Milen Koychev

Multikanalfähigkeit von Dialogsystemen

Ausarbeitung im Rahmen der Seminar- Ringvorlesung
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
Prof. Dr.-Ing. Bernd Schwarz

Abgegeben am 15. Februar 2007

Milen Koychev

Thema der Ausarbeitung

Multikanalfähigkeit von Dialogsystemen

Stichworte

Multikanalfähigkeit, Dialogsysteme, Interoperabilität, Softwareergonomie, Softwareengineering

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 nur für eine Endplattform entwickelt wird, sonder für einen multiplattform- bzw. multikanalfähigen Einsatz konzipiert werden muss. Diese arbeit Stellt einige der Möglichkeiten für solche Optimierungen dar.

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 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 sich 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.

Der Fokus dieser Arbeit ist Analyse der Anwendbarkeit der im Kapitel 2 und in [Koychev07.Aw2] ausführlich vorgestellten Verfahren zur dynamischen Optimierung der Präsentationsschicht von Web-Anwendung in Hinsicht deren Einsatz für lokale Dialoganwendungen[1] auf Hardwareplattformen mit unterschiedlichen Ausgabe bzw. Eingabemöglichkeiten. Die Ergebnisse der Analyse sollen als Basis für eine weitere Forschung im Rahmen der im Sommersemester 2007 auf diesem Gebiet geplanten Masterarbeit dienen. Im Kapitel 3 werden die genauen Ziele der Masterarbeit festgelegt sowie die Ergebnisse der Analyse zusammengefasst und kritisch bewertet.

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.

[1] Als lokale Dialoganwendungen werden in dieser Arbeit Softwaresysteme bezeichnet, bei denen mindestens die Präsentationsschicht des Systems auf der Client-Plattform ausgeführt wird.

2 Grundlagen

In diesem Kapitel werden die technischen Grundlagen zum Thema „Multikanalfähigkeit von Dialogsystemen“ erläutert. Dabei werden die verschiedenen Annsätze bei unterschiedlichen Problemstellungen kurz vorgestellt und bewertet. Eine umfangreichere Dokumentation der in diesem Kapitel vorgestellten Technologien ist in [Koychev07.Aw2] zu finden.

2.1 Was ist Multikanalfähigkeit?

Bevor die technischen Grundlagen 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 Systemteilen 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).

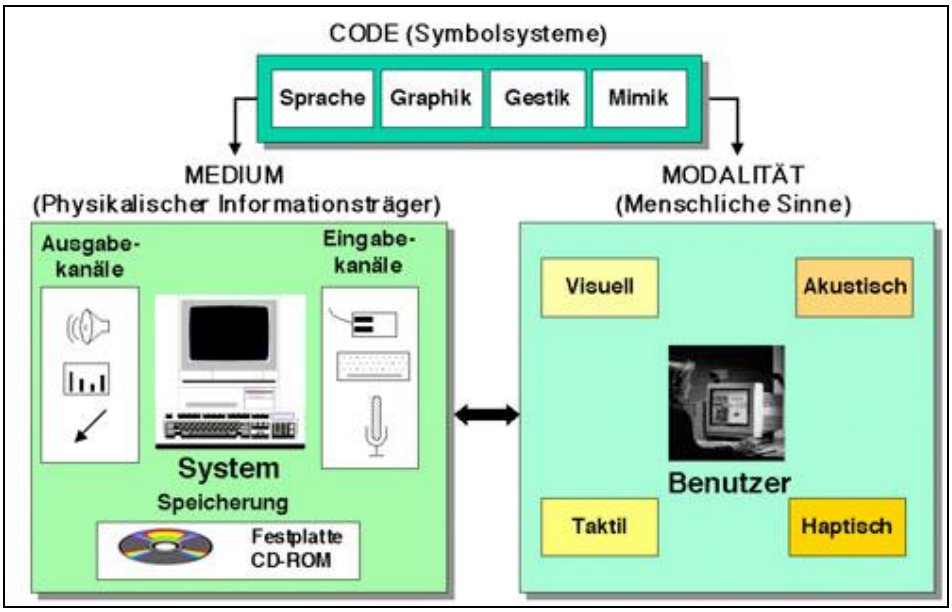

Abbildung 2-1 Smartkom-Modell [smartkom].

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 und Systeme mit Ausgaben auf Powerwalls) und deren Eingabemöglichkeiten betrachtet.

