University of Siegen

Diploma Thesis

iManual -

Context-Aware Mobile Help Systems

by

Daniel Humberg

March 2004

Universität Siegen

Diplomarbeit

iManual -

Kontextsensitive mobile Hilfesysteme

von

Daniel Humberg

Fachbereich 5

Wirtschaftswissenschaften

Lehrstuhl für Wirtschaftsinformatik

insbesondere Kooperations- und Mediensysteme

Betreuer: Prof. Dr. Volker Wulf (Universität Siegen,

Fraunhofer Institut für Angewandte Informationstechnik FIT)

Dipl.-Inf. Markus Klann (Fraunhofer Institut für Angewandte Informationstechnik FIT)

Zweitprüfer: Prof. Dr. Manfred Grauer (Universität Siegen)

I have always wished that my computer would be as easy to use as my telephone. My wish has come true. I no longer know how to use my telephone.

– Bjarne Stroustrop (originator of C++) [from Yates et al. (2003)]

Abstract (English)

This work will present the iManual concept of using PDAs as context-sensitive and interactive help systems for sufficiently complex devices that was developed along with this diploma thesis. The concept was prototypically realized for using the navigation system of a BMW 7, but is also transferable to using coffee machines, washing machines, photocopiers, industrial machines etc.

Using handheld computers such as PDAs leads to several advantages compared to conventional paper-based manuals. These advantages base on the following characteristics of a PDA: mobility, connectivity, interactivity, and personality as well as the resulting adaptivity and adaptability of the help system running on the PDA.

Mobility and connectivity are prerequisites for a specific situation of using the help system: The user can use the PDA immediately when operating the device, e.g. in his car, in the kitchen or on the way, and the PDA is connected both to the device (e.g. via Bluetooth™) and to external information sources such as the manufacturer's product database or a user community (e.g. via GPRS or UMTS). This results in a close situative integration of using and learning processes: First, the PDA can request the internal state from the device and send commands to it and second, up-to-date information about the device and annotations and tips from other users can be transferred to the PDA. Furthermore, information about current user interaction, e.g. user problems, ratings or solutions to common problems, can be send to the manufacturer or the user community. Based on this, context-aware mobile help systems can enable transferring many of the successful features of modern help systems for software applications to help systems for any physical devices.

Interactivity, i.e. the opportunity to not only get information about the device on the PDA, but to remote control it from the PDA, can by used to achieve an an even closer integration of using and learning. Altogether, these kinds of help systems enable the projection of the user interfaces of all kinds of devices onto the familiar, personalized interface of the PDA and therefore lead to a more efficient usage.

Analyzing the internal state and the interaction history of the device, the attributes of the user (e.g. previous knowledge, level of expertise) as well as other contextual parameters of the environment allows to provide help information and control functionality that is

tailored to the current situation and the user's needs. This adaptivity is based on processing contextual and individual information on the PDA or a central server and the resulting selection and presentation of appropriate information and functionality. The resulting delivery of adaptive and individualized help can even be enhanced by providing adaptability, i.e. letting the user actively tailor the system to his needs.

This diploma thesis will discuss these issues in detail and introduce the iManual system, which is a prototypical realization of the concept of a context-aware mobile help system. This includes presenting the system architecture and describing how adaptive, context-aware help was implemented using a Case-Based Reasoning engine that is based on a CBR package developed at Fraunhofer FIT.

Abstract (German)

In dieser Arbeit wird das in Zuge in meiner Diplomarbeit entwickelte iManual-Konzept der Verwendung von PDAs als kontextsensitive und interaktive Hilfesysteme bei der Benutzung von Geräten hinlänglicher Bedienkomplexität vorgestellt. Dieses Konzept wurde prototypisch am Beispiel der Bedienung des Navigationssystems im 7er BMW implementiert, ist aber gleichermaßen auf die Bedienung von Kaffeeautomaten, Waschmaschinen, Kopierapparaten oder industriellen Maschinen etc. anwendbar.

Durch den Einsatz von PDAs können eine Reihe von Benutzungsvorteilen im Vergleich zu konventionellen papiergebundenen Handbüchern erzielt werden. Diese Vorteile beruhen auf den folgenden Eigenschaften von PDAs: Mobilität, Konnektivität, Interaktivität und Personalität sowie der daraus resultierenden Adaptivität und Anpassbarkeit des darauf ausgeführten Systems.

Mobilität und Konnektivität sind dabei die Voraussetzungen für eine spezifische Nutzungssituation: Der Benutzer kann seinen PDA unmittelbar bei der Bedienung des Gerätes verwenden, also z.B. in seinem Fahrzeug, in der Küche oder unterwegs, und sein PDA ist einerseits mit dem zu bedienenden Gerät (z.B. über Bluetooth™) und andererseits mit externen Informationsquellen wie der Produktdatenbank des Herstellers oder einer Nutzer-Community für dieses Produkt verbunden (z.B. über GPRS oder UMTS). Auf diese Weise kann eine enge situative Verzahnung von Bedien- und Lernvorgängen erzielt werden: Der PDA kann den Status des zu bedienenden Gerätes abfragen und an dieses Kommandos übermitteln, und es können aktuelle Produkt- und Bedieninformationen, Tipps von anderen Benutzern etc. an den PDA übertragen werden. Weiterhin können Informationen über die aktuelle Benutzung, wie z.B. Benutzungsschwierigkeiten, Bewertungen oder Lösungen von typischen Bedienungsproblemen, an den Hersteller oder andere Benutzer übermittelt werden. Durch vernetzte PDAs ergibt sich somit die Möglichkeit, die inzwischen für die meisten Software-Systeme verfügbaren leistungsfähigen Hilfesysteme für die Benutzung von allen Arten von Gerätschaften nutzbar zu machen.

Durch die Interaktivität, d.h. die Möglichkeit über den PDA nicht nur Informationen zur Bedienung zu erhalten, sondern das jeweilige Gerät direkt zu bedienen, kann eine noch weitergehende Integration von Bedienung und Unterstützung erzielt werden. Insgesamt

kann dieses Szenario dazu führen, dass die Benutzeroberflächen verschiedenster Gerätes auf die vertraute, personalisierte Oberfläche des PDA projiziert werden und so eine einfachere Bedienung ermöglichen.

Durch die Auswertung des Zustands und der Bedienungsgeschichte des Gerätes, sowie weiterer kontextueller Merkmale der Umgebung und der Eigenschaften des Benutzers (z.B. Vorwissen, Bedienungskompetenz) lassen sich Bedieninformationen und -funktionen bereitstellen, die speziell auf die aktuelle Nutzungssituation und die Bedürfnisse des Benutzers abgestimmt sind. Diese Adaptivität des Hilfesystems beruht auf der Verarbeitung der Kontext- und Benutzerinformationen auf dem PDA oder einem zentralen Server und der entsprechenden Auswahl und Darstellung geeigneter Informationen und Funktionen. Die dadurch erreichte individuelle Benutzerunterstützung kann durch umfangreiche Anpassungsmöglichkeiten noch verbessert werden.

Die Diplomarbeit wird diese Gesichtspunkte genauer diskutieren und die Umsetzung des Konzepts eines kontextsensitiven mobilen Hilfesystems durch das prototypische iManual-System vorstellen. Dabei werden insbesondere die entwickelte System-Architektur und die technischen Umsetzung des adaptiven Hilfesystems durch ein Case-Based-Reasoning Modul vorgestellt, das auf eine von Fraunhofer FIT entwickelte CBR-Klassenbibliothek aufsetzt.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ALE	Adaptive Learning Environment
API	Application Programming Interface
CBR	Case-Based Reasoning
CMS	Content-Management System
CRM	Customer Relationship Management
CSS	Cascading Style Sheet
EBL	Explanation-Based Learning
Fraunhofer FIT	Fraunhofer Institute for Applied Information Technology FIT
eVC	Embedded Visual C++
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
GUI	Graphical User Interface
HCI	Human-Computer Interaction
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ICE	In-Car Entertainment
IDE	Integrated Development Environment
IrDA	Infrared Data Association
IT	Information Technology
ITS	Intelligent Tutoring System
IUI	Inductive User Interface
JVM	Java™ Virtual Machine
kb/s	Kilobits per Second
LAN	Local Area Network

MFC	Microsoft Foundation Classes
OEM	Original Equipment Manufacturer
PDA	Personal Digital Assistant
PDF	Portable Document Format
RBR	Rule-Based Reasoning
RFID	Radio Frequency Identification
SMS	Short Message Service
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
UI	User Interface
UMTS	Universal Mobile Telecommunication System
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VRM	Vehicle Relationship Management
WAP	Wireless Application Protocol
WLAN	Wireless LAN
XML	eXtensible Markup Language

1 Introduction

1.1 Thesis Statement

Transferring help systems for devices to mobile handheld computers holds a great potential. Main advantages of handhelds are mobility, connectivity, interactivity and personality. These advantages can be exploited to create better help systems for devices. One such way, which is shown is this work, is to obtain contextual information such as the recent interaction with the device and the help system, the device's internal state, and user characteristics (e.g. user experiences, level of expertise, implicitly or explicitly stated preferences etc.) to give assistance according to that information. When help systems for software moved from paper to electronic media, similar strategies, among others, were successfully used to improve the help system's usability. The system prototype that was build along with this thesis is planned to be used for usability studies to evaluate if similar improvements can be achieved.

A help system that is based on mobile handheld computers creates a completely new learning situation. Considering the main advantages mentioned above, the user can take his mobile help system directly to the device, let it connect to the device, and interact with both help system and device through the handheld's familiar interface, while getting situated help when problems occur. So all in all, learning about a device and using it will be integrated.

Along with this thesis, the concept of a context-aware mobile help system was realized by developing the iManual prototype. The iManual help system is implemented for the navigation system of a BMW 7. However, the concept can be realized for many kinds of devices and the iManual system can easily be transferred to other domains. This thesis generally talks about help systems for *devices* but this term represents a wide range of possibilities. A device in this context can be any entity that could need further explanation in certain situations. This includes consumer electronics such as VCRs or digital cameras, office equipment such as laser printers or a copy machines, powered tools such as drill or grinding machines, and large-scale industrial machines as well as small-scale products of our daily life such as RFID-tagged clothes that tell us how to wash them or tagged food items that give cooking instructions.

1.2 Remaining Thesis Structure

This thesis discusses the concept of context-aware mobile help system and presents a prototypical realization.

Section 2 of this thesis analyzes the state of the art in research fields that are relevent for this concept. The first subsection about context-aware computing introduces the terminology that is used in this thesis, provides a classification of context dimensions and identifies the dimensions relevant for context-aware mobile help systems. Furthermore, it provides a classification of context-aware computing systems and shows that a context-aware mobile help system is different from most existing context-aware computing systems. The next subsection about intelligent help provides a classification of help systems and evaluates different strategies to improve help. It is shown that several strategies that are mostly used for software systems have a positive influence on the quality of the given help. To provide the psychological background for this evaluation, the questions why and how adults learn are addressed. The subsection starts with answering the question if help systems are needed at all. The third subsection about mobile Human-Computer Interaction analyzes the chances and challenges that providing help on a handheld computer brings. The challenges are addressed by examining empirically founded guidelines for designing user interfaces and for writing text on handheld devices. The fourth subsection justifies why a context-aware help system is particularly promising for the automotive domain – the domain used for the prototype.

Section 3 introduces the concept of context-aware mobile help systems. First, it explains in which ways contextual information can be exploited to provide context-aware help. This includes describing integrated help that exploits contextual information related to the device, personalized help that adapts assistance to the individual user, and location-based help that provides help according to the geographic or relative position. The next subsection describes how machine learning strategies can be used to provide intelligent help. It is explained why Case-Based Reasoning is a particularly promising strategy, which is underlined by giving examples for systems that successfully use case-based strategies. The third subsection shows that help systems that can remote control the device over wireless links lead to a new learning situation where learning and using are integrated. The fourth subsection analyzes how content has to be organized for context-aware mobile help systems. Finally, chances and challenges of the concept are summarized.

Section 4 describes the iManual system, a prototypical implementation of a context-aware mobile help system, that was developed along with this thesis. This section justifies the chosen domain of the prototype, summarizes the most important features of the system and gives a functional overview of the system architecture, which consists of three logical parts: the iManual PDA, the iManual server and an iManual-compatible device. Subsection 4.4 gives design and implementation details. This includes a brief description of the used platforms and programming languages, explains the communication over web services and Bluetooth™, describes how the Fraunhofer FIT CBR engine was used to implement context-aware adaption of the help system, and shows how the Fraunhofer FIT Adaptive Learning Environment was integrated to handle the help content. Finally, further work that is needed to improve and extend the iManual system is listed.

Section 5 summarizes how the opportunities of the concept were realized and the challenges were met in the iManual system and discusses if the expectations that were expressed prior to developing the system were fulfilled.

