durchaus zum Erfolg führen. Für den professionellen Einsatz müssen jedoch weitere Methoden in Betracht gezogen werden.

3.3.2 Erweitern des Dialogflusses

Um eine effiziente Methode zur Sicherung der Multikanalfähigkeit zu entwerfen, muss der Herstellungsprozess der Dialogsysteme genau analysiert werden. Für den Anwender eines Dialogsystems steht die Interaktion im Vordergrund. Die genaue Interaktionsreihenfolge wird durch Dialoge beim Softwareherstellungsprozess von den Entwicklern des Systems festgelegt.

Um die Software für einen Multikanaleinsatz vorzubereiten, müssen sowohl die Dialoge als auch der Dialogfluss an die neuen Kanaleigenschaften (in unserem Fall - die Bildschirmgröße) angepasst werden.

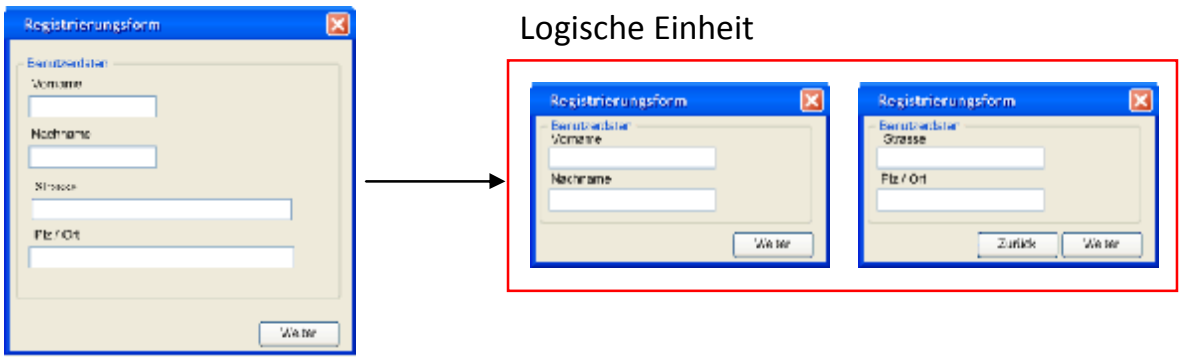

Abbildung 3-2 Beispiel für Erweiterung des Dialogflusses

Ein Verfahren zur dynamischen Anpassung von Dialogen und deren Flusses für ein Webszenario ist in [Book06] beschrieben worden. Kernidee dabei ist, die Informationen, die

die Dialoge darstellen, und die Interaktionsmöglichkeiten, die von den Dialogen angeboten werden, einer Software, an die Bildschirmgröße des Endgerätes zur Laufzeit anzupassen. Das Verfahren überprüft, ob der aktuell vom Benutzer angeforderte Dialog auf dem Endgerät angezeigt werden kann. Falls der Dialog z.B. für den Bildschirm eines mobilen Gerätes zu groß ist, wird der ursprüngliche Dialog in für das Endgerät passende Dialoge aufgeteilt und erst dann an das Endgerät weiter gegeben (s. Abbildung 3-2). Dabei bilden die so neu entstandenen Dialoge eine logische Einheit. Über diese Eigenschaft wird in weiteren Verlauf der Arbeit diskutiert.

Im Vergleich zu der Anpassung der Dialoggröße weist die Erweiterung des Dialogflusses einige Vorteile auf. Die Dialoge bleiben übersichtlich, da nicht alle Informationen des Originaldialoges auf in einem Dialog auf dem Endgerät dargestellt werden. Die Informationsmenge bzw. die Anzahl der Dialogelemente hängt von der Bildschirmgröße ab. Da die Abmessungen der Endgeräte auf den Menschen abgestimmt sind, kann die menschliche Aufnahmefähigkeit optimal ausgenutzt werden, indem die Dialoggröße an die maximale Bildschirmgröße des Endgerätes ausgerichtet wird.

Bevor auf die technische Realisierung der Dialogflusserweiterung näher eingegangen wird, muss die Standard-3-Sichten-Architektir bezüglich der Multikanalfähigkeit von Dialoganwendungen untersucht werden.