2.2 Erweitern des Dialogflusses

Die Mensch-Maschine-Interaktion wird bei den heutigen Software-Dialogsystemen mittels geeigneten Eingabe- bzw. Ausgabemasken - die so genannten Dialoge durchgeführt. Die genaue Interaktionsreihenfolge wird durch solche 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.

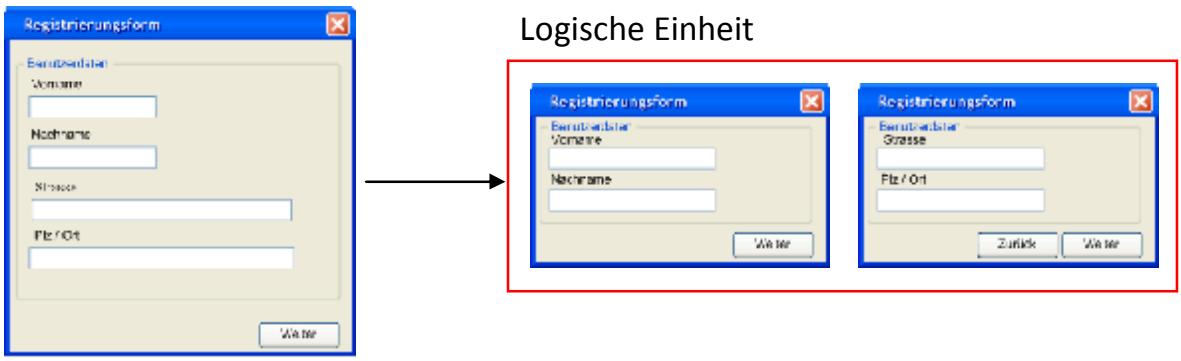

Abbildung 2-2 Beispiel für Erweiterung des Dialogflusses

Ein Verfahren zur dynamischen Anpassung von Dialogen und deren Zusammenspiel für ein Webszenario ist in [Book06] beschrieben worden.

Kernidee bei dem in [Book06] vorgestellten Verfahren 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 2-2). Dabei bilden die so neu entstandenen Dialoge eine logische Einheit. Über diese Eigenschaft wird in weiteren Verlauf der Arbeit diskutiert.

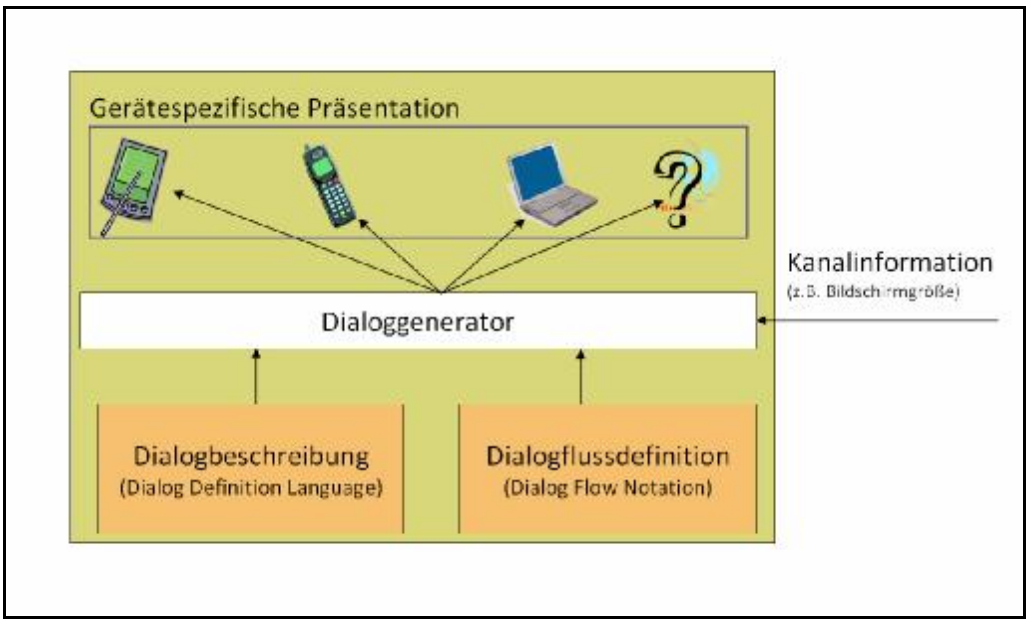

Abbildung 2-3 Erweiterung der Präsentationsschicht im 3-Schichtenmodell

Im Rahmen der dynamischen Dialoggenerierung wird die Präsentationsschicht der 3-Schcihtenarchitektur erweitert (s. Abbildung 2-3). 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.