2 State of the Art

A context-aware mobile help system is a hybrid concept that touches upon several research fields. This section describes the theoretical background, recent findings, and existing applications in these fields.

It is a complex problem situation where a user operates a physical device and needs context-aware help provided by a help system on a handheld computer. Users form a very heterogeneous group, i.e. they differ in their knowledge level, attitude towards learning, experience with the device and various other aspects. Every user is in a unique situation that is characterized by his current activity, his previous interactions with the device, his location, and his environment. This makes applying context-aware computing to a mobile help system both promising and challenging. Therefore, the next chapter analyzes relevant research in context-aware computing and how theoretical and empirical findings can be related to the usage of context-aware computing in mobile help systems.

2.1 Context-Aware Computing

One reason why many people find existing computing systems hard to use, even if their user interfaces (UIs) are carefully designed, lies in a major difference in how humans communicate with other humans and with machines. In human-to-human communication we are able to use implicit information that is specific to the current situation. This context-related information is mostly either neglected or impossible to obtain by computing systems in Human-Computer Interaction (HCI) [see Dey et al. (1999)].

Making more context-information available to computing systems holds a great potential in improving their usability, making them less obtrusive, and therefore making them *calm* [Weiser et al. (1995)] or even *invisible* [Norman (1998)].

2.1.1 Terminology

Currently, many researchers and software developers are using terms like context-awareness, context-sensitivity, situation-awareness etc. to describe their research activities or software developments. In this chapter, the most dominant definitions are given and compared.

2.1.1.1 Context, Sensed Context and Context Fusion

When researchers began to realize that considering *context* is important to improve the usability of computing systems, they found that the existing linguistic definitions [e.g. Schulenburg (1995)] of context were not sufficient for their domain. Several works provided different definitions while Goodwin et al. (1992) even stated that "it doesn't seem possible at the present time to give a single, precise, technical definition of context, and eventually [...] such a definition may not be possible." Nevertheless, a general definition is given by Dey:

> *Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.* [Dey et al. (1999)]

A more specific definition that better fits the system proposed in this thesis states that:

> *context is the constantly changing execution environment including the following pieces of the environment:*
>
> - ***Computing environment*** *available processors, devices accessible for user input and display, network capacity, connectivity, and cost of computing*
> - ***User environment*** *location, collection of nearby people, and social situation*
> - ***Physical environment*** *lightning and noise level*
>
> [Schilit et al. (1994b)]

Chen et al. (2000) adds the "*time context*" to Schilit's three pieces of environment and stresses that new possibilities arise when considering the "*context history*" by recording context aspects across a time span.

However, not all "information to characterize a situation" (Dey) and not all aspects of the "execution environment" (Schilit) can be made available to computing systems. Properties that cannot be sensed or obtained from a readable source are impossible to use in context-aware computing. This leads to the definition of *sensed context* as

> *properties that characterize a phenomenon, are sensed and that are potentially relevant to the tasks supported by an application and/or the means by which tasks are performed.* [Gray et al. (2001)]

This does, however, not mean that all aspects of sensed context have to be obtained from a single source. Some aspects like the social situation cannot be read form a single sensor but have to be determined by evaluating several sources (*context services)*. For example, the same contextual aspect can be obtained from different sets of sensors according to the situation, or results from several sensor sources can be evaluated according to a probability function. Abowd et al. (2001) call this *context fusion.*

2.1.1.2 Context-Awareness and Context-Aware Computing

The first research related to context-awareness was the "*Active Badge*" system developed by Want et al. (1992) [see Dey et al. (1999)]. Small badges could be used to track the location of office-workers. The *Call Forwarding* system based on the Active Badge system could then be used to forward telephone calls to the phone that was closest to the callee's current position. Want didn't discuss c*ontext-aware computing* explicitly, so it was Schilit who first defined it as software that

> *adapts according to its location of use, the collection of nearby people and objects, as well as changes to those objects over time.* [Schilit et al. (1994b)]

The term *context-aware* has been used in various ways. Furthermore, many expressions are used synonymously, as listed by Dey et al. (1999), including "*reactive,*" "*responsive,*" "*situated,*" "*context-sensitive,*" "*environment-directed,*" and "*adaptive*". The relationship of adaptivity to adaptability that was analyzed by some authors is described in the next chapter. Dey compares previous definitions of context-awareness and complains that they either concentrate on using context or on adapting to context. Therefore, he gives a new general definition:

> *A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.* [Dey et al. (1999)]

2.1.1.3 Adaptivity vs. Adaptability

Adaptivity is an important facet of context-awareness. Some authors even use both terms synonymically. Several authors have stressed the potential of better meeting the user's needs by letting the computing system adapt to the current situation [Oppermann (1994a), Lieberman et al. (2000)]. However, Wulf et al. (2001) also shows the boundaries of automatic adaptivity [see also Friedrich (1990)]. Computing systems sometimes cannot provide all information that is needed to make a decision in a certain situation. Another way to increase usability and flexibility of software systems is adaptability, i.e. the possibility to let the user actively tailor applications according to his needs [Stevens (2002b)].

Bringing adaptivity and adaptability together, Klann et al. (2003) proposed are "shared initiative" approach. While most research activities address the former two approaches separately, they propose to use a joint approach to compensate for the respective shortcomings of the others. For example, adaptability can be enhanced through adaptivity by identifying recurring tailoring activities and reusing existing solutions. On the other hand, adaptivity could be enhanced through adaptability when the user chooses what properties to take into account so that an automatic decision can be made, or when he provides missing parameters in the user-specific settings of an adaptive software component [see also Oppermann (1994b)].

Using a system changes the user, which lets him use the system in new ways [Nielsen (1993)]. These new ways often require new functionality in the system. This "co-evolution of system, technology, environment and users" [Bourguin et al. (2001)] cannot only be supported by enabling tailoring functions, but also by letting end users develop the system according to their needs. These "activities or techniques that allow people, who are non-professional developers, at some point to create or modify a software artifact" are called *End User Development (EUD)* [Costabile (2002), Klann et al. (2004)]. To realize EUD, the focus has to move from "easy-to-use" to "easy-to-develop interactive software systems" [Lieberman et al. (2003)]. EUD can be supported by component-based architectures [Stevens (2002a)] to achieve component-based tailorability [Stiemerling (2000)], which lets end users „select, modify, integrate, and share components to create new functionality" [Mørch et al. (2004)]. To encourage users to actually use tailoring functions, many authors demand a *gentle slope to tailorability* [Mørch et al. (2004)]. This demand is taken from the more general claim of a *gentle*

slope of complexity, which means to make simple tasks easy to fulfill, to demand a gradual increase of knowledge for more complex tasks, and therefore to "keep a reasonable trade-off between ease-of-use and expressiveness" [Costabile et al. (2003)].

2.1.2 Context Dimensions

Most theoretical works on context-awareness also try to describe what context and context-awareness consist of. While some authors simply list different features of context, others provide a more structured categorization.

Chen et al. (2000) list location, time, nearby objects, network bandwidth, orientation, light level, proximity of humans, sound, temperature and pressure as *low-level* types of context as well as the user's current activity and the social context as examples for *high-level* types of context.

Dey et al. (1999) distinguish *primary* and *secondary context*. Primary context information describes the entity's situation and is used as an index to retrieve secondary context. For example, the user's personal preferences can be derived from his identity. Dey's primary context types are picked up and revised by many other authors in the discussion of context dimensions. Oppermann et al. (2001) summarizes four main context dimensions:

- *Location*: Most authors mention the physical location of an entity that can be used for so called *location-based services*, but Oppermann also includes the electronic location, that can be described by Uniform Resource Locators (URLs) or similar mechanisms.
- *Identity*: The user's identity can be used to infer many secondary context types, such as the user's preferences, his knowledge, and his history of activities.
- *Time*: The current point in time as well as categorical scales (e.g. working-hours vs. weekend, day vs. night) can be relevant for context-aware applications.
- *Environment or Activity*: This dimension varies most among different authors. Dey et al. (1999) consider activity as the fourth context dimension and describe it as the answer to "the question of what is occurring in the situation." Oppermann calls this dimension "activity or environment" that "describes the objects and the physical location of the current situation."

Pham (2001) substitutes the more general dimension of environment or activity with *infrastructure* as his forth context dimension [based on a classification by Schmidt et al. (1998)]. He describes it as the "knowledge about the available computing resources and infrastructure in the close vicinity."

2.1.2.1 Dimensions Relevant to Context-Aware Help Systems

For context-aware help systems, several of the context dimensions are relevant.

Location can be used in two ways: The position of the handheld computer relative to the device and the geographic position. The geographic location can be particularly useful for mobile help systems in the automotive domain, for example when the car indicates an error and the help system shows a list of nearby dealers. The relative position could be used to detect which part of the device the user currently watches. A help system for a large machine might need to show different information if the handheld computer detects that the users stands at the backside of the machine.

Identity can be exploited in various ways in a context-aware help system. The personal handheld device monitors the user's interaction and stores various personal information. Both can be used to determine the user's preferences and his level of knowledge. Based on this, the content and UI of the system can adapt to the identity.

The most important entity of the environment is the device that the help system is made for. Therefore, relevant aspects of the environment or activity dimension include the internal state of the device as well as current or previous interaction with it. For a user that is stuck in the middle of a task, his recent interaction with the device is central for the help he needs. For example, if a user just opened the hood of his car and the help system is able to detect this, it can provide the user with descriptions of common tasks for the current situation, e.g. refilling the water for the windshield washer. If the car's internal state also includes the oil temperature being too high, the help system might show information for checking the oil level.

2.1.3 Categories of Context-Aware Computing Systems

Patterns to classify context-aware applications have been provided by several authors. Schilit et al. (1994a) provide a taxonomy based on two orthogonal dimensions as shown in Table 1.

The horizontal dimension describes whether the application obtains contextual information manually or automatically, whereas the vertical dimension describes if the application's task is to get information or to execute a command. *Proximate Selection and Contextual Information* is the class of applications that presents a number of choices (e.g. a list) to the user where the choices most relevant to the user's context are emphasized. *Automatic Contextual Reconfiguration* stands for applications that show context-relevant information to the user automatically. Hence, it also includes resource discovery, i.e. the dynamic search for new or changed nearby objects. *Contextual Commands* are the manual execution of commands that are available due to the user's context. *Context-triggered Actions* are based in a simple if-then logic and are executed automatically based on the user's context.

	MANUAL	**AUTOMATIC**
INFORMATION	**Proximate Selection & Contextual Information**	**Automatic Contextual Reconfiguration**
COMMAND	**Contextual Commands**	**Context-triggered Actions**

Table 1: Categories of context awareness [Pham (2001)].

Dey et al. (1999) combine this classification with the work of Pascoe (1998). He merges Schilit's Proximate Selection and Contextual Commands to a new category of "*presentation of information and services to a user.*" Furthermore, Schilit's Context-triggered Actions are renamed "*automatic execution of a service.*" Dey leaves out Automatic Contextual Reconfiguration because he sees it as a part of his first two categories. Automatically showing contextual information and resource discovery can be seen as part of presenting information according to the context. Finally, he adds a new category called "*tagging of context to information for later retrieval*" (based on Pascoe's "*contextual augmentation*"). Applications of this class attach digital information to physical or virtual objects to augment the environment.

The context-aware mobile help system that is presented in this thesis has features that fall into two of Dey's categorizes. The features are described in chapter 4.2. The contextual links-feature that shows a number of links to topics that are most relevant according to the current context belongs to the category of "presentation of information and services to a user" while the context-triggered help-feature that shows important warnings or alerts according to the device's context belongs to "automatic execution of a service."

The context-aware mobile help system that is presented in this thesis belongs to Dey's category of because it presents a number of links to information that is helpful according to the current context, but leaves the decision to follow one of those links to the user.

In the following chapter a categorization of previous context aware computing systems according to Dey's taxonomy is presented. This will show that the concept of exploiting contextual information in mobile help systems is a rather new approach that differs from many early works in the field because it concentrates on several context dimensions including activity and identity and combines using these dimensions with presenting information to the user. Furthermore, it is used in a different domain and integrates contextual help with interactive usage.

2.1.4 Existing Systems - Categorization and Evaluation

Dey et al. (1999) provided the basis for Oppermann's summary of context dimensions as well as a categorization of context-aware computing systems, as described in the previous chapter. Furthermore, he tried to catalogue various context-aware applications of recent years. Table 2 lists which application uses which context dimensions and which applications fall into which class of his taxonomy.

This categorization shows that applications can be found for every context dimension and every category of context-awareness. However, the systems listed by Dey concentrate on using location. This was also found by a later work of Chen et al. (2000) that provided a similar categorization of mobile context-aware systems and concluded that most context-aware application in research so far concentrate on location.

The iManual system presented in this thesis concentrates on identity and activity rather than location and was categorized as "presentation of information and services to a user" and "automatic execution of a service". There is no other application in Table 2 that also falls into these categories (A, I; P, E). This indicates that integrating context-awareness into a mobile help system by using activity (or "environment or activity," as other authors call it) and identity has not been investigated by many research groups and could therefore be a promising approach.