Die angestrebte Multikanalfähigkeit von Dialogsoftwaresystemen wirkt sich nicht nur auf die Dialoge und den Dialogfluss des Systems, sondern auch auf die Systemarchitektur aus. Die Standard-3-Schichten-Architektur kann nur sehr begrenzt für multikanalfähige Dialoganwendungen eingesetzt werden.

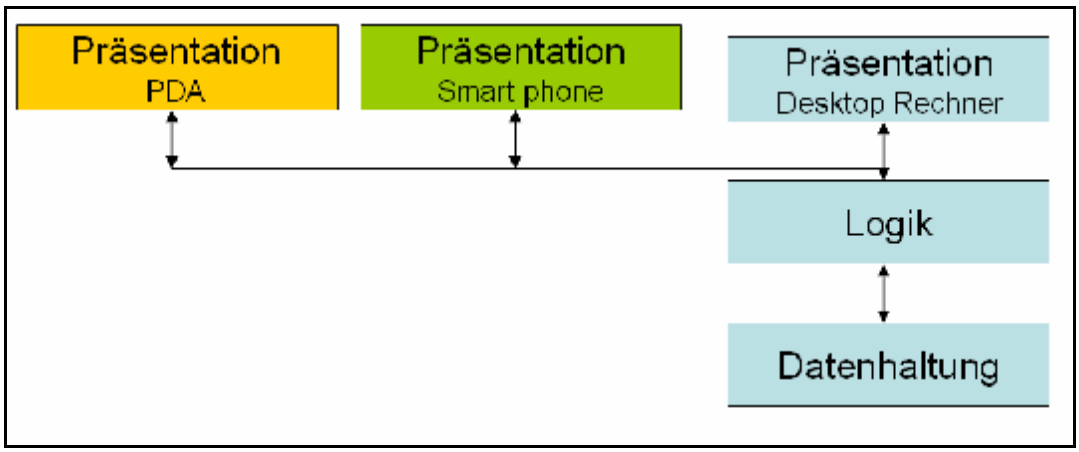

Abbildung 3-3 erweiterte 3-Schichten-Architektur

Bei dem Multikanaleinsatz muss die Präsentationsschicht einer Anwendung für jede Endplattform angepasst. Dies kann aufgrund der Standard-3-Schichten-Architektur realisiert werden, indem für jede Endplattform eine zusätzliche Präsentationsschicht entwickelt wird. Denkbar ist, aus der Schicht für den Desktop die Schichten für PDA und Smartphone abzuleiten, indem z.B. die Dialoge und der Dialogfluss angepasst werden (s. Abbildung 3-3). Eine solche Lösung ist auf der Architekturebene schnell und einfach umzusetzen, dennoch weist diese gravierende Nachteile auf. Pflege, Modifikation und Erweiterungen der Präsentationen sind sehr umständlich. Jede Änderung muss bei allen vorhandenen Präsentationen durchgeführt werden. Weiterhin gewährleistet diese Lösung keine

Unterstützung von zur Entwicklungszeit nicht berücksichtigten oder noch nicht bekannten Endplattformen. Falls die Multikanalfähigkeit des Systems auf ein weiteres Endgerät erweitert werden muss, wird oft eine Entwicklung zusätzlicher Präsentation benötigt. Dies hat zu Folge, dass ein solches Softwaresystem in der heutigen dynamischen Welt nie fertig gestellt werden kann.