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.

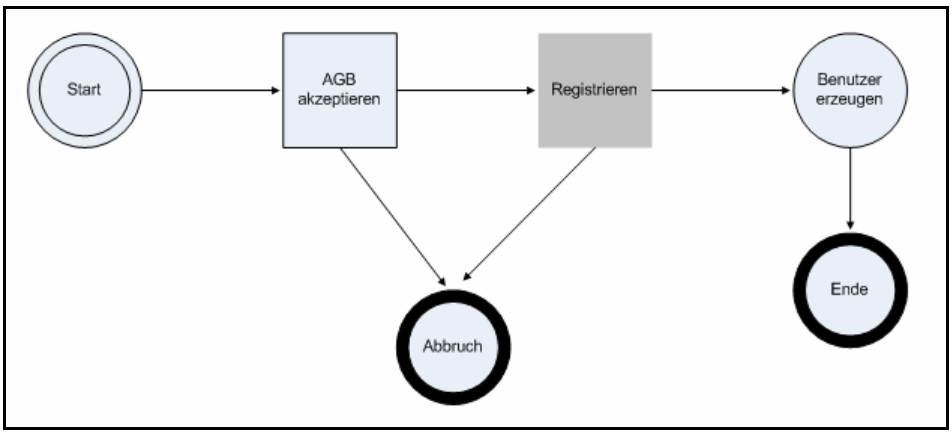

Abbildung 2-4 abstrakter DFN-Graph [Boll06]

Bei der Softwareentwicklung werden nur die abstrakten DFN-Graphen (s. Abbildung 2-4) definiert. In diesen Graphen haben die multikanalfähigen Dialoge den Status „abstract“ (in der Abbildung 2-4 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 konkret für das Endgerät erstellt werden. Dadurch entsteht der konkrete DNF-Graph (s. Abbildung 2-5).

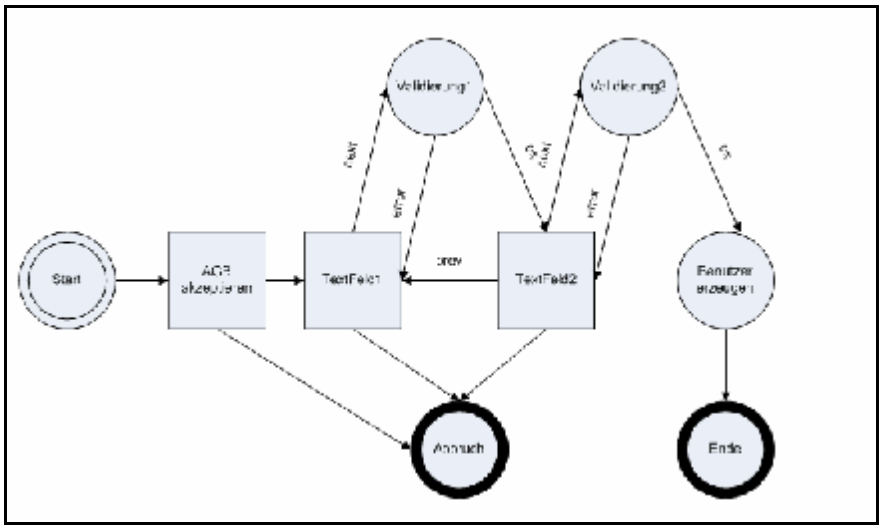

Abbildung 2-5 konkreter DFN-Graph [Boll06]

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).

Das in [Boll06] beschriebene Verfahren, bietet 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.

2.3 Benutzeroptimierte Multikanalfähigkeit

Das im letzten Kapitel vorgestellte Verfahren zur dynamischen Dialoggenerierung weist einen Nachteil auf. Nachdem die Software einmal fertig gestellt wurde, kann die dynamische Dialoggenerierung nicht mehr geändert werden, da die Parameter dieser Generierung nur bei der Softwareentwicklung festgelegt werden können. 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.

Das CDA-Verfahren ermöglicht dem Benutzer des Systems, die dynamisch generierte Präsentation zur Laufzeit zu optimieren bzw. zu ändern. 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.