System Name	System Description	Context Type				Context-Aware		
		A	**I**	**L**	**T**	**P**	**E**	**T**
Classroom 2000 [1]	Capture of a classroom lecture			X	X			X
Cyberguide [1]	Tour guide		X	X		X		
Teleport [2]	Teleporting	X	X	X			X	
Stick-e Documents [3,4,5]	Tour guide Paging and reminders	 X	X X	X	X	X	 X	X
Reactive Room [6]	Intelligent control of audiovisuals	X	X	X			X	
GUIDE [7]	Tour guide			X	X			
CyberDesk [8,9,10]	Automatic integration of user services	X				X	X	
Conference Assistant [11]	Conference capture and tour guide	X	X	X	X	X		X
Responsive Office [12]	Office environment control			X	X		X	
NETMAN [13,16]	Network maintenance			X		X		
Fieldwork [17,18,22]	Fieldwork data collection			X	X	X		
Augment-able Reality [19]	Virtual post-it notes			X		X		X
Context Toolkit [24]	In/Out Board Capture of serendipitous meetings		X X	X X	X X	X	 X	 X
Active Badge [28]	Call forwarding		X	X		X	X	

Symbols used: Context Type: A = Activity, I = Identity, L = Location and T = Time; Context-Aware: P = Presentation of information and services to a user, E = Automatic execution of a service and T = Tagging of context to information for later retrieval. Numbers in brackets stand for references used in Dey's original work.

Table 2: Application of context and context-aware categories [Dey et al. (1999)].

2.1.5 Challenges of Context-Aware Computing

Realizing context-aware systems holds a great potential, but there are also several challenges that have to be met to ensure success of these system.

Lueg (2002) describes the basic problems of context-aware systems when trying to answer a question of Satyanarayanan (2001): "Why do these [pervasive computing] scenarios seem like science fiction rather than reality today?". Context-aware systems are based on "approximations of potential users and usage situations". These approximations are described as "assumptions about which aspects of human activities and their

physical and social environment will be important in future usage situations". However, making good approximations can be extremely difficult in some areas of context-aware computing. It can be very hard to decide which contextual parameters are relevant for the system because of the dilemma that "context is shaped by the specific activities," but "these activities also influence what participants treat as *relevant* context."

A related problem is the *frame problem*, which describes the difficulty to decide which aspects of the world are relevant parts of a sufficiently detailed world model and how a system can keep these aspects up-to-date [Lueg (2002)]. A complete world model is infeasible because the features of the world are infinitely rich and constantly changing, so context-aware systems have to find a well-balanced compromise. If a context-aware help system was evaluated and weaknesses were identified, the frame problem could be the reason. It would then have to be analyzed if other parts of the context have to be considered to provide a better system behavior.

Other challenges that all kinds of context-aware systems have to face are privacy and security issues. These potential problems arise because context-aware systems store various kinds of personal data to be able to adapt to the user's identity. If context-aware systems get more popular and powerful, the amount of data also increases. If the context-aware system is used by several users, this personal information is possibly stored on a central server. Furthermore, users might use portable devices to connect to a server over a wireless link. These links might then be used to get unauthorized access to this variety of personal information. A context-aware system therefore has to ensure that no unauthorized sources can access any kind of user data [Amor (2002), Cas (2002)]. For scenarios where physical devices can be controlled by handheld computers, a *secure and transient association* has to be ensured [Stajanao (2002)]. This means that a physical device can only be remote controlled by authorized users, that an established connection cannot be cut by other users and that the association is completely removed if the user is not authorized anymore.

A challenge that is specific for a context-aware help system is the *granularity problem*. This means that the help system has to find a representation to store the pieces of content in a way that various versions of the content can be presented according to each user's context. This problem is described in more detail in chapter 3.4 because it is not a challenge for every context-aware mobile system but very specific to a mobile help system.

2.1.6 Summary

Context-awareness is a promising approach to improve existing systems or even to build systems with features that would otherwise be infeasible to realize. Using it in help systems holds some special possibilities that most research activities in context-awareness did not especially concentrate on. But context-awareness is not the only strategy to improve existing help systems and create new intelligent ones. In the following chapter, these strategies and their background are described and evaluated.

2.2 Intelligent Help Systems

Help systems exist for all kinds of systems that users have to learn about, including software systems and physical devices. As Covi et al. (1995) explain, the development of help systems is one of the hardest problems of HCI because it requires understanding of the tasks that the system support, the system itself and the characteristics of the user. This, of course, is not only related to software systems but to all systems that require a help system. This chapter classifies help systems, explains the psychological background and describes strategies to solve this hard problem, i.e. to develop an intelligent help system. However, I will first motivate this chapter by answering a common question that is often asked when talking about help system.

2.2.1 Do we Need Help Systems at all?

Users don't want to use manuals, they want to achieve goals. A perfect device would be one that does not need a help system at all because all the functions can be used correctly without additional explanation or training. This device could be used to achieve the goal without demanding too much of the users attention. However, the scenario that all devices will turn into such easy-to-use and unobtrusive things that don't need any explanation is not very likely for two reasons:

First, our surrounding devices are getting more powerful and therefore, more complex. Cars for example come with more and more equipment that makes them safer, more secure, more comfortable, and more flexible. With more things to control and to use, the effort to learn and understand all these features gets of course more difficult. Figure 1 exemplifies how complex the GUI of a standard text editing program can get.

Norman (1998) distinguishes *complexity* and *difficulty*. Complexity is used for the internal state of the device while difficulty refers to the external state. In this terminology,

difficulty determines the ease-of-use of the device while complexity is only relevant for technicians and manufacturers. Norman stresses that it is possible to hide internal complexity behind good user interfaces. It is important for a device to

> *present a clear conceptual model of itself, making the possible actions apparent. [...] when the world itself helps tell you what to do, [...] no manuals are necessary.* [Norman (1998) p. 175)]

However, this is only possible to a certain extend for simple devices being used for simple tasks. In specialized devices

> *ease of use is directly related to the effort level of the task. Easy tasks can be done effortlessly; hard tasks may still be difficult.* [Norman (1998) p. 60)]

Figure 1: Microsoft Word with all toolbars shown [Baecker (2003)].

Furthermore, a good conceptual model has to make it easy to remember the function of control elements and to give immediate and continuous feedback. Both can be delivered by an intelligent well-designed help system.

Second, new UIs for complex devices might even get highly intuitive. However, as Raskin (1994) points out, “intuitive” does not mean being usable without explanation or training. It just means using “readily transferred, existing skills.“ Introducing new intui-

tive features means that users can rely on well-established concepts and are able to transfer their existing skills to use these new features. Therefore, when introducing new features of a device for the first time, there always has to be some kind of explanation or training to show how to transfer existing skills. E.g., a car manufacturer might introduce a new intuitive interface to control the air conditioning. Nevertheless, all users that don't have the existing skills to use it will need at least one explanation from the manual.

Even for users that belief to know everything about the system they use, an intelligent help system can be helpful. First, it can show how to use advanced features that the users might not be aware of and second, it can show ways to complete tasks that are faster or more efficient than the way the user fulfills the task. As an example from the author's personal experience: I always thought it would be unnecessary to read my mobile phone's manual [Siemens (2003)] because I thought I knew all its functions. However, I always used a procedure of ten single key presses to insert a number into an SMS and return to the text entry mode when using T9 [America Online (2003)], until a friend showed me to press and hold the key to insert the number with a single key press – a feature that was well documented in the manual.

Hence, the solution is not to ship complex devices without manuals or documentation but to integrate using the device and getting help from the intelligent help system so that users achieve their goals more easily.

2.2.2 The Psychology of Requesting Help

Designing a help system, whether intelligent or not, cannot be successful without understanding the process of learning. Various authors have investigated the psychological background of why and how people learn. Most of them also differentiate children and adult learners as they behave differently in many situations. This thesis only considers adult learners.

2.2.2.1 Why people learn

Most adults do not learn for the sake of learning. When they consult the help of a new tool, they don't want to learn how to use the tool but how to complete a task. When they are stuck in the middle of an action and access reference help, they don't want to learn how to operate the system but how to achieve their goal [see Smart et al. (1998), Carroll (1990)]. Users always have individual goals and their personal expectations when they learn. If they are misled by their expectations, learning can even be frustrating [Carroll

(1990)]. Because users want to achieve goals and complete tasks, they are impatient when confronted with long texts and want to get started quickly instead [Brockmann (1990), Rettig (1991)].

2.2.2.2 How people learn

The way adults learn was investigated by Carroll et al. (1984) and summarized by Baecker (2003) in three general forms of learning: learning by doing, learning by knowing and learning by thinking.

The importance of *learning by doing* was analyzed theoretically and empirically by several authors. Adult learners are discouraged by long texts and tend to skip around in help documents. They prefer to work on real tasks rather than reading theoretical information. They always want to try things out and explore the world they learn about. Carroll showed in empirical tests that users are more satisfied and more effective in learning by exploring the system [Carroll (1990)]. Brockmann (1990) summarized that adult learners are "best motivated by self-initialized exploration." Furthermore, humans always make mistakes when first using a system. Brockman states that users most often learn from recognizing and correcting those errors. Carroll showed that recovering from errors often is a "matter of creative problem solving," an approach that can lead to a deeper understanding of the problem domain [Carroll (1990)].

Learning by knowing means to use one's prior knowledge to gain new understandings. Adult learners always have a history of learning and building knowledge, and they always, consciously or not, try to relate new information to their prior knowledge. Furthermore, they are situated in a complex world full of contextual information. Everything they read or hear is associated with their current context, their experiences, and expectations. A help system that is too detailed or too strict and does not leave room for making one's own associations fails to encourage learning. Carroll (1990) calls this the *paradox of sense-making*.

Learning by thinking as the third form of learning lets the learner use his mind to construct a cognitive representation of the things learned. This *mental model* [Naranyanan et al. (1998), Steehouder et al. (2000)] can then be used to solve problems without always needing the help documents and to adapt one's knowledge to new situations.

Various authors have tried to use the findings of why and how people learn to improve the quality of help systems. I will now first classify help systems, situations of request-

ing help, and help content and then, based on that classification, describe and evaluate several strategies to improve help.

2.2.3 Classification of Help Systems

2.2.3.1 Terminology

In the rest of this thesis, I will use the term *help system* when talking about a general facility that can provide assistive information [compare Kearsley (1988)] and *manual* when talking about a help system that is related to a software program or physical device. Some authors use a stricter definition of these terms. Lacey (1996) describes a manual as a „set of instructions, lengthy enough to be bound yet small enough to be easily handled“. An *intelligent help system* is not strictly defined in this thesis. Instead, *intelligent* is used to describe a broad range of approaches that make a help system act and react more helpful for the user.

2.2.3.2 Classification of Help Systems

Some authors classify help systems according to the medium that is used, e.g. paper-based help, electronic help (or online-help) etc. Furthermore, most modern help systems use various media types such as text, image, video, audio, or any combination of these. Of course, paper-based systems can only use a limited variety of media types. Hypertext can be seen as a special type of text that allows non-linear navigation through hyperlinks, and animations, e.g. 3D-Animations, can be seen as a combination of images, text and video. An electronic help system can also be interactive, e.g. when giving the possibility to tailor the presentation of the content or to navigate through the content individually. The special chances and challenges of interactive help are described in chapter 2.2.4.5. The distinction as to which medium is used by a help system and which media types it includes is not helpful when talking about intelligent help, because using more advanced media does not mean being more or less intelligent. New media bring new possibilities, but what makes a system intelligent are the strategies that use the possibilities of the medium, not the medium itself.

Covi et al. (1995) provides a very broad definition of online-help including *document-oriented systems* and *computer mediated help*. Document-oriented systems, according to Covi, can be online manuals, hypertext systems and self-explanatory objects while computer mediated help can be e-mail communication with a human expert, a web-

based group forum and a searchable archive. Intelligent help systems can also support computer-mediated help, but in the remaining chapter, I will concentrate on what Covi calls document-based systems.

Help systems can be classified as being passive or active, context-independent or context-sensitive, and user-independent or user-sensitive according to Bauer et al. (1988). *Active help* is given when the system assumes that the user might need assistance while *passive help* is triggered by an explicit action of the user. *Context-sensitive help* (or *dynamic help* [Oppermann (2001)]) considers the system's current contextual situation while *context-independent help* (or *static help*) does not. *User-sensitive help* (or *individual help*) adapts the information to the user's preferences, requirements or experiences while *user-independent help* (or *uniform help*) provides identical information for every user. According to the definition of context-awareness given in chapter 2.1, user-sensitive help systems would also be classified as context-aware, so in this thesis, I use the term context-aware help for systems that consider any of the context dimensions and refer to user-sensitive help only to stress that these help system concentrate on the context dimension of identity. Active and context-aware help systems are described and evaluated in chapter 2.2.4.