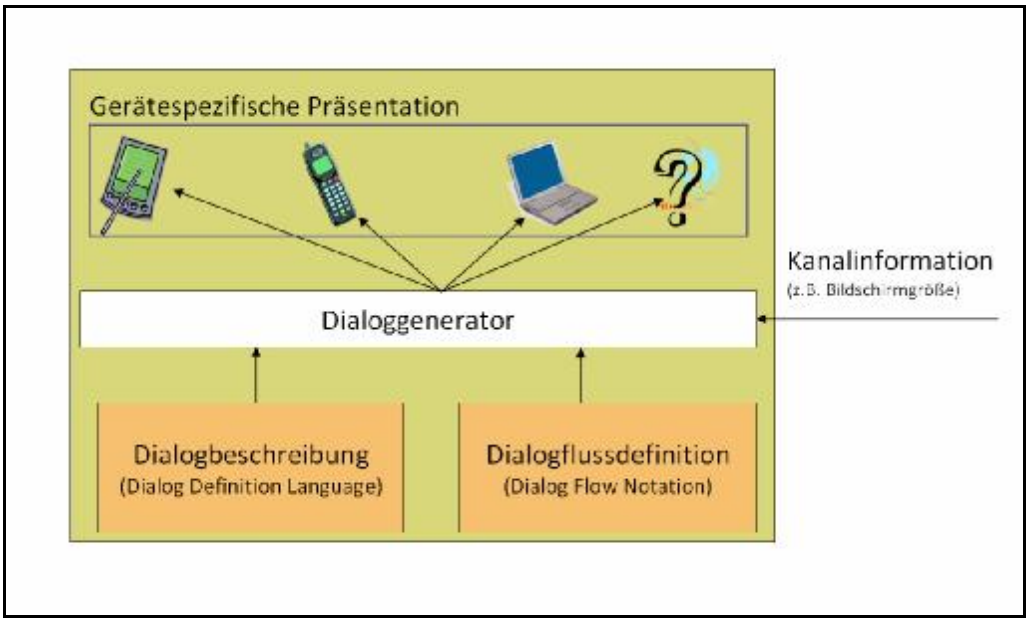

Abbildung 3-4 Erweiterung der Präsentationsschicht im 3-Schichtenmodell

Um die 3-Sichten-Architektur für den Multikanaleinsatz besser geeignet zu gestalten, wird die Präsentationssicht der Standard-3-Sichten-Architektur erweitert (s. Abbildung 3-4). In der Präsentationsschicht werden folgende drei neune Elemente definiert:

- **Dialogbeschreibung:** Geräteunabhängige, semantische und funktionale Beschreibung eines Dialoges. Dialoge werden mittels Dialog Definition Language (DLL) beschrieben und in DDL-Dokumenten gespeichert. Jeder Dialog ist genau in einem DDL-Dokument mit seinen Data- (das Datenmodell des Dialoges) und Interface-Bereichen (die geräteunabhängige Präsentation des Datenmodells) definiert.
- **Dialogflussdefinition:** Definition des Dialogflusses in einer Anwendung mittels Graphen. Diese Definition wird in der Fachliteratur als Dialog Flow Notation (DFN) bezeichnet.
- **Dialoggenerator:** Der Dialoggenerator ist eine aktive Komponente, die aufgrund DDL- , DFN-Dokumenten und der vorhandenen Kanalinformation, die endgültige Präsentation für das Endgerät zur Laufzeit aufbereitet. Dabei bietet der Dialoggenerator die Möglichkeit, das Standardgenerierungsverfahren zu ändern bzw. zu optimieren.

Während des Softwareentwicklungsprozesses werden die DDL-Dokumente von den Softwareentwicklern erstellt. Dabei müssen die Entwickler kein Wissen über das Endgerät, auf dem die Software eingesetzt wird, haben. Die DDL-Dokumente sind wie oben beschrieben geräteunabhängig. Der weitere Schritt ist, den genauen Dialogfluss und somit die Mensch-Maschine-Interaktion zu definieren. Dies wird über die Dialogflussdefinition von den Entwicklern festgelegt. Der Dialogfluss wird mittels Graphen definiert.

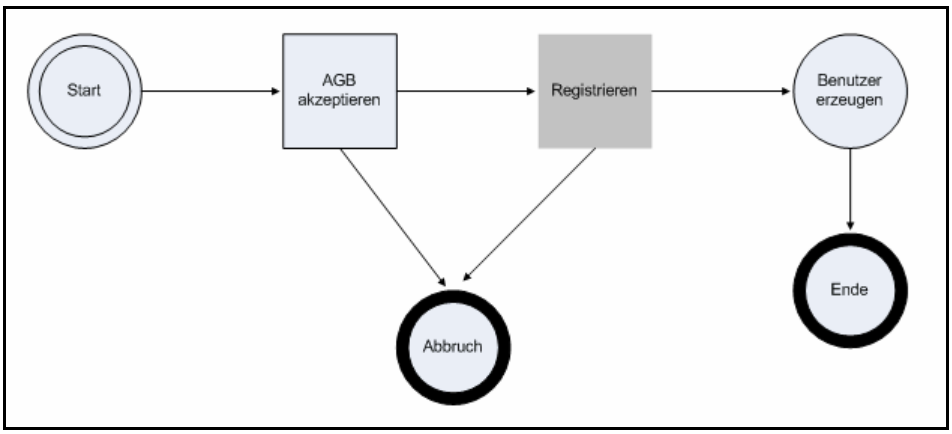

Abbildung 3-5 abstrakter DFN-Graph [Book06]