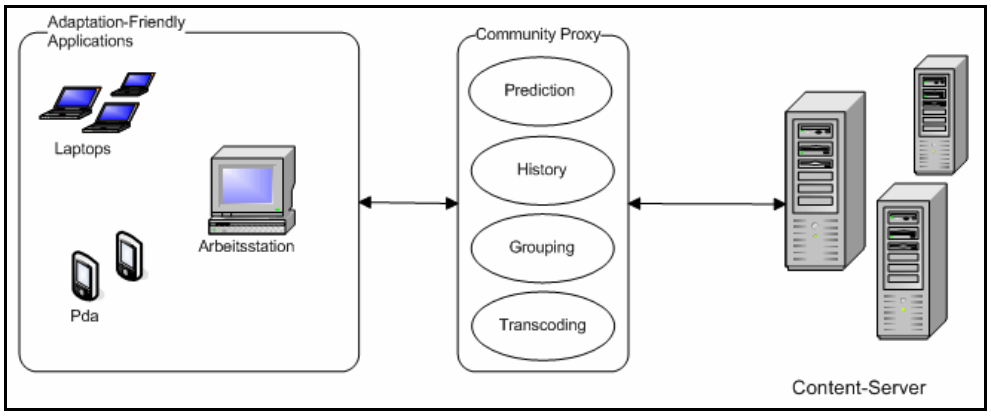

Abbildung 2-6 CDA-Architektur [Chin04]

Das CDA-Verfahren definiert eine entsprechende Architektur (s. Abbildung 2-6). 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 [Boll06] 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.

Durch das CDA-Verfahren werden die herkömmlichen Verfahren zur dynamischen Generierung der Präsentationsschicht (wie z.B. [Boll06]) optimal eingesetzt. Die Benutzerzufriedenheit und somit die Akzeptanz des Systems auf diesem Weg werden verbessert.

2.4 Fazit

Nachdem die technischen Grundlagen in diesem Kapitel beschrieben und erläutert worden sind, ist es offensichtlich, dass die Multikanalfähigkeit von Dialogsystemen nicht trivial ist. Diese ist immer mit einem zusätzlichen Aufwand bei der Softwareentwicklung verbunden. Trotzdem gewinnt die Multikanalfähigkeit in der heutigen Zeit immer mehr an Bedeutung und wird immer öfter von den Benutzern eines Systems gefordert. In den weiteren Kapiteln wird vorgestellt, wie die hier vorgestellten Grundlagen in einem reellen Szenario praktisch eingesetzt werden können.

3 Masterarbeit im Sommersemester 2007

In diesem Kapitel wird das Vorgehen bei der im SS2007 geplanten Masterarbeit auf dem Gebiet der Multikanalfähigkeit von Dialogsystemen vorgestellt. Zuerst wird die Zielsetzung klar definiert. Danach werden die Grundlagen aus dem Kapitel 2 kritisch in Hinsicht auf deren Einsatz bei der Masterarbeit und damit verbundenen Risiken betrachtet. Ein Risikomanagementplan wird aufgestellt, um die erfolgreiche Durchführung der Masterarbeit zu sichern.

3.1 Zielsetzung

Die Zielsetzung der geplanten Masterarbeit ist Entwicklung eines Frameworks[2] für multikanalfähige lokale Dialoganwendungen (z.B. Windows-Dialoganwendungen). Die Entwicklung wird auf den im Kapitel 2 und in [Koychev07.Aw2] vorgestellten Verfahren zur dynamischen Optimierung der Präsentationsschicht einer Anwendung aufbauen. Das Framework soll den Softwareherstellungsprozess optimieren, indem die Präsentationsschicht der Dialoganwendungen nicht mehr gerätespezifisch entwickelt wird, sondern es sichergestellt, dass nach einer Fertigstellung des Softwareprodukts dieses auf möglichst vielen Hardwareplattformen mit unterschiedlichen Bildschirmgrößen einsatzfähig ist. Die gerätespezifische Anpassung der Darstellung, wird vom Framework automatisch zur Laufzeit realisiert. Ein weiterer Punkt, der bei der Konzipierung des Frameworks in Betracht gezogen werden muss, sind die unterschiedlichen Eingabemöglichkeiten der Hardwareplattformen.

.

Neben der Entwicklung des Frameworks werden zusätzlich folgende Punkte untersucht:

- Sichern der Softwareergonomie bei Multikanal-Anwendungen,
- Grenzen der Multikanalfähigkeit von Dialogsystemen.

3.2 Vorgehensweise

In den folgenden Unterkapiteln wird ein Plan aufgestellt, wie die gesetzten Ziele im Rahmen der Masterarbeit erreicht werden sollen. Dabei werden die Risiken und Schwierigkeiten im Vorfeld dokumentiert.