2.2.3.3 Classification of Situations of Requesting Help

When trying to create intelligent help systems it is also important to consider in which situation help is requested by the user. Some authors provided a classification of situations of requesting help and most of them simply distinguish two basic situations: the learning situation and the reference situation [Ummelen (1997)]. The requirements to the help systems and the way of using it are fundamentally different in these situations. A user is in the learning situation when he decides to learn how to perform a task with the system. The reference situation occurs when the user is in the middle of using the systems and consults the help system to find specific information to complete his task.

Solem (1986) distinguishes situations of requesting help according to three user attitudes called *no-time-to-learn, want-to-learn,* and *know-what-I-want.* While the want-to-learn attitude is mostly equivalent to the learning situation and the know-what-I-want attitude to the reference situation, Solem adds the no-time-to-learn attitude to describe a situation, where the user wants to get started quickly and needs compact "quick-steps"-information.

Baecker et al. (1990) provide a more detailed classification of situations of requesting help. This "taxonomy of user questions" identifies several questions that a help system should answer:

Identification:	*What is this?*
Transition:	*Where have I come from and gone to?*
Orientation:	*Where am I?*
Choice:	*What can I do now?*
Demonstration:	*What can I do with this?*
Explanation:	*How do I do this?*
Feedback:	*What is happening?*
History:	*What have I done?*
Interpretation:	*Why did that happen?*
Guidance:	*What should I do now?*

[Baecker et al. (1990)]

2.2.3.4 Classification of Help Content

When designing an intelligent help system it is important to be aware of the different kinds of help content and to present the kinds of content to the user that best fit the current situation. Several authors classified different kinds of help content and investigated which kind of content should be used in which situation.

Many authors basically distinguish between *declarative* and *procedural information*, although several different terms are used. Ummelen (1997) defines procedural information as "directly supporting actions" that describe conditions for actions, actions and results from actions, whereas declarative information "supports comprehension and factual knowledge." This classification is orthogonal to the media type classification of help systems, because both procedural and declarative information can contain any kind of media and because within one type of media, there can be both procedural and declarative parts. The discussion whether declarative information is useful and needed at all will be described in chapter 2.2.4.2. Declarative information is also sometimes called *descriptive*, *explanatory*, or *background* information. Carroll et al. (1988) calls declarative information *How-it-works help* and procedural information *How-to-to help*. Steehouder et al. (2000) describes *operational information* as a subtype of declarative information that "describes how the system works." Farkas (1998) describes four subtypes of procedural help: *procedures*, which are used to present various kinds of instruc-

tions, *tutorials*, which are specially designed instruction that explicitly support retention, *performance support help*, which includes "wizards" and "coaches" [Microsoft (1995)] and supports working and learning simultaneously, and *interface-based documentation*, which is directly embedded into the graphical interface of the software.

Procedural information is often used to support learning by doing while learning by thinking is supported by declarative information.

The classification of help systems, situations of requesting help and help content is the basis of identifying and describing different strategies to improve help systems, which are presented in the following.

2.2.4 Evaluation of Strategies to Create Better Help Systems

This chapter presents different strategies to improve help systems, i.e. to develop more effective system with a higher usability that provide assistance that better meets the user's needs. The strategies are not meant to be disjunctive. Some advanced approaches are based on earlier approaches and some features of modern help systems can be assigned to more than one strategy. In general, to achieve improved the help they provide, help systems have to take a hybrid approach – which most modern help systems do – using the most successful features of the following strategies.

2.2.4.1 Systems Approach

The Systems Approach [Dick et al. (1996), Andrews et al. (1980)] is based on the work of Gagne et al. (1979) and influenced by cognitivist psychology theories. This approach tries to decompose the educational content into small pieces that can better be organized and presented. These pieces can then be used as a basis for instructions.

Carroll made empirical experiments that showed that users had significant trouble in using manuals that followed the systems approach [Carroll (1990)]. A detailed list of instructions that users simply have to follow fails to support the basic form of learning – learning by doing – because it does not leave enough room for individually exploring the system. Following simple steps

> *may be both too much and too little to ask of people. […] People are always already trying things out, thinking things through, trying to relate what they already know to what is going on, recovering from errors. In a*

> *word, they are too busy learning to make much use of the instructions.* [Carroll (1990), p.74]

2.2.4.2 J. Carroll's Minimalism

Based on his findings about the weaknesses of the systems approach, Carroll (1990) developed the *Minimalist Design Principles* also called *Minimalism*. He also designed an exemplary manual that followed these principles called the *Minimal Manual*. Several authors continued to investigate about these principles, both theoretically and empirically.

Carroll describes several principles that a minimalist manual should follow. First, it should provide "training on real tasks" and allow "getting started fast." Users don't want to learn to use the system, they want to complete a task. Brockmann (1990) adds to "focus on what readers need to know to immediately apply it to productive work" and to "cut secondary features […] – overviews, introductions, summaries etc." Second, the manual should support "reasoning and improvising" and "reading in any order" to give the user a chance to explore the system and the manual on his own and to start learning by thinking. This was, for instance, achieved in Carroll's exemplary minimal manual by organizing the content as a set of training cards that each addressed a specific question and that did not depend on material covered by other cards. Brockmann suggests supporting active exploration by intentionally incomplete information. Third, Carroll stresses to „coordinate system and training." In paper-based systems, minimalist manuals should support the coordination of system and manual, especially in step-by-step instructions. Fourth, a minimalist manual should support "error recognition and recovery" by identifying common errors, giving appropriate information, and supporting to recover from errors on one's own. Carroll mentions that the well-known *undo* facility of many software applications supports error recovery and helps to "unravel the error situation." Finally, the manual should support learning by knowing through "exploiting prior knowledge," e.g. by using common metaphors or analogies.

Meij et al. (1998) later summarized four major principles of minimalism and also provided heuristics to each principle in order to make them easier to apply to real manuals (see Figure 2).

Principle 1: Choose an action-oriented approach
Heuristic 1.1: Provide an immediate opportunity to act
Heuristic 1.2: Encourage and support exploration and innovation
Heuristic 1.3: Respect the integrity of the user's activity
Principle 2: Anchor the tool in the task domain
Heuristic 2.1: Select or design instructional activities that are real tasks
Heuristic 2.2: The components of the instruction should reflect the task structure
Principle 3: Support Error Recognition and Recovery
Heuristic 3.1: Prevent mistakes whenever possible
Heuristic 3.2: Provide error information when actions are error prone or when correction is difficult.
Heuristic 3.3: Provide error information that supports detection, diagnosis, and recovery
Heuristic 3.4: Provide on-the-spot error information
Principle 4: Support reading to do, study and locate
Heuristic 4.1: Be brief; don't spell out everything
Heuristic 4.2: Provide closure for chapters

Figure 2: Four main design principles for minimalist instruction [Meij et al. (1998)].

Farkas (1998) showed how minimalist help systems can be enhanced with a technique called *layering*. A risk of minimalism is that too many information might be cut so that users are unable to complete a task or develop an incorrect mental model. Layering means providing reading choices to the users. Thus, the user can chose to read more detailed information on a topic or chose to skip that extra information. Layering can be achieved in hypertext by simply providing carefully-named links (e.g. "HOW DOES THIS WORK?") to additional information. Layering is one way to tailor content to the individual needs of the users. Expert users may choose not to click on the links while novices might need several "layers" of links to more detailed information.

Several studies rewrote existing manuals according to the minimalist design principles and then compared using the original and the minimalist manual. All of them showed that minimalism brings significant improvements to the usability of manuals.

Carroll (1990) showed that users with a minimal manual required 40% less training time than users with a manual following the systems approach. Nevertheless, they completed 2.7 times as many subtasks in the test period. Furthermore, they made 20% less errors and, even more important, spent 10% less time recovering from their errors. When using the manual in a reference situation, minimalist manual users were successful in finding the desired information in 41% of all tries in contrast to 31% for the system-style group. Hewlett-Packard [Anson (1998)] applied minimalist design principles to some of

their product documentation. In a comparison with traditional documentation users, minimalist manual users needed 10% more time to work through the tutorial but then had fewer problems and completed a test faster and more accurate. HP later showed in usability tests that users liked minimalist documentation shipped with network printers in 1994 (270 pages) better that traditional documentation for network printers shipped in 1993 (650 pages).

The question of how much declarative information can be left out of a manual is still discussed controversially. Minimalism in general suggests leaving out most declarative information without being more specific. Carroll et al. (1984) found out that users tend to skip declarative information and Charney et al. (1988) observed that providing no, little or many declarative information had no effect on user performance. Price (1984) only advises technical writers to "separate what to do from what it means" while Boedicker (1990) stressed that too much background information is not only unnecessary but even harmful. However, Ummelen (1997) showed that users still spend 31% of their time on reading declarative and 69% on procedural information, so the fact that users completely skip declarative information would be overdrawn. This was tested using the *click-and-read technique* that blurs blocks of the text except for the headings so that the user has to click on a block to read it. Another test showed that users that both got procedural and declarative information performed better in tests that were carried out after some period of time and in tests that concentrated on reasoning than users with procedural information only. They were faster, needed fewer attempts and knew better how procedures could be done more efficiently. These results seem to show that providing procedural *and* declarative information lets users build a more complete mental model, which then can be applied it in future tasks. These findings were used by Steehouder et al. (2000) to suggest three enhancements to step-by-step instructions. He suggests adding declarative information, to create a hierarchical structure by adding a functional level, and to relate sequences of steps to real life goals.

Minimalism and the systems approach are not restricted to any type of medium, although most studies refer to paper-based manuals. The following strategies are limited to electronic media. Most studies are related to help systems for computer programs. Later I will show that some of the following successful strategies can also be applied to help systems for physical devices.

2.2.4.3 Active Help

In chapter 2.2.3 I already described the difference between active and passive help. Active help presents assistive information to the user although he did not explicitly request help. The help system has to assume that the user might need help and to show information that the system has identified to be helpful in the current situation. Determining which information the user currently needs according to his situation is part of context-aware help, which is described in the next section, so this section only describes the advantages and challenges of help that shows up without a user request. Embedded help, which is also mostly active because it is embedded into the GUI and shown during the user's interaction with the system, is described and evaluated in chapter 2.2.4.6.

Active help is often tested with the *wizard-of-oz technique* [Carroll et al. (1984), Nitschke et al. (2001)]. This technique uses a human expert that simulates advanced features of the computer system that are not implemented yet. Carroll showed that many users appreciated active help that explained how the currently used functions work, previewed consequences of those actions, and provided confirmation of a plan that the user was unsure about. However, he also showed that there is a very thin line between giving confirmation and being annoying when information appears that the user already knows or that disturbs completing the current task. Nitschke also shows the problem that too detailed or unrestricted active help does not support the learning process. Test users did not improve in delayed tests after having received unrestricted active help in contrast to users that used paper-based operational instructions.

2.2.4.4 Context-Aware Help

Context-aware help, which is also called *context sensitive help* or *situated help,* is described by many authors and used by companies in various ways. What differs in how authors or companies understand context-aware help is the use of which of the context dimensions and the extent to which context-awareness is being tried to achieve. Brockhaus' dictionary defines context sensitive help as giving „information that always automatically refers to the current working situation, the currently used commands and control elements" [Bibliographisches Institut & F.A. Brockhaus AG (2001), translated by the author]. According to the context-dimensions defined in chapter 2.1, this definition only uses the environment or activity dimension. This thesis uses a broader definition of context-aware help that considers all context dimensions.

Silveira et al. (2001) extend the definition of context-aware help and distinguish between interface-context sensitivity, user-profile sensitivity, task sensitivity and user-intent-sensitivity.

Interface-context sensitive help gives information according to user's interaction with the application's GUI and is embedded into the GUI. It is described in a separate chapter (2.2.4.6 Embedded Help and Self-Explanatory Objects) because the feature of being embedded into the application is more predominant than being context-aware. *User-profile sensitive help* is a system that presents information tailored to the user profile, which typically contains the user's preferences, his expertise, and history of interaction. User-profile sensitivity uses a profile to adapt the help content to the user's identity. Selker (1994) presented a system that builds an "adaptive user model" which is used to achieve user-profile sensitivity. Silveira's last two categories classify help systems that take into account what the user currently does (tasks) and what he plans to do (intentions). *Task-sensitive help* is help that guides the user through several steps to let him complete his task. If the system knows what the user's task is, this is, according to Silveira, rather easy to realize, for example in Wizards or Step-by-Step instructions. The more difficult part is trying to find out what the user intends to do and which tasks he plans to carry out. Silveira calls this *user-intent sensitive help*. One way of achieving user-intent sensitivity is providing feedback opportunities that are described in the following chapter. The iManual system presented in this thesis also tries to achieve user-intent sensitivity, which is described in chapter 3.1.