Bei der Softwareentwicklung werden nur die abstrakten DFN-Graphen (s. Abbildung 3-5) definiert. In diesen Graphen haben die multikanalfähigen Dialoge den Status „abstract“ (in der Abbildung 3-5 der Dialog „Registrieren“) und werden durch einen speziellen Knoten dargestellt. Zur Laufzeit der Software identifiziert der Dialoggenerator das Endgerät und dessen Kanaleigenschaften. Im weiteren Schritt werden aus den abstrakten DFN-Graphen konkrete Graphen generiert, indem die Dialoge mit dem Status „abstract“ aufgrund deren DDL-Beschreibung und der vorhandenen Kanalinformation vom Dialoggenerator konkret für das Endgerät erstellt werden. Dadurch entsteht der konkrete DNF-Graph (s. Abbildung 3-6).

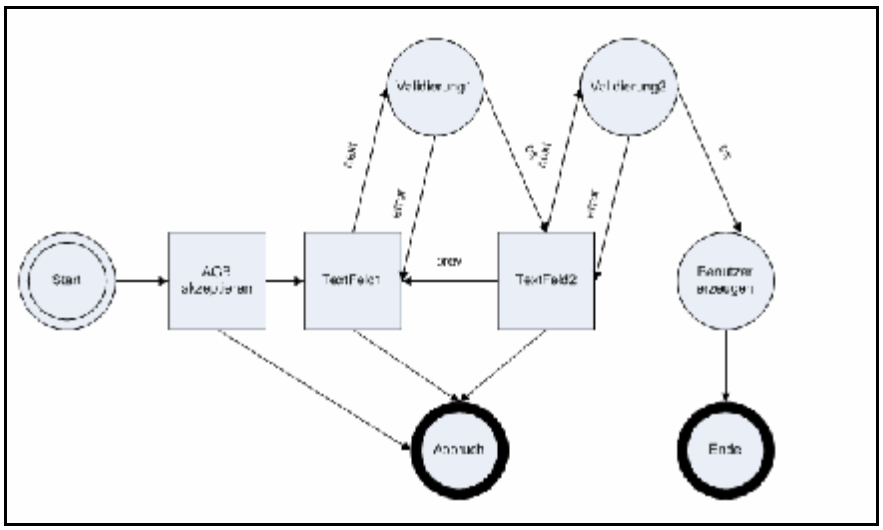

Abbildung 3-6 konkreter DFN-Graph [Book06]

Der konkrete DFN-Graph bildet den tatsächlichen Dialogfluss auf dem Endgerät ab. Dabei wird dieser Graph so erstellt, dass, wenn die früheren abstrakten Dialoge in mehreren Schritten auf dem Endgerät abgearbeitet werden müssen (dies wäre der Fall bei dem Dialog „Registrieren“ in unserem Beispiel), diese durch mehrere Knoten dargestellt werden aber als ein ganzes Teil nach Außen erscheinen. D.h. in dem Dialogfluss ist ein Dialog erst dann abgearbeitet, wenn alle seine Unterdialoge (Knoten eines abstrakten Dialoges) auch abgearbeitet sind, dazu ist wichtig, dass der Benutzer zwischen den Unterdialogen in alle Richtung navigieren kann. Im Grunde bilden die Teile eines abstrakten Dialoges im konkreten Dialogfluss eine logische Einheit (diese Eigenschaft wurde früher in diesem Kapitel erwähnt).

Erste Implementierung nach diesem Verfahren ist unter dem Namen Dialog Generation Framework an der Universität Leipzig als Web-Anwendung mit Java Server Pages realisiert worden. Das Verfahren bieten dem Softwareentwickler die Möglichkeit, von dem Endgerät zu abstrahieren. Der Entwickler ist nicht mehr darauf hingewiesen, mehrere Präsentationsschichten entsprechend für jede bekannte Plattform zu pflegen. Durch das Verfahren können auch „unbekannte“ Endgeräte unterstütz, da das Verfahren die Präsentationssicht einer Anwendung für jede Plattform zur Laufzeit entsprechend deren Kanaleigenschaften generiert. Nebeneffekt ist, dass durch die dynamische Generierung der Dialoge mehrere Ressourcen angefordert werden, das System größere Reaktionszeiten aufweist und die Software nach einer umfangreicheren Architektur erstellt werden muss. Weiterhin gewinnt der Dialogfluss an Komplexität. Dies kann dazu führen, dass bei Systemen mit sehr komplexen Dialogen die Benutzerakzeptanz des System nicht mehr gegeben ist.