3.2.1 Softwareergonomie

Ein spannendes und wichtiges Thema, das im Rahmen der Masterarbeit betrachtet werden muss, ist die Sicherung der Softwareergonomie. Dies bedeutet, dass das Framework für multikanalfähige Anwendungen nicht nur die Software an die Kanaleigenschaften der Endplattformen anpasst, sondern auch dafür verantwortlich ist, dass die Software auf den unterschiedlichen Plattformen bedienbar ist. Dies ist keine triviale Aufgabe, da jede Plattform unterschiedliche Interaktionsmöglichkeiten anbietet und dadurch der Mensch auf unterschiedlicher Art und Weise die Software wahrnimmt.

Die Softwareergonomie ist durchaus für bestimmte Plattformen (z.B. der alltägliche Desktoprechner mit einem üblichen Bildschirm) sehr gut erforscht und dokumentiert worden.

[2] Ein Framework ist ein Rahmenwerk, der den Kontrollfluss vorgibt [Wolfgang97]

Weiterhin gibt es Plattformen wie z.B. die mobilen Geräte oder Systeme mit Powerwalls als Interaktionsschnittstelle, bei denen die Softwareergonomie noch zu erforschen bzw. zu definieren ist. Die ersten Schritte in diese Richtung sind an der Hochschule für Angewandte Wissenschaften Hamburg gemacht worden – unter der Leitung von Prof. Dr. Jörg Raasch entstehen spezielle Labore zur Untersuchung der Softwareergonomie von mobilen Geräten (s. Abbildung 3-1) und von großen Interaktionsflächen (z.B. Powerwalls).

Abbildung 3-1 Labor zur Untersuchung der Softwareergonomie von mobilen Geräten

Um die Softwareergonomie bei den multikanalfähigen Anwendungen zu sichern, wurde von Prof. Dr. Jörg Raasch ein spezielles Konzept vorgeschlagen. Die Erfahrungen aus den neuen Laboren müssen in die Entwicklung des Frameworks einfließen. Es ist eine formale Beschreibung der Softwareergonomie unter der Berücksichtigung folgender Punkte aufzustellen:

- **Geräte-Klassen:** Für jede Plattformklasse muss ein Softwareergonomie-Profil definiert werden. Dabei ist zu beschrieben, was für Eingabemöglichkeiten (z.B. Maus, Tastatur, Joystick usw.) bzw. Ausgabemöglichkeiten eine Plattformart anbietet und wie diese am besten für den Anwender zu benutzen sind.
- **Anwendung-Szenarien:** Eine weitere sinnvolle Verfeinerung der Ergonomie-Profile ist die Entwicklung von Unterprofilen für unterschiedliche Anwendung-Szenarien. Z.B. für ein Softwaresystem auf einem PDA im Rahmen des Rescue-Szenario können spezielle Anforderungen an die Ergonomie aufgestellt werden. Diese werden sich sicherlich von den „üblichen“ Ergonomieanforderungen an Softwaresysteme für PDA unterscheiden, obwohl es sich um dieselbe Endplattform handelt.
- **Benutzer-Gruppen:** Bei der Erstellung von Ergonomie-Profilen müssen auch die Benutzer und deren unterschiedlichen Anforderungen berücksichtigt werden. Bestimmte Benutzergruppen können ganz unterschiedliche Vorstellungen haben, was eine ergonomische Software ist. Solche Effekte sind bei den mobilen Telefonen aufgetreten – die Geräte sind mit dem Vorschritt der Technik immer kleiner geworden und somit für die älteren Menschen oder Menschen mit Behinderungen nicht bedienbar. Dies muss das Framework verbessert können.

Wie oben beschrieben wird die Softwareergonomie eine wichtige Rolle bei dem multikanalfähigen Ansatz spielen. Aus diesem Grund müssen die in Kapitel 2 vorgestellten Verfahren modifiziert werden, damit auch die Softwareergonomie in Form der formalen Definition (Ergonomie-Profile) berücksichtigt werden kann. Dafür existieren folgende zwei Möglichkeiten:

- **Modifikation der erweiterten Präsentationssicht:** Die Präsentationsschicht (s. Abbildung 2-3) wird erweitert, indem der Dialoggenerator nicht nur „einfache“ Kanalinformation bei der dynamischen Erstellung von Dialogen berücksichtigt, sondern auch ein für die entsprechenden Endplattform definiertes Ergonomie-Profil. Somit hat der Dialoggenerator die Möglichkeit, nicht nur die Dialoggröße anzupassen, sondern auch für das Endplattform passende Bedienelemente zu wählen, um die beste Bedienbarkeit der Software anzubieten. Ein Beispiel dafür wäre, auf einem Smartphone, das hauptsächlich durch einen Joystick von dem Benutzer bedient wird, statt Listbox-en die Information durch Checkbox-en oder Radiobutton-s zu visualisieren.
- **Erweiterung des CDA-Verfahren:** Das CDA-Verfahren bietet eine sehr passende Schnittstelle, um die Ergonomie-Profile anzubinden. Das Verfahren kann so erweitert werden, dass die Benutzergruppen aufgrund der Ergonomieanforderungen gebildet werden. Somit werden die drei Punkte (Geräte-Klassen, Anwendung-Szenarien und Benutzer-Gruppen) abgedeckt.

Da die Multikanalfähigkeit ein relativ neues Forschungsthema ist, das gerade aktuell wird, müssen die schon vorhandenen Verfahren wie oben beschrieben modifiziert werden, um auch die Softwareergonomie bei den multikanalfähigen Anwendungen zu sichern.

3.2.2 Grenzen der Multikanalfähigkeit

Während der Masterarbeit werden die Grenzen der Multikanalfähigkeit erforscht werden. Die Fragen, ob eine Anwendung auf herkömmlichen Desktop-Rechnern, auf PDAs, auf Smartphones und auf Powerwalls sinnvoll eingesetzt werden kann, und mit welchem technischen Aufwand dies verbunden ist, müssen beantwortet werden.

Erste Erfahrungen auf diesem Gebiet sind schon vorhanden. Die Verfahren zur dynamischen Erstellung von Dialogen, betrachten nur den Fall, wenn eine Software vom Desktoprechner zu einer mobilen (kleineren) Plattform migriert. Sicherlich ist die Fragestellung interessant, was zu tun ist, wenn eine Software vom Desktop oder von einer mobilen Plattform auf einen Powerwall migriert. Die ersten Überlegungen und Diskussionen schlagen den Einsatz der Simulation vor. Dies bedeutet, dass der Benutzer auf den großen Interaktionsflächen mit der Software in Originalgröße auf einem originalgroßen virtuellen Bildschirm agiert. Einige Testversuche, wie der Benutzer mit großen Interaktionsflächen agiert, sind im Rahmen des Masterprojektes im WS0607 an der HAW-Hamburg durchgeführt worden. Einige der Ergebnisse haben gezeigt, dass der Benutzer eine für den Desktoprechner entwickelte Software nur sehr schwer auf einer großen Interaktionsfläche bedienen kann. Aus diesem Grund sind weitere tiefer gehende Untersuchungen notwendig.

Ein weiterer Faktor, der bei den Grenzen der Multikanalfähigkeit betrachtet werden soll, sind die heutzutage vorhandenen Entwicklungsbibliotheken und Technologien (z.B. Java oder .NET). Um Multikanalfähigkeit zu gewährleisten, müssen die Softwaresysteme oder bestimmte Systemteile (bei einem verteilten Szenario) für unterschiedliche Betriebssysteme entwickelt werden. Viele Hersteller (auch Sun und Microsoft) werben mit der Interoperabilität deren Plattformen (Java bzw. .NET). Nach ersten Erfahrungen im Masterprojekt, ist eine Interoperabilität bei den heutigen Softwaretechnologien nicht hundert Prozent gegeben – viele Funktionen, die auf einem Betriebsystem angeboten werden, werden von derselben

Technologie (z.B. .NET, Windows XP vs. Windows mobile 2005) auf einem anderen Betriebssystem nicht mehr unterstützt.

3.2.3 Multikanalfähigkeit von lokalen Dialoganwendungen

Da die im Kapitel 2 beschriebenen Verfahren für das Web-Szenario entwickelt worden sind, müssen diese für lokale Dialoganwendungen angepasst werden. Dabei müssen die Unterschiede zwischen den beiden Ansätzen berücksichtigt werden.

Bei dem Web-Szenario werden die Dialoge zentral im Backend-Bereich (s. Abbildung 3-2) als HTML-Formulare gehalten. Die Anpassung von Dialogen findet auch zentral auf dem Proxy statt. Es ist keine zusätzliche Logik auf der Client-Seite notwendig – das Verfahren ist „transparent“ für den Benutzer. Die Kanaleigenschaften der Endplattform werden innerhalb des HTTP-Request-s ermittelt.