Research has not defined clearly what context-sensitive help means, so most empirical evaluations were related to systems that were context-aware only to a certain extend. Nevertheless, many researchers showed that well-designed context-sensitive help can be valuable for learning and working with a system. In an empirical study simulating intelligent help systems, Carroll et al. (1988) summarized "being smart can help." They showed that intelligent help systems that were aware of the current context could give users very helpful assistance. The problem for most real-life systems is, however, the difficulty to measure all relevant aspects of the current context. Furthermore, the more context-aware and therefore intelligent a help system is, the more responsible is it made by users for every inconvenience. With context-aware help systems, users can get used to getting information that perfectly fits their situation. However, they might get frustrated when the system fails to deliver appropriate information or makes any other kind of mistake.

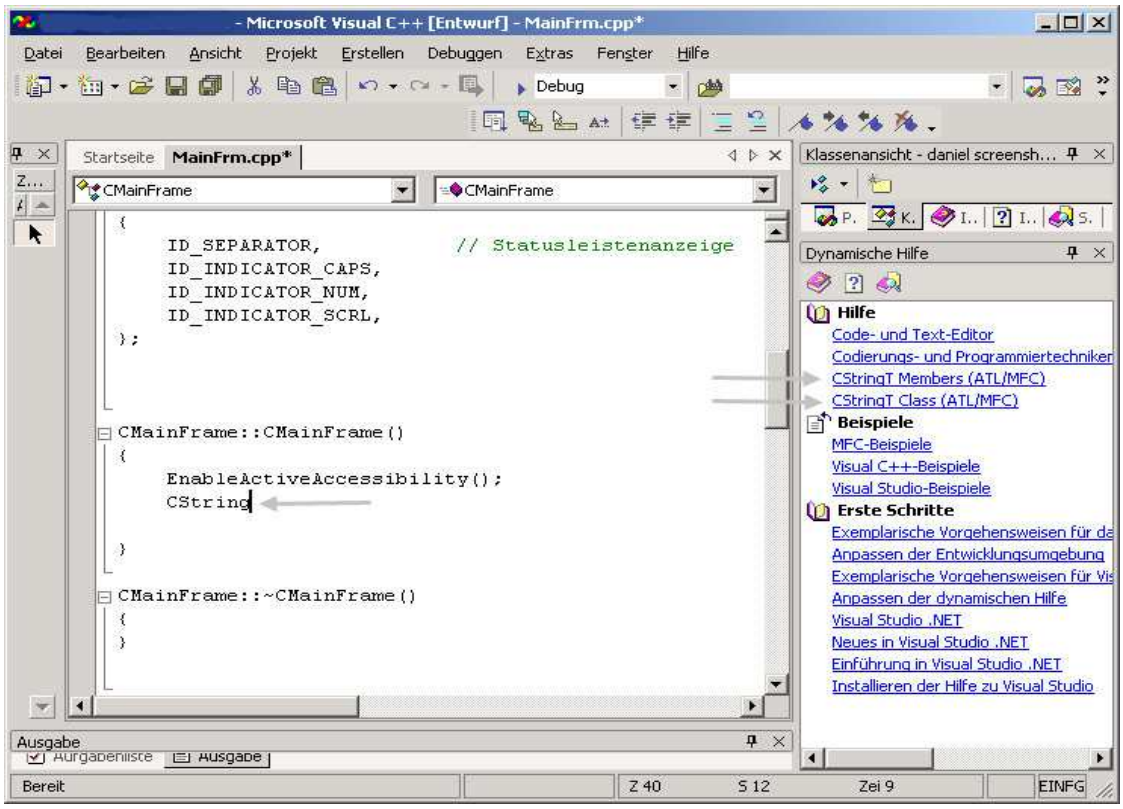

Figure 3: Microsoft Visual Studio.NET Dynamic Help [Microsoft (2003a)].

Microsoft included an improved version of context-sensitive help into their IDE Visual Studio.NET [Microsoft (2003a)]. Their *dynamic help* tries to exploit activity, a context dimension defined earlier, by keeping track of the user's actions. Then, hyperlinks to help topics that are related to these actions are shown. In usability tests, Microsoft found that this kind of help improved overall usability of the product [Microsoft MSDN News (2001), Microsoft (2002), ExtremeTech.com (2003), see Figure 3].

Randall et al. (1998) stated that context-aware help is one step towards building a human-like *ethos* in the help system. Ethos first appeared in Aristotle's Rhetoric [Aristotle Rhetoric (1990)] as the "character" of a human speaker and was transferred to the situation of a help system as "the character, image, and the presence of human-like or human-appealing attributes within the system." Even though there is no human present within a system, the user might get the impression of the help system reacting like a human. A Context-aware help system suggests to users that it is "intelligent enough to know what they are trying to discover," a feature that underlines the ethos of a useful, informative human assistant. Randall et al. point out that such an ethos can narrow the distance between users and help system or even lead to the impression that the user does

not need help at all, because the help system unobtrusively presents the information needed to fulfill the task.

Oppermann (2001) mentions the problem of being up-to-date. As user preferences, wishes, intentions and moods can change from one moment to the other and depend on many factors including the current task, it is difficult for the help system to keep the user-model up-to-date and present help content according to it. He also mentions that it is difficult to decide how to represent conceptual information. Furthermore, if the help system uses a metaphor to help the user understand new concepts, it is difficult for the system to decide which metaphor is appropriate for the current situation.

All in all, most studies show that context-aware can be helpful to create help systems that users like better, but also that the biggest problem of how to retrieve all relevant contextual information must be solved for every system individually.

2.2.4.5 Interactive Help and Help With Feedback

Early help systems were always paper-based and therefore couldn't be used interactively. Help systems using electronic media can offer various possibilities for interactive usage. These include interactive navigation through the content, interactive adjustment of views and presentations and interactive animations. Another form of interactivity results from the integration of software and help systems. As in most modern software systems the help system is available on an electronic medium and closely interconnected with the application, users can interactively switch between help and software, sometimes without recognizing that they just used the help system. Mirel (1998) stresses that interactivity that goes beyond navigation through documents has a very positive influence on the learning performance. Interactively using help system and software helps the user to "understand their computational choices" and build a solid mental model. Jonassen et al. (1993) also showed that interactivity supports learning.

Interactivity is an enabling technology of context-aware help systems. E.g. for user-profile sensitive systems, interactivity is needed to measure and track the user's activities, which are an important part of the user profile. Layering is also mostly achieved with interactive elements.

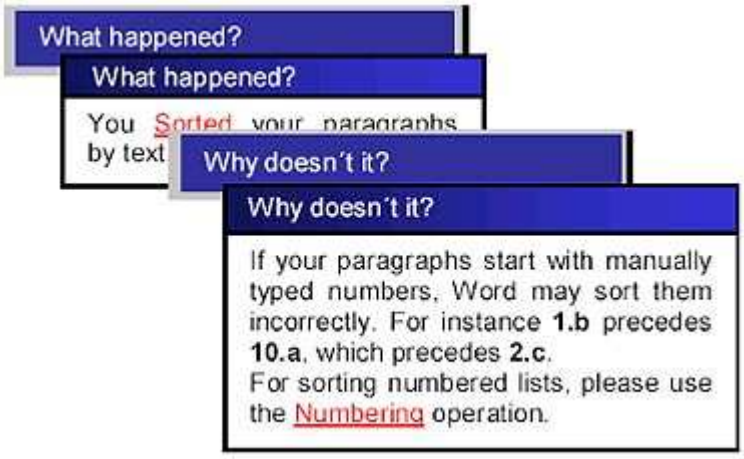

Figure 4: Accessing help via the „Why doesn't it" utterance [Silveira et al. (2001)].

Interactivity is one way to let the user communicate with the system. Some approaches try to provide communication facilities that go beyond those interactive features described above. Silveira et al. (2001) see software systems as "meta-communication artifacts," i.e. they "communicate about communication." Software artifacts are messages from the designer about possible user-system communication. Silveira et al. stress, that this communication is bidirectional. In their approach based on "semiotic engineering" they use standardized utterances to achieve user-intent sensitivity. In a word processing application, the user can express his intention by 15 utterances like "Where is," "What happened," or "What now." These utterances, which are based on the taxonomy of user questions by Baecker et al. (1990) described previously, are triggered either explicitly by clicking a button or implicitly when the user acts in a way that makes a certain intention likely. For example, when the user navigates through menus and toolbars for a certain time, the utterance "Where is" is triggered. The utterances are meant to let users express their intention when accessing help and therefore are an example to use interactivity to realize user-intent sensitivity.

This approach is one way to use feedback to improve the usability of a help system in use. Another interesting approach is to use feedback to improve the design of the next version of a help system. If many users express their opinion about the help system and even state what they like and what could be improved, this information can be used to periodically improve the help system in design-redesign cycles.

2.2.4.6 Embedded Help and Self-Explanatory Objects

The first major change for help systems for software was to go from paper to the computer screen, often still as a separate set of documents. Later, most help systems were linked closely to the software system. The next logical step was to embed the help system completely into the application. Most modern software systems have parts of their assistive information integrated into their system's GUI. Examples for this *embedded help* are wizards, tool tips (also called balloon help) and status bars [Microsoft (1995), Microsoft (2001b), Apple Computer (1996)]. While many companies start to see their documentation as a product, Rettig (1991) suggests to see the "product as documentation." Gery (1995) calls embedded help *intrinsic support* and describes it as being "so integrated into the interface structure […] that it is impossible to differentiate it from the system itself."

Another way of making objects self-explanatory other than embedding documentation into the objects is to design them such that everybody can use the respective feature without help. While there have been some positive examples for this, the increase in complexity of many systems, shown for example in Figure 3, suggests that this is only possible to a certain extent.

This approach, when seen as one part of a hybrid solution to build highly usable help systems, is quite promising for help systems for software. When considering help systems for physical devices, it is not always possible to embed help content into the device. The device has to have a display and the computational power to present the information and still, all information would be limited to some kind of display and not really embedded into the control elements. A tool tip showing personalized help integrated into a button on your washing machine is not likely to be realized in the near future. The approach that is presented in this thesis takes an alternative approach: It does not embed the help system into the device but the device's interface into the help system. This kind of help system runs on a handheld computer and integrates ways to remote control the device into the help pages.

2.2.5 Summary

The strategies that were described above are successful ways to improve the usability of a help system. While minimalism can be applied to all kinds of help systems, active help, context-aware help, interactive help and embedded help are related to help sys-

tems for software so far. However, not only minimalism, but also active, context-aware and interactive help are strategies that can also be applied to help systems for physical devices, provided that the help system is given on an electronic device and that it can communicate with the physical device in some way.. Therefore, using handheld computers for mobile context-aware help systems is a promising approach that can help to realize some of the strategies described above – of course without forgetting the problems and success factors that were identified. The following chapter describes the challenges and opportunities that arise when using such handheld computers.

2.3 Mobile Human-Computer Interaction

The previous two parts of the State of the Art-section concentrated on describing context-awareness and intelligent help systems as a theoretical foundation of a context-aware help system. However, they didn't consider the great challenges and possibilities of the fact that the context-aware help system described in this thesis is *mobile*. This chapter describes the basic terminology of mobile devices, stresses the new possibilities of using mobile devices, and analyses the challenges that have to be met when designing the UI and content for a mobile device. Furthermore, results from recent usability studies with mobile devices and design guidelines based on these studies are described. These guidelines influenced the design of the UI and content of the iManual prototype.

Human-Computer Interaction (HCI) in general and mobile HCI as a subsection is a large field of research, so this chapters is not intended to give an overview of the state of the art in HCI or mobile HCI but to analyze the possibilities and challenges in mobile HCI that are relevant to a mobile context-aware help system.

2.3.1 Terminology

Mobile devices nowadays come in various sizes, shapes, and looks and are equipped with a huge range of functions. Vendors use several terms, often influenced by their marketing and sales departments. Weiss (2002) defines three characteristics for a *handheld device*: It must operate without cables (except temporarily), must be easily used while in one's hands, and must allow the addition of applications *or* support Internet connectivity. *Laptops* and *palmtops* (which look like laptops but are significantly smaller and often fit into a pocket) are not classified as handhelds because they are not operated in one's hands. Handheld devices can be subclassified into Personal Digital

Assistants (PDAs), pagers, and mobile phones as shown in Figure 5. These classes are overlapping and *communicators* are described to combine the characteristics of all three classes. PDAs typically fit into a shirt pocket, have larger displays than mobile phones, have touch screens, offer input via stylus or some hardware keys and, unlike pagers, can be extended with software and sometimes hardware in various ways. Some PDAs offer Internet access and Bluetooth™ communication.