3.3.3 Benutzeroptimierte Multikanalfähigkeit

Das im letzten Kapitel vorgestellte Verfahren zur dynamischen Dialoggenerierung weist einen Nachteil auf. Obwohl der Dialoggenerator eine gewisse Dynamik bei der Definition der Dialoggenerierung zulässt, kann diese nicht mehr geändert werden, nachdem die Software einmal fertig gestellt worden ist. Die Parameter der Generierung können nur bei der Softwareentwicklung festgelegt werden. In einigen Situationen kann es dazu kommen, dass das System aufgrund des festgelegten Generierungsverfahrens nicht alle Benutzer oder bestimmte Benutzergruppen zufrieden stellen kann und somit die Akzeptanz des Gesamtsystems nicht gegeben ist.

Um den Benutzer in die Generierung der Präsentation mit einzubeziehen und somit die Akzeptanz des System zu sichern bzw. zu verbessern, ist ein weiteres Verfahren „Community Driven Adaptation (CDA)“ (s. [Chin04]) entwickelt worden. Dieses Verfahren kann als „Personalisierung von Diensten“ basierend auf kollaborativem Filtern und einem Regelwerk (s. [Koychev06]) betrachtet werden. Der Dienst wäre in einer solchen Betrachtung die Darstellung der Dialoge einer Anwendung.

Das CDA-Verfahren ermöglicht dem Benutzer des Systems, die dynamisch generierte Präsentation selbst zur Laufzeit zu optimieren bzw. zu ändern. Der Benutzer ist im Stande, die Dialogelemente neu zu organisieren, oder Dialogelemente bzw. Informationen, die für ihn irrelevant sind, auszublenden bzw. deren Übertragung zu unterdrücken. Solche Änderungen werden vom System verfolgt und bei der nächsten Generierung der Präsentation mitberücksichtigt. Weiterhin teilt das Verfahren die Benutzer eines multikanalfähigen Softwaresystems in Gruppen auf. Die Benutzer einer solchen Gruppe haben ähnliche Anforderungen an die Präsentationsschicht. Es gibt verschiedene Merkmale, nach denen die Gruppen gebildet werden können, eins davon könnte die Bildschirmgröße oder die Hardwareplattform sein. Benutzeralter oder Art der angeforderten Anwendung sind auch als Gruppenmerkmale denkbar, weil jeder Anwendungsfall bestimmte Anforderungen an die Bedienung eines Softwaresystems stellt (denken wir z.B. an das Rescue-Scenario (s. [Hinck06]), bei dem die Interaktion mit dem System auf das Minimum begrenzt werden muss). Im Nachhinein versucht das CDA-Verfahren, die Anforderungen einer Anwendergruppe aufgrund der Benutzerrückmeldungen (Änderungen der Präsentation) zu treffen.

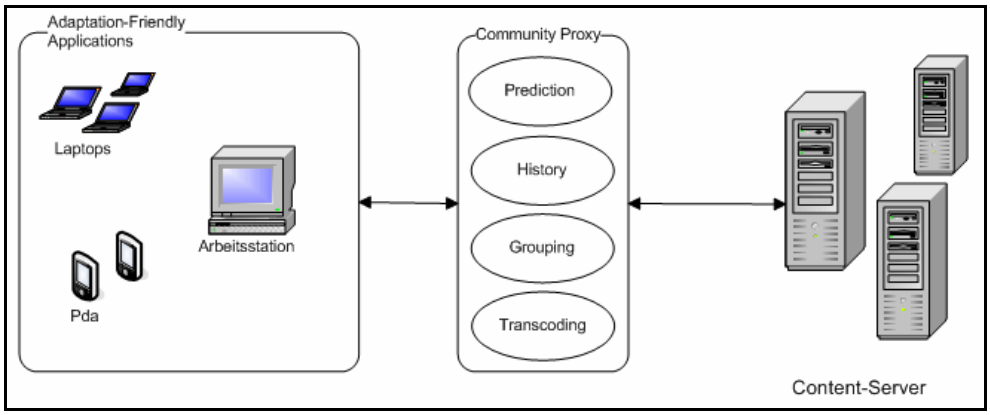

Abbildung 3-7 CDA-Architektur [Chin04]