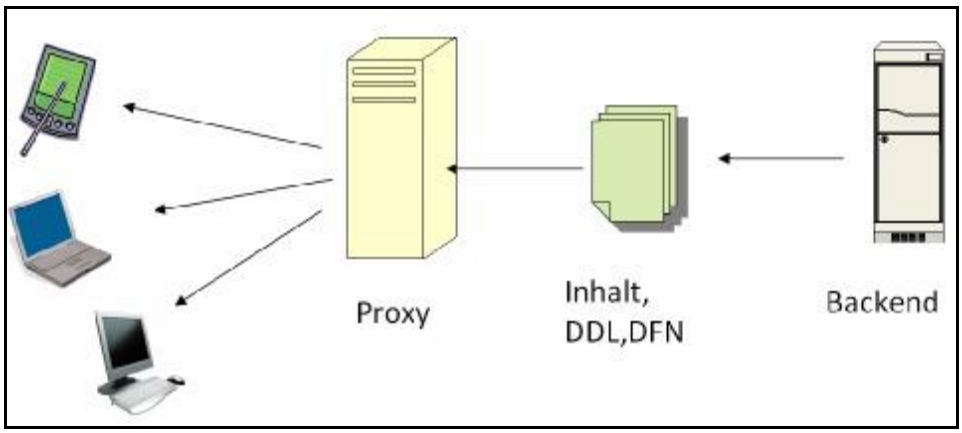

Abbildung 3-2 Systemarchitektur Web-Szenario

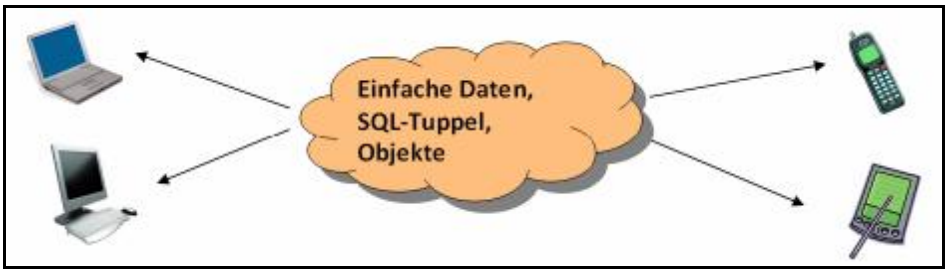

Abbildung 3-3 Systemarchitektur für ein nicht Web-Szenario

Im Gegensatz zu den Web-Anwendungen sind die Dialoge bei den lokalen Dialoganwendungen nicht zentral in dem Backend-Bereich, sondern auf den Clients in Form von Ressource-Dateien vorhanden. Im Backend-Bereich werden einfache Daten, SQL-Tuppel oder Objekte (s. Abbildung 3-3) gehalten. Diese Daten sind für die Multikanalfähigkeit irrelevant. Aus diesem Grund kann die Anpassung der Präsentationsschicht nicht zentral durchgeführt werden. Diese muss auf unterschiedlichen Clients stattfinden. Dies fordert eine hundertprozentige Interoperabilität des Frameworks, da die Clients unterschiedliche Betriebsysteme haben können. Weiterhin müssen spezielle Mechanismen für die

Kanalidentifikation sowie die Modifizierung der Dialoge auf der Clientseite entwickelt werden.

Die Befürworter des Web-Szenarios werden die Entwicklung eines solchen Frameworks für lokale Dialoganwendungen in Frage stellen, da die Web2.0-Methapher die lokalen Anwendungen ablösen soll. Trotzdem ist ein solches Framework notwendig, da die lokalen Anwendungen für bestimmte Aufgaben aufgrund lokaler Rechte auf den Endplattformen unersetzbar sind. Weiterhin werden solche Anwendungen im Intranetbereich von den großen Unternehmen überwiegend bevorzugt.

3.3 Risiken

Zu jedem Softwareentwicklungsprojekt gehört auch eine ausführliche Risikoanalyse. Diese wird in diesem Kapitel aufgestellt, indem die schon erkannten und in den früheren Kapiteln erwähnten sowie noch nicht dokumentierten Risiken der Masterarbeit aufgezählt und bewertet werden.