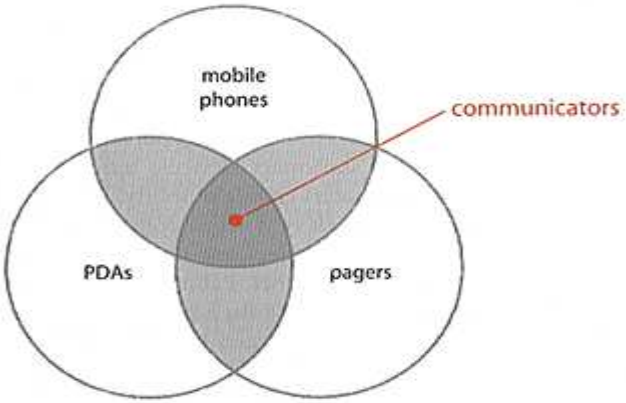

Figure 5: Handheld devices [Weiss (2002)].

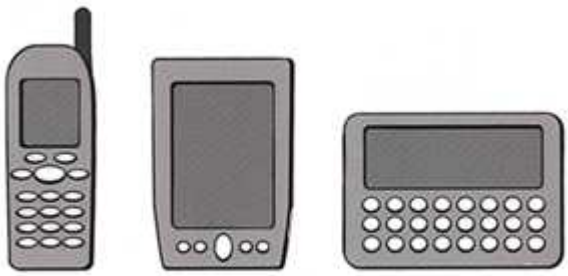

Figure 6: Handheld device types: mobile phone, pager and PDA [Weiss (2002)].

The PDA used for the iManual prototype is a Siemens Fujitsu Pocket LOOK 600 [Fujitsu Siemens Computers (2004)] with a 400Mhz processor, 64MB RAM, a 240x320 pixel color display, battery for 12 hours operating time, integrated Bluetooth™ hardware and an external WLAN card.

Figure 7: Fujitsu Siemens Pocket LOOX PDA [Fujitsu Siemens Computers (2004)].

2.3.2 PDAs as an Enabling Technology

Transferring a help system for a device from the traditional medium of paper to an electronic handheld device brings a lot of new possibilities. Many of these are described in other chapters of this thesis because the whole concept of a context-aware mobile help system is not feasible without an electronic device, so this chapter is intended to give an overview.

A PDA is a mobile, network-connected, interactive, and personal device. *Mobility* enables learning and using at the same place. Paper-based manuals tend to be unavailable when they are needed. PDAs, on the other hand, are used for multiple purposes, so they are better available when the user needs help about a certain device. Furthermore, future PDAs can possibly connect to any device and download the contents of its help system either from the device or the Internet, so the PDA will be the single help source of the future holding the manuals of all surrounding devices.

Connectivity to different kind of networks or devices also brings many opportunities for a mobile help system. Today's PDAs are connected to the Internet or local LANs via WLAN, GSM, GPRS or UMTS. Furthermore, they can communicate with other devices via Bluetooth™ or IrDA. Communication with the device means that the UI of the device can be projected onto the PDA. The user therefore can remote control the device using the familiar interface of his PDA [compare Feldbusch et al. (2002), Nichols et al. (2002b), Myers et al. (2002), Zimmermann et al. (2003)]. When using an interface element on the PDA, a corresponding command is send to the device to invoke the appropriate action. Connection to the Internet can be used in many ways. The Internet can for example be used to provide the content of the help system that can be downloaded by

the PDA or the device, which would make it easier for content provider to maintain their content. Furthermore, information of the help system can directly be linked to related web-based content. E.g., a page of a help system explaining how to change tires of a car could provide links to dealers that sell tires or offer a changing service. A third way to exploit the Internet connection is to enable collaboration. For example, Internet-based forums can be offered to let users of the mobile help system share their knowledge and cooperate with each other. If a user could find a way of solving a problem that is not covered by the help system, he could then share the solution in a user forum and many others could use this new information if they run into the same error. Of course, the content provider could also analyze that information to update the help content or – better – improve his product to prevent common errors or identify usability flaws.

Interactivity lets users individually navigate through the help content, mostly supported by hypertext. Connectivity to the device even enables interactivity on another level, i.e. interactive operation of the device *from* the help system.

Personality brings further advantages. Unlike a paper-based manual, a mobile help system can be personalized. The content of the help system and the way it is presented can adapt to the user-model, either because of explicitly stated preferences or implicitly measured user-related information. The PDA will be the user's personal electronic device that accompanies him all day being used for various tasks and storing all kinds of personal information. Thus, the owner gets used to the interface of this device, especially if this interface can be tailored to his needs.

Having a mobile, connected, interactive, and personal electronic device is also the basis of gathering several kinds of contextual information such as location, identity and action that are then used to deliver content that adapts to the current situation.

2.3.3 PDAs as a Design Challenge

2.3.3.1 Key Challenges

Compared to a typical desktop application, a system running on a PDA has to cope with some basic disadvantages: The PDA has less computing power, less storage capacity, often an Internet connection with a lower bandwidth, a smaller display and limited input possibilities. Furthermore, these systems have to face a limited operation time because batteries have to be recharged frequently. Looking at the PDA used in the iManual prototype, we already see that the limitations of computing power, storage, connection

bandwidth, and battery time have started to diminish. Today's PDAs reached the CPU performance that desktop computers had some years ago, offer acceptable Internet bandwidth when using WLAN or UMTS and offer batteries that allow more than a business day of operation and more than a week of standby time. Storage capacity is also less relevant because data can be delivered over network connections instead of being stored on the device. While computing power, storage capacity, connection bandwidth, and battery life will probably further improve, display size and limited input facilities will remain the key challenges for PDA-based systems. Being portable and handheld always implies small displays and one-handed input facilities.

Based on these technical limitations, Dunlop et al. (2002) identified five key challenges for HCI designers:

- *Designing for mobility*
- *Designing for a widespread population (e.g. users without formal training)*
- *Designing for limited input/output facilities*
- *Designing for (incomplete and varying) context information*
- *Designing for users multitasking at levels unfamiliar to most desktop users*

To cope with these challenges, various authors set up design guidelines. These guidelines are based on the findings of usability studies in mobile HCI. As a basis for the design of the iManual prototype, these findings and design guides are described in the next chapters.

2.3.3.2 Empirical Findings

A lot of research in the mobile HCI field has dealt with the limitations of handheld devices. This chapter concentrates on the influences of the small display and the limited input facilities on reading speed, comprehension and typical ways to access information. These are seen as key influences that have to be considered in the design of a context-aware mobile help system.

Early works showed that users generally are able to read and understand information even on very small displays [Dillon et al. (1990), Duchnicky et al. (1983)]. However, Nielsen (2000) reports that reading from a screen on a desktop is about 25% slower than on paper, while others showed that reading from a small display is even slower. It was also found that display height and width play different roles. A *height* of about four

lines seemed to be sufficient because speed and comprehension did not improve significantly on higher screens. On the other hand, a greater *width* increases reading speed, as reading on a full width display was 25% faster than on a display with one third of the full width. Laarni (2002) presents a study that lets a display width of about ten centimeters seem to be optimal for reading speed and comprehension, which is significantly above the typical width of a PDA. This all suggests that content on a PDA has to be designed especially careful to overcome the disadvantages of the limited width.

Help systems on electronic devices almost always use hypertext to present their help. For hypertext-based systems, not only the ability to read and understand text is important but also the way the user navigates through the hypertext system. This also includes intra-page navigation such as scrolling and following links between parts of the same page.

Jones et al. (1999) tested users in searching for certain information in the Internet with two displays that differed in width and height. Not surprisingly, these tests showed more scrolling to the left and down on the smaller displays. This is remarkable because other studies showed that even on desktop systems, scrolling can make up significant portions of overall browsing time (Byrne et al. (1999) measured 40 minutes of scrolling in five hour tests). Even if users with smaller displays tend to more scrolling, they still have more trouble finding information located in the middle or at the end of a page. Kim et al. (2001) showed worse results for smaller display users on pages of 275 or more words. However, Jones also showed that the number of clicks on links between pages were equal on both displays, indicating that small display users are not browsing through more information to find their answers and that they do not get lost while surfing. What differs is the way users found the information they needed: Users with bigger displays tend to follow longer paths of hyperlinks while small screen users use the search facility more frequently and return to that facility sooner. All in all, large screen users were 50% better in solving information retrieval tasks in the Internet in Jones' tests.

2.3.3.3 User Interface Design Guidelines

When designing a text-based system intended for a mobile device, both the design of the UI of the mobile system and the text itself is important to ensure usability. This chapter concentrates on guidelines for UI design that are relevant for a context-aware mobile help system while the next chapter describes how text should be written for mobile devices.

As stated above, the key challenge for UI design for handheld devices is a rather small display. However, as uidesign.net (2000) put it, “the need for a big screen is simply a poor excuse for bad design.” Reeves et al. (1996) states that a large screen can even impede the usability of an educational or instructional system because a larger screen also leads to more arousal.

A first thing that has to be considered is a different way people access information on handheld devices than on desktop systems, particularly for mobile help systems. As mentioned above, users of small screens don’t like to follow long paths of hyperlinks and use the search facility more frequently. Generally, mobile users appreciate ways of direct access to information [Billsus et al. (2002)]. Therefore, mobile help systems have to offer several ways of accessing the mobile help content, such as search engines, structured tables of contents, indices, links of related content on each page, and links to helpful content according to the user’s situation.

Along with providing multiple access to information, a handheld system should provide a well-designed, consistent way of navigating through the system [Microsoft (2003b)]. Using consistent and familiar concepts is especially important in mobile help systems because providing access to information is the main purpose of such a system [compare Uther (2002)]. Some aspects of a design concept called *Inductive User Interface* (IUI) can also be useful for mobile help systems [Microsoft (2001a)]. This concepts intends to make UIs easier to understand by “breaking features into screens or pages” that are easy to learn. This includes having a single screen per task, stating the task explicitly in the title and making the next step obvious from each page. These design principles can also be applied on mobile help systems.

Several authors also stress the importance of handheld, as well as desktop systems, to enable interactive discovery [Microsoft (2003b)]. This was also part of the minimalist approach to writing manuals that was described in chapter 2.2.4.2. Interactive discovery is encouraged by guaranteeing *forgiveness*, i.e. by making actions reversible or recoverable [Weiss (2002), Billsus et al. (2002), Microsoft (2003b)]. Furthermore, to support interactive discovery, hyperlinks should be self-descriptive. As already said, Jones showed that handheld device users don’t like to follow long paths of hyperlinks, while Kim et al. (2001) states that users are “very reluctant to click a link” because they try to anticipate the content of the linked page and only click if they are “fairly certain that they will discover what they are looking for.” Therefore, a link should explain as exactly as possible what the link leads to. The fact that interactive exploration has a posi-

tive effect on learning was shown by Wulf (2000). He implemented “explorations environments” that could be used to interactively explore the functions of three tailorable groupware tools. Although this test was not related to a mobile environment, it shows that exploration supports self-directed learning and should therefore also be supported by mobile as well as stationary UIs that want to let their users learn new concepts of their interface quickly.

The finding that mobile device users spend much time on scrolling and have more trouble finding information located in the middle or at the bottom of a page suggests that pages should be designed to reduce or avoid scrolling, especially horizontal.

The layout of the UI and the pages of a help system are also important to designing a usable interface. Icons are a way to place more interface elements on a small display than with text, so they can be a helpful way to design handheld UIs [Marsden et al. (2001)]. Marcus et al. (2002) showed how the use of icons can help to design a more personal device and to let “the user know what is happening on the device in a constant, noticeable, but non-obtrusive manner” (see Figure 8 and Figure 9).

However, icons and graphics have to be used carefully because they also lead to more arousal and might be hard to grasp and to remember for new users. To make icons usable and easy to remember, they should be intuitive and therefore they should be carefully designed and tested with typical users [Nielsen et al. (1994)]. Graphics in hypertext pages should only be used if they complement the written text, regardless if the system if mobile or not [Morkes et al. (1997)].

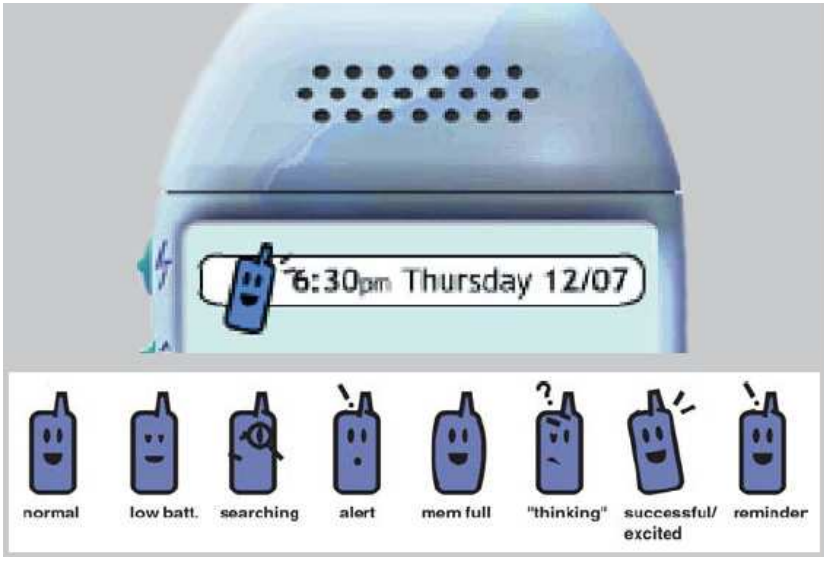

Figure 8: Using icons in a constant, noticeable, non-obtrusive manner [Marcus et al. (2002)].