Das CDA-Verfahren definiert eine entsprechende Architektur (s. Abbildung 3-7). Es sind drei wesentliche Komponenten vorgesehen:

- **Adaptation-Friendly:** Diese Komponente beinhaltet die Applikationen, die der Benutzer in Anspruch nimmt. Wichtig dabei ist, dass die Applikationen in diesem Bereich eine zusätzliche Funktionalität anbieten müssen. Diese Funktionalität soll dem Benutzer ermöglichten, die Präsentation der Applikation selbst zu optimieren bzw. selbst zu ändern.
- **Content-Server:** Dieser Bereich liefert die primäre dynamisch generierte Präsentation (z.B. in diesem Bereich kann das in [Book06] beschriebene Verfahren eingesetzt werden).
- **Community Proxy**: Diese Komponente ist der Mittelpunkt des Verfahrens und wird in vier Unterkomponenten aufgeteilt. Die Grouping-Komponente führt die Aufteilung der Benutzer eines Systems in Gruppen bzw. die Zuordnung von Benutzern zu einer bestimmten Gruppe durch. Die History-Komponente speichert die Benutzerrückmeldungen (Änderungen der Präsentation). Die Transcoding-Komponente fordert von dem Content-Server eine primär aufbereitete Präsentation. Die Prediction-Komponente versucht aufgrund der History-Daten und der Gruppenzugehörigkeit die primär aufbereitete Präsentation, optimal für den entsprechenden Benutzer aufzubereiten.

Die ersten Erfahrungen mit CDA (implementiert für das Web-Scenario) sind in [Chin04] dokumentiert worden. Interessant dabei ist, dass in einer optimierter Version das Verfahren versucht, ausdrücklich nicht immer die beste Präsentation für den jeweiligen Benutzer zu generieren. Das Verfahren generiert in gewissen Abständen Präsentationen mit niedriger Benutzerzufriedenheit. In solchen Fällen reagiert der Benutzer darauf und optimiert die Präsentation nach seinen Vorstellungen. Dadurch wird sichergestellt, dass der Benutzer ein aktiver Teilnehmer im System bleibt. Weiterhin werden die beiden Situationen, die von dem Verfahren selbst nicht unterschieden werden können, ausgeschlossen. Dabei handelt es sich darum, wenn ein Benutzer mit der Präsentation voll zufrieden ist und keine Änderungen vornimmt und wenn ein Benutzer mit der Präsentation so unzufrieden ist, dass er keine Änderungen vornehmen möchte und das System ehe ablehnen würde.

Durch den ständig aktiven Benutzer, der die generierten Präsentationen nach seinen Bedürfnissen gestalten kann, wird die Systemakzeptanz erhöht. Wichtig zu erwähnen ist, dass die Qualität des Verfahrens mit der Anzahl der vorhandenen Benutzer-Feedbacks

zusammenhält. Dennoch kann das Verfahren auch ohne Feedback aufgrund des Regelwerkes (die Transcoding-Komponente) gut funktionieren.

Außer zur Erhöhung der Benutzerzufriedenheit und die Systemakzeptanz kann dieses Verfahren zur Optimierung des Netzwerkdatenvolumens beitragen, indem der Benutzer bei einer langsamen oder teueren Netzverbindung auf die weniger relevanten Inhalte verzichtet. Beispiel dafür wäre eine Dialoganwendung, die Informationen zu Produkten eines Unternehmens in Text und Bild dem Benutzer zu Verfügung stellt und dieser sich nur auf die Textinhalte begrenzt. Dadurch werden die Bilder nicht über das Netzwerk transferiert.