Softwareergonomielabore: Wie im Kapitel 3.2.1 beschrieben, sind im Rahmen der Masterarbeit Untersuchungen mittels der neuen Softwareergonomielaboren (für mobile Geräte und für Powerwalls) geplant. Da diese Labore, besonders das Labor für mobile Anwendungen noch in Entwicklung sind, sind die geplanten Untersuchungen gefährdet, falls die Labore bis Sommersemester 2007 nicht fertig gestellt werden. Wenn dies der Fall wäre, könnten die Untersuchungen mit dem Prototyp des mobilen Labors (s. Abbildung 3-1) durchgeführt werden. Dabei muss sichergestellt werden, dass die Ergebnisse aussagekräftig sind, da diese sich auf den weiteren Verlauf der Masterarbeit auswirken werden.

Benutzerakzeptanz: Eine Gefahr ist, dass trotz der gut funktionierenden Verfahren zur dynamischen Anpassung der Präsentation einer Software, das Ergebnis für den Benutzer nicht akzeptabel ist. Das wäre der Fall, wenn z.B. große Dialoge in viele kleine Unterdialoge aufgeteilt werden, und somit der Benutzer nicht mehr dem Fluss der Anwendung folgen kann. Um dagegen zu steuern, müssen während der Ergonomieuntersuchungen und der Erkundung der Grenzen von Multikanalfähigkeit Richtlinien entwickelt werden. Diese sollen dokumentieren, in wieweit und wie genau sich große Dialoge in kleine aufteilen lassen. Es muss weiter geforscht werden, um für Dialoge, die sich nicht in Unterdialoge aufteilen lassen, eine akzeptable multikanalfähige Darstellung zu finden.

Interoperabilität: Die gewünschte hundertprozentige Interoperabilität der Softwareplattformen, wie oben besprochen, ist momentan nicht gegeben. Die Softwarehersteller arbeiten an diesem Thema ständig weiter. Bis zum Sommersemester 2007 könnte sich die Interoperabilität verbessern, jedoch eine vollständige ist noch nicht in Sicht. Aus diesem Grund sollen bei der Entwicklung des Frameworks für multikanalfähige Anwendungen nur solche Funktionalitäten benutzt werden, die auch interoperabel sind. Dies wird die Entwicklung erschweren und verlangsamen.

Neues Thema: Das Thema Multikanalfähigkeit ist relativ neu, da bis vor kurzem die Voraussetzungen dafür (mobile Geräte sowie große interaktive Flächen) nicht für das breite Publikum zugänglich waren. Wie bei jedem neuen Gebiet sind noch viele Fragen offen und es fehlt an Informationen. Dieses Risiko wird durch intensive Recherche, Besuch von Konferenzen zu diesem Thema, sowie eigene Forschungsarbeit bewältigt.

Umfangreiche Forschungsarbeit: Wie oben erwähnt muss im Rahmen der Masterarbeit eine umfangreiche Forschung betrieben werden. Dies hat als Folge, dass die Zeit für solche Forschungsarbeiten vorgesehen und einkalkuliert werden muss. Einige Forschungsarbeiten sind noch zwingend zu leisten, um den Erfolg der Masterarbeit zu sichern (s. Punkt Benutzerakzeptant).

3.4 Fazit

Das Thema Multikanalfähigkeit gewinnt in der heutigen Zeit immer mehr an Bedeutung, die ersten Forschungsarbeiten auf diesem Gebiet sind schon durchgeführt worden. Weiterhin bietet das Thema noch viele Herausforderungen und offene Fragen. Im Rahmen der geplanten Masterarbeit im Sommersemester 2007 an der Hochschule für Angewandte Wissenschaften Hamburg besteht die Möglichkeit, basieren auf den schon vorhandenen Ergebnissen auf einige der offenen Fragen Antwort zu geben sowie neue Fragen und Aufgaben im Bereich Multikanalfähigkeit zu definieren und somit zu der Erforschung dieses Gebietes einen eigenen Beitrag zu leisten.

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

[Hinck06]
Steffen Hinck, „Einsatz von Wearable Computing in Desasterszenarien“, WS06-07, Haw-Hamburg

[Koychev07.Aw2]
Milen Koychev, AW2, 2007, Vortrag und Ausarbeitung zum Thema „Multikanalfähigkeit von Dialogsystemen“

[Piening06]
Andreas Piening, „ Leitstand für Desaster-Szenarien “, WS06-07, Haw-Hamburg

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

[Wolfgang97]
Wolfgang Pree, „Komponentenbasierte Softwareentwicklung mit Frameworks“, 1997, Heidelberg : Dpunkt-Verl., ISBN 3-920993-68-3