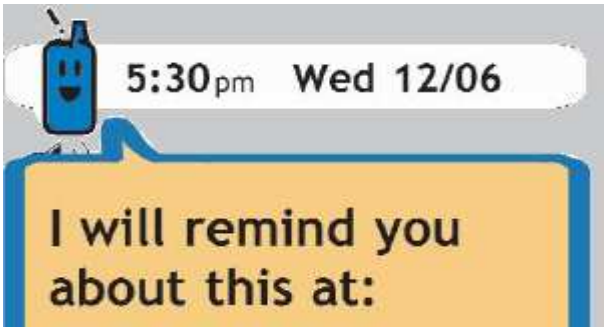

Figure 9: Using icons to create a personal UI [Marcus et al. (2002)].

A possible way to learn from the tests that limited height is not as important as limited width could be to flip the PDA so that the text can be shown having a bigger width in exchange for a smaller height. This applies to most of today's PDAs that typically have a display with a bigger height than width.

Buyukkokten et al. (2000a) suggest a way to cope with the problem that it is harder to get an overall image of the information when using a small screen device. They implemented a system that first shows a "link view" that presents an automatically generated overview of all pages of a web site. The link view just contains a hierarchical view of all the links on the pages of a site. The same system also offers two further approaches to increase handheld usability: a local search facility that allows searching for keywords on a particular site and a support for text entry that suggests keywords when only few characters are typed based on recent input [Buyukkokten et al. (2000b)]. These features resulted in a 45% gain in browsing speed and a significant reduction of stylus movements. Hence, if usability tests indicated that users of a mobile help system had similar trouble, such an approach could be considered.

A lot of authors also worked on ways to cope with the limited input facilities on handheld devices. This is not central for a mobile help system, because it is primarily used to retrieve information. However, a context-aware help system on a PDA can also be used to operate the device as a kind of remote control. Therefore, the user does not only read information on the PDA but use the PDA's input facilities to operate the device.

Central to all works on limited input facilities is the claim for as few input as possible [Uther (2002)]. Because one-handed interaction with a handheld device is always more cumbersome than using mouse and keyboard on a desktop, the mobile application has to provide ways to reduce input. This can be supported by context-aware computing, so the user does not have to provide data that the system also can obtain from the current

context [Billsus et al. (2002)]. In the iManual prototype described later, the user can input a target into the navigation system of his car using his PDA. The user can either use his stylus to type data such as city, street and number and send it to the car or – better – use one of the addresses of the PDA's address book.

Another way of coping with limited input facilities of handheld computers is to enable voice control and audio output. This is especially relevant for using computerized information systems in cars [ATX North America (2003b)]. For security reasons, the driver of the car cannot read from a PDA's screen or use the stylus to tap on buttons. The iManual system presented in this thesis did not implement an audio interface. This work concentrates on exploiting contextual information to improve a mobile help system and integrating remote control elements into the PDA's existing interface, while leaving research about creating multimodal interfaces for mobile scenarios to other groups [e.g. Nichols et al. (2002a), Haniff et al. (1999), Kirste et al. (2001)].

2.3.3.4 Text Production Guidelines

Guidelines for writing text have been set up by many authors. Most of the findings from research on how to write hypertext for systems that use larger screens can also be transferred to handheld scenarios.

All works on how to write hypertexts that will be read on any size of screen have one guideline in common: "keep it short" [Nielsen (2000), Uther (2002), Microsoft (2003b), Koyani et al. (2003), Kim et al. (2001)]. Reading on screens is harder for human eyes and mostly also slower, especially on small screens. Furthermore, it can often be interrupted in mobile scenarios, readers often do several things in parallel and small screens can only show small chunks of content at the same time. Therefore, having a short and compact representation of the content will increase reading usability. The need for short pages that don't demand scrolling is supported by hypertext, which allows splitting longer texts into several pages linked together. Layering, which has been described in chapter 2.2.4.2, is a technique that users hypertext to split longer units into several parts, allowing the reader to skip content that he probably already knows. However, Kim et al. (2001) warns of "fragmenting ideas to smaller-than-meaningful grains."

For similar reasons than listed above, many authors advise writers to design their texts for scannability [Nielsen (2000), Morkes et al. (1997), Koyani et al. (2003), Spyridakis (2000)]. Nielsen reports that 79% of web-page users scan a web page rather than reading it word-by-word. Tests on handheld displays would probably lead to similar results.

Scannability can be supported by careful layout and appropriate text, e.g. by short, meaningful and well placed headings, short and readable paragraphs, bullet lists that break the flow of uniform text blocks and various other kinds of text highlighting. Furthermore, a writing style that puts the primary theme in the first sentence of each paragraph improves scannability. Well-formulated headings are also important because they are often used as search results and therefore have to provide a good idea of what the paragraph is about [Russell (2001)].

There are many other guidelines that several authors have published about writing text for hypertext systems, e.g. avoiding unfamiliar acronyms, using words that are easily understandable by typical users, or avoiding long prose text on navigational pages [Koyani et al. (2003)]. Most of them are rather obvious and therefore not described further here.

2.4 IT in the Automotive Domain

This thesis analyzes the concept of context-aware mobile help system. The iManual prototype developed along with the thesis uses the automotive domain to show this concept but could the easily transferred to any other domain. The prototype contains the content of a car's manual and especially concentrates on help about using the navigation system. The iManual concept uses a product that is mostly used by consumers because this makes the application easier to understand. However, the concept is also very promising in the industrial domain where workers have to learn and know how to operate various complex machines. Using a context-aware mobile help system is especially useful in the automotive domain for several reasons. Cars are complex devices with more and more features. Controlling and using these features is getting more and more difficult for users, especially when they use that kind of car for the first time. Therefore, manuals for cars get quite large[1]. The need for an intelligent presentation of the assistive information increases with the number of features and size of the manual. Second, getting information about the status of the car as a part of the context is quite promising because modern cars are equipped with thousands of sensors[2]. Third, a car is a relatively expensive device and for most consumers the most or second most expensive invest-

[1] E.g. the manual for the BMW 7 has 323 pages [BMW AG (2002)]

[2] E.g. to connect the various sensors and electronic devices, Volkswagen's Phaeton includes 3860m of wires and 2110 trims [c't (2003)].

ment (after their house) they make in they life. The high price leads to a low relative price of the PDA compared to the car. A low relative price makes it easier for manufacturers to equip the product with a PDA-based mobile help system. Fourth, some cars already contain systems that use PDAs, as described as follows.

After this section, section 3 describes the concept of a context-aware help system. It does not consider the specific opportunities and challenges of the automotive domain. The subsequent section 4 describes the prototypical implementation of the iManual-system and *does* consider the specifics of the chosen domain.

2.4.1 Computerized Systems in Cars

Modern cars already contain various IT-supported systems. Some of these can be used via PDA, either by using it as the main input and output device, as an alternative output device, or by being able to synchronize data in the car with the PDA. Therefore, a mobile help system can use that existing interface to connect the PDA-based system to the car in order to reduce costs. Typical IT system that are used in modern cars are:

- **Information Portals**: Features range from access to the address book, tasks or appointments to Internet connectivity, e.g. to check e-mails or news. Navigation systems, which many of today's cars already contain, are also integrated into these systems. Obviously, new UIs with voice control or easy-to-use buttons and audio feedback are needed. More and more people today spend an increasing part of their day in their cars, e.g. sales persons or commuters, so these systems bring an added value for those users.
- **Security and Emergency Systems**: If equipped with appropriate sensors, a vehicle can detect crashes automatically, e.g. when the air-bag is released. This is also called *Automatic Crash Notification (ACN).* The car or the connected PDA then send an emergency call automatically. Additional information such as the car's current position, indicators for the intensity of the crash, and even personal medical data (if saved on the PDA previously) can be transferred [Hansmann (2001)].
- **Remote diagnostics**: Without being physically present in a garage, a car can measure maintenance-relevant data and send this to the dealer's garage or service center [Hansmann (2001)]. The service center can use this information to suggest maintenance appointments only when necessary. The garage can use it

to allocate personnel and order spare parts. Future cars could even repair simple defects remotely by the cars electronics.

- **In-Car Entertainment (ICE):** Various entertainment applications are included in modern cars, ranging from digital radio to DVD players. Many can be supported by PDAs, for example the car audio system can play MP3 songs from the PDA or passengers can use the handheld to play "backseat games" [Brunnberg (2002)]. Systems that offer ICE, remote diagnostics, security and emergency features, and access to information portals are often summarized as *telematics* [Telematics Research Group (2003b), Duri et al. (2002), Hansmann (2001)].

- **Personalized Cars**: Modern cars adapt to its user when they detect the user's personal key, e.g. by adjusting the seats and mirrors to the driver and by loading favorite music. This is interesting for cars that are shared by more than one driver (e.g. rental cars) or drivers who drive a lot of different cars. The identification can of course also be based on the user's PDA instead of the key.

- **Vehicle Relationship Management (VRM)**: VRM systems transfer relevant data to the car manufacturer, so he can build a database about status, errors and breakdowns of his cars and the way drivers interact with the vehicle. He can then analyze where errors occurred quite frequently, which reasons led to breakdowns very often etc. The more information the car manufacturer gets about his customers and his cars, the better are his possibilities to use this information for marketing, market research or CRM. Via different communication channels the manufacturer can contact his customers to inform them about new car equipment, special offers, or necessary car inspections. This can generate additional revenue for the manufacturer and enhance customer loyalty [Wheeler (2002)].

- **Inter-Vehicle Communication**: Many more possible applications arise when using communication *between* vehicles. When all vehicles on the road automatically communicate with each other, these networks can be used to prevent traffic jams, accidents etc. [Fraunhofer FOKUS (2003), FleetNet (2003), Huber et al. (2001),Hansmann (2001)].

2.4.2 Existing Products

Car manufacturers, suppliers, and research institutions are developing systems or investigating possibilities in most of these fields, while in some areas there are already sev-

eral products on the market. Most existing products cover several of the functions described above.

GM Corporation (2003b) sells 110 000 cars equipped with its telematics system „OnStar“ per month and has more than two million subscribers to that service [Telematics Research Group (2003a), GM Corporation (2003a)]. OnStar offers remote diagnostics, emergency and security services, as well as access to traffic or routing information. It consists of three buttons on the steering wheel and a call center that handles the service requests, so most outgoing connections are simply established via a mobile phone call. Usage of OnStar is subscription-based, so customers pay a monthly fee for their service.

Daimler-Chrysler (2003)’s system „TeleAid“ also offers remote diagnostics and emergency services using a car-phone connection. Unlike OnStar, it does not use an own call center but sends data directly to the emergency call center or the dealer’s garage.

BMW AG (2003b), Audi AG (2003) and Volkswagen AG (2003) offer or develop similar services.

Information portals are for example offered by BMW AG (2003a) and Daimler-Chrysler (2002). BMW uses built-in displays (with PDA synchronization) for displaying information while Daimler-Chrysler offers docking stations for PDAs and mobile phones or stationary displays alternatively.

Some component suppliers and technology companies also develop systems that major car manufacturers use in their systems [Hirschmann (2003), MobileAria Inc. (2003), Siemens (2002), ATX North America (2003a), Siemens (2002)]. ATX North America (2003b), a supplier working for BMW, Daimler-Chrysler and Jaguar, has announced to offer a help system application called “vehicles owner’s manual.” This completely audio-based system can be used while driving and can even suggest inspection appointments when defects are detected.

2.4.3 Using PDA-based Help Systems in Cars

A context aware mobile help system for a car could either be used on a mobile device (e.g. PDA) or on a built-in display in the car. However, using the PDA has several advantages:

A PDA is a personal device that accompanies its user the whole day. One PDA can be used to hold help systems for many surrounding devices including the car. The help content could either be stored on the PDA, in the Internet, or on each device and be

transferred to the PDA when used. Using a PDA for the prototype is reasonable because this stresses the fact that a single mobile device can be used to offer the help system for various physical devices.

Secondly, PDAs are personal devices. A driver can use his PDA in many cars, and a car can be driven by many users with different PDAs. In this scenario the PDA can be used to identify the user, which is an important aspect of the scenario's context.

Thirdly, a PDA can be handled faster and better than a car's buit-in display and control devices. Nichols (2001) showed that test users controlled different devices more successfully using a PDA than directly. They were faster in fulfilling certain tasks and made less mistakes. The reason for that could be that PDAs are personal devices that are used much more frequently than the controlled devices. Hence, the PDA's UI is more familiar than the UIs of those devices.

2.5 Conclusions - Going Beyond the State of the Art

This section described the state of the art relevant for the concept of context-aware mobile help systems. It provided a classification of context dimensions and context-aware computing systems, and it showed that most existing context-aware systems concentrate on other dimensions and fall into other categories than a context-aware mobile help system.

It then classified and evaluated strategies to create better help systems. It was shown in empirical studies that several strategies successfully improved help systems, mostly however for software and not for physical devices. The following sections will show that many of these successful strategies can also be applied to help systems for devices.

Based on the disadvantages of a handheld computer such as small screen size and limited input facilities, this section listed relevant guidelines for designing UIs and writing texts for handhelds. These guidelines were considered in the iManual system that is presented in section 4.

Finally, this chapter justified why the automotive domain that the iManual prototype was implemented for is a promising domain for a context-aware mobile help system.

3 Context-Aware Mobile Help Systems - the Concept

This section presents the concept of context-aware mobile help systems. The following section then shows how this concept was put into practice by developing the iManual system. In this section, I will describe the most important opportunities the concept brings. Furthermore, I will mention the challenges that have to be met when designing such a system. It will be described in which way such a system can provide context-aware help by analyzing different contextual parameters. Then, I will describe how strategies of machine learning can be used to cope with the multitude of contextual constellations and to learn from previous experiences to improve future reactions. Then I will show that the concept of context-aware mobile help systems leads to a new learning situation that integrates learning and using. Furthermore, I will mention a main challenge to develop context-aware mobile help system, i.e. to organize the help content so it can adapt according to the context. Finally, I will summarize chances and challenges of context-aware mobile help systems.

3.1 Providing Context-Aware Help

As said in chapter 2.1.2.1, the context dimensions of identity, environment or action, and location are particularly relevant for context-aware mobile help systems. Relevant aspects of these dimensions that were identified above include

- the internal state of the device that the help system is provided for
- recent and current interaction with that device
- recent and current interaction with the help system
- the user's knowledge and level of expertise
- the user's experiences
- the user's preferences
- the geographic location
- the relative location of the handheld to the device

A context-aware mobile help system adapts the presentation and the content of the given help to these contextual parameters. The next chapters show in which way the help system considers these parameters.

3.1.1 Integrated Help

Integrated help in the context of this section does not mean that the given help is physically integrated into the device but that the device and the help system are connected by a wireless link and are able to use this link to transfer contextual information. Based on that wireless communication, the device constantly sends relevant contextual data to the help system that is measured by one of its sensors. This includes changes of the internal state of the device and monitored user interaction. State changes can include defects and errors that are measured by the device's sensors, internal states that have changed because of user interaction (e.g. the on/off-state) and any other internal values that can be measured and are relevant for a help system. Therefore, the help system is constantly kept up-to-date about relevant contextual parameters related to the device. The new learning situation that is enabled by the ability to exchange contextual information and to send control commands from the help system to the device will be described in chapter 3.3. Knowledge about the internal state of the device can improve the provided help in several ways. For example, descriptions of interaction options can be filtered out if they are not available in the current state of the device or urgent information related to defects or errors can be shown if they are detected by the device. Adapting the given help to the current or recent user interaction with the device will be described in the following chapter because providing personalized help is based on monitoring user interaction.

3.1.2 Personalized Help

A context-aware mobile help system takes contextual information about the individual users into account and therefore provides personalized help. This means that the presentation of the help and the help content itself is adapted to the user's knowledge, level of expertise, experiences, preferences and his interaction history. This chapter describes how two special kinds of context-awareness that adapt to the users' individual needs can be achieved: user-profile sensitivity and user-intent-sensitivity.

3.1.2.1 User-Profile Sensitivity

As described in chapter 2.2.4.4 according to Silveira et al. (2001), user-profile sensitive help is a system that presents information tailored to a user profile. A user profile for a context-aware help system is build combining some of the context-parameters just described, i.e. the user's previous interaction with the help system and with the device and his explicitly stated preferences. From this user-profile, the user's knowledge, his expertise, and his implicit preferences and wishes can be inferred. Given that the help system has obtained sufficient information about the user and given that this information allows reasonable inferences, the help system can provide valuable help tailored to the user's needs.

For example, when the user demands information about a certain topic, by clicking on a link or searching for a keyword, the content can be adjusted considering the user profile: If the profile indicates that the user has detailed knowledge about the topic he requests, the presented help can be short and compact. If it indicates that he prefers procedural instead of declarative help, the declarative parts of the content can be left out or hidden behind links. If recent interaction shows that a user has significant trouble with a certain topic because he repeatedly requests help about it, the help system could offer alternative representations or related information.

3.1.2.2 User-Intent Sensitivity

As already said, user-intent sensitivity is difficult to realize. No software system can precisely predict the future, so determining what the user plans to do is hard. Silveira et al. (2001) addressed the problem by allowing the user to express his intention by choosing from a given set of typical utterances. The context-aware help system described in this thesis take a different approach. In order to determine the user's intention, a user profile as described in the previous chapter is used. The current interaction with the device and the help system, the user's preferences, which are taken from the user profile, as well as other contextual parameters such as the internal state of the device are used to anticipate which task the user might plan to do and which help he might need.

For example, similar to the dynamic help feature in Microsoft (2003a) (see Figure 3), a context-aware help system can constantly try to anticipate the user's intention and show appropriate help. The iManual-System described in this thesis offers a contextual links-feature that is accessible from anywhere in the help system. The feature uses contextual information to anticipate the user's intention and machine learning strategies to evaluate

this information, which results in showing a number of links that offer assistance to the anticipated task.

3.1.3 Location-Based Help

The help that a context-aware help system provides can also be adapted to the current location. As said before, location can either mean the geographic location of the help system or its relative location to the device. Geographic location can improve the help if the device is used differently in different locations or if the functions of the device itself are dependent on the location. For example, if the navigation system of a car computes a route that leads to another country, the help system could offer additional information about driving in that country.

The relative position of the handheld help system to the device can be exploited to offer help about particular parts of the device that the help system is pointed at. If the mobile help system of a large industrial machine is pointed at a particular element, the help system could offer assistive information about that element. The LISTEN system that will be described in chapter 3.2.2.4 is based on a similar concept. It determines the user's position in a room and his head orientation to give context-aware information about the surrounding objects.

The following chapter describes a characteristic of context-aware applications that is also relevant to context-aware mobile help systems: context fusion. This means that contextual parameters can often be obtained or combined from several sources on several logical layers.

3.1.4 Contextual Information on Different Layers – Context Fusion

As in most context-aware applications, in a context-aware help system contextual information is obtained from various sources. These sources can be integrated into the device or be part of the handheld system. To determine contextual parameters that can be used in context-aware systems, values from different sources are aggregated on several layers.

Imagine a context-aware help system for a car. The car will be equipped with various sensors. One sensor will measure the liquid level in the gas tank. Based on this "sensor layer," this information is enriched by a semantical meaning on several layers. A low gasoline level on a more semantical level means that the car can only drive a certain

distance. This distance is transferred to the help system as a part of the internal state of the car. If the car also transfers that the user just defined a new target in the navigation system that is not within this distance, this interaction information can be combined with the distance information to gain another meaning on a higher semantical level. Analyzing this semantical level, the help system should show an advice that the driver has to refuel on the way if he wants to reach the target. Along with that, the help system can show help on how the route in the navigation system can be changed to drive to a nearby gas station. Additionally, the help system could determine these gas stations itself.

All in all, a context-aware help system uses sensor data from various sources that are aggregated on several layers to enrich that data. Zimmermann (2003) describes an advanced framework with several different layers that can be used to model these different layers of a context-aware application.

3.2 Learning to Improve Future Help

The context-aware mobile help system described in this thesis intends to give context-aware help and thus enhance human competence by taking advantage of a mobile, personal, and network-connected device such as a PDA. The mobile help system has to implement a strategy that allows it to react to the multitude of possible situations accordingly. The PDA as a personal device allows monitoring the user's activities with the device and the iManual system. This information can be part of an explicit or implicit user-model. As the user interacts with the system, the user-model might constantly change while the help system has to keep delivering the appropriate kind of help in these changing situations. The ongoing interaction does not only make context-aware reaction more complex but also brings the opportunity to improve future reactions based on analyzing interaction of the past.

This chapter addresses this challenge – a multitude of possible situations to react to – by exhausting the opportunity to improve the systems based on analyzing previous user interaction.

Possible ways of meeting the challenge are using automated reasoning and knowledge based learning systems, which are treated in the scientific field of *machine learning*. Machine learning studies "computational methods for acquiring new knowledge, new skills and new ways to organize existing knowledge" [Michalski et al. (1983)].

After describing several strategies of machine learning, I will describe which strategy fits best to the specific requirements of the scenario of a context-aware mobile help system and explain why it was chosen.

3.2.1 Strategies of Machine Learning

Learning in a knowledge based learning system „denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks (...) more efficiently and more effectively the next time" [Simon (1983)].

Research identified three general kinds of learning processes in machine learning: synthetic (also called inductive) learning, analogy-based learning and analytic (deductive) learning [Richter (1992), Bergmann (2002)]:

- *Synthetic learning* strategies generate knowledge by looking for general rules in their observations and inferring conclusions from that. These conclusions are hypothetic because they are only based on observations. Strategies that use synthetic learning are e.g. learning of decision trees, predicate logic systems, or Neuronal Networks.
- *Analytic learning* strategies extend their knowledge by making deductive conclusions. They try to combine existing knowledge with observed information to infer new rules. Therefore, they only transfer implicit knowledge to explicit knowledge and cannot generate knowledge that is not related to the existing knowledge base. An example of a method using analytic learning is Explanation-Based Learning (EBL) [Choi (1993)].
- *Analogy-based learning* strategies use similarities to previous experience to solve present problems. Solutions of previous problems are transferred to existing problems making analogy-based conclusions. Unlike synthetic strategies, analogy-based strategies do not generalize what they observe to build new knowledge. Instead, learning just means storing current experience for future use. Case-Based Reasoning (CBR) is an example of this kind of learning strategy.

Comparing these three strategies, as can be seen in Figure 10, shows that the amount of knowledge that possibly can be learned increases from analytical over analogy-based to synthetic learning strategies while the confidence in the reliability of the inferred data decreases.

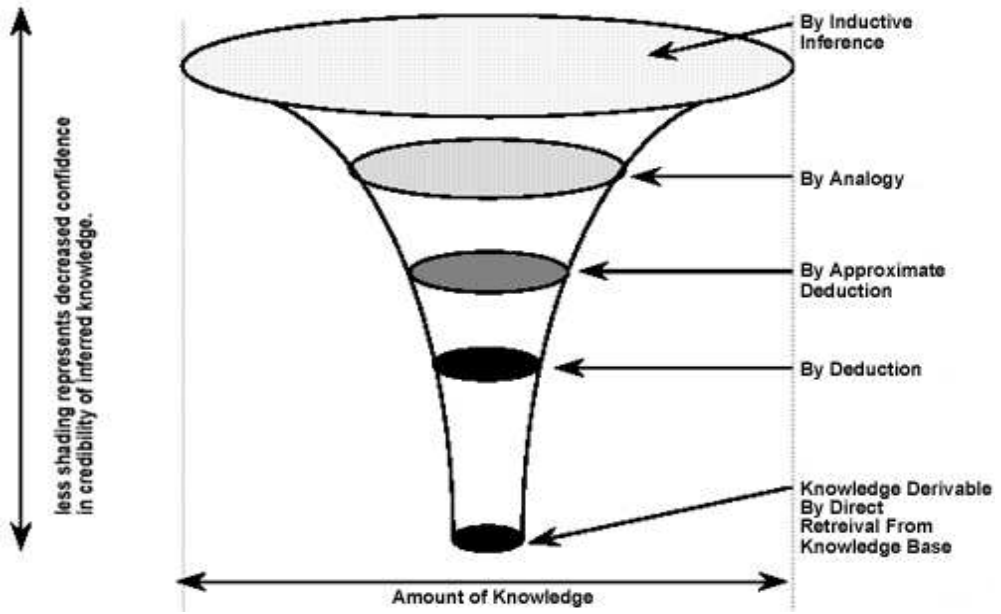

Figure 10: Comparison of strategies of machine learning [Bergmann (2002)] (translated from German source by the author).

While the term machine learning focuses on building and extending a knowledge base that can be used for problems, the term *automated reasoning* does not concentrate on the ability to learn, but on automatic problem solving. However, many strategies include features of both learning systems and automated reasoning. Köster (2000) describes automated reasoning as analyzing a knowledge base to draw conclusions and to solve problems automatically. He lists CBR, Rule-Based Reasoning (RBR), Constraint-techniques and non-classical techniques such as Fuzzy Logic as examples.

The following chapter describes CBR and why it was used in the prototype of a context-aware mobile help system. To underline that CBR is a promising strategy in the scenario of this thesis as well as in many other situations, its usage in