Das CDA-Verfahren bietet eine solide Plattform für multikanalfähige Dialoganwendungen, indem es auf die schon vorhandenen und erprobten Verfahren auf diesem Gebiet aufbaut und den Benutzer als aktiver Teilnehmer einbezieht. Dennoch um das Verfahren sinnvoll einzusetzen, muss eine umfangreiche Infrastruktur aufgebaut werden. Aus diesem Grund sollte dieses Verfahren eher bei größeren verteilten Dialoganwendungen eingesetzt werden.

3.4 Fazit

Multikanalfähigkeit bei Dialoganwendungen kann durch unterschiedliche Verfahren realisiert werden. Jedes Verfahren hat Vor- und Nachteile, die berücksichtigt werden müssen. Die Entscheidung, welches Verfahren am besten geeignet ist, sollte für jeden Anwendungsfall separat getroffen werden.

4 Zusammenfassung und Ausblick

Die Multikanalfähigkeit von Softwaresystemen und insbesondere von Dialogsystemen gewinnt an Bedeutung. Immer mehr Softwareprodukte müssen für mehrere unterschiedliche Plattformen realisiert werden. Dabei ist die Multikanalfähigkeit nicht trivial. Um diese zu gewährleisten, muss gewisser Aufwand in die Softwareentwicklung investiert werden.

Die Multikanalfähigkeit muss weiter ausgebaut werden. Ein wichtiger Punkt bei der Gewährleitung der Multikanalfähigkeit der heutigen Software ist die Softwareergonomie. Die Software soll nicht nur multikanalfähig sondern auch ergonomisch sein. Diese Anforderung wird von den vorgestellten Verfahren nur sehr begrenzt berücksichtigt.

Die in dieser Arbeit vorgestellten Lösungen, gewährleisten die Multikanalfähigkeit in Richtung vom Desktoprechner zu mobilem Gerät. Momentan zeichnet sich der Trend ab, dass viele Softwareprodukte nicht nur auf den mobilen Geräten mit kleinen Bildschirmen zum Einsatz kommen sondern auch auf Plattformen mit großen Interaktionsflächen (wie Powerwalls) eingesetzt werden. Diese Entwicklung wird noch nicht von den vorgestellten Lösungen berücksichtigt. Eine weitere Herausforderung für die multikanalfähigen Anwendungen ist die Collaborative-Workspace-Metapher wegen der großen Anzahl an unterschiedlichen Plattformen.

Zusammengefasst: auf dem Gebiet der Multikanalfähigkeit ist eine gute Basis vorhanden. Dennoch müssen auf diesem Gebiet noch weitere Forschungsarbeiten durchgeführt werden, um alle Anforderungen an die multikanalfähige Software erfüllen und die Trends in der Softwareentwicklung berücksichtigen zu können.

Literaturverzeichnis

[Boll05]
Susanne Boll, Ansgar Scherp,“Paving the Last Mile for Multi-Channel Multimedia Presentation Generation“, 2005, IEEE 1550-5502/05

[Book06]
Matthias Book, Volker Gruhn, „Automatic Dialog Mask Generation for DeviceIndependent Web Applications“, *ICWE’06,* July 2006, ACM 1595933522/06/0007

[Chin04]
Alvin Chin, Iqbal Mohomed, „Community-Driven Adaptation: Automatic Content Adaptation in Pervasive Environments“, 2004, IEEE 1550-6193/04

[Fischer06]
Christian Fischer, „Seminarbericht: Multimodale Interaktionen in Collaborative Workspaces“, SS2006, Haw-Hamburg

[Koychev06]
Milen Koychev, „Ausarbeitung: Personalisieren von Diensten“, SS2006, Haw-Hamburg

[Lemlouma04]
Tayeb Lemlouna, Nabil Layaida, „Context-Aware Adaptation for Mobile Devices“, 2004, IEEE 0-7695-2070-7/04

[openmobilealliance]
Open Mobile Alliance - WAP-Specifikation, http://www.openmobilealliance.org/tech/affiliates/wap/wapindex.html , (Stand 02.2007)

[Pfeifer06]
Tom Pfeifer, Helene Haughney, „UGetMobile End-user Mobile Publishing Platform“, 2006, IEEE 0-7695-2511-3/06

[smartkom]
Projekt “Smartkom” – Intuitive Mensch-Technik-Interaktion, http://www.smartkom.orgl/, (Stand 02.2007)

[w3c.smil]
SMIL Spezifikation, http://www.w3.org/AudioVideo/, (Stand: 02.2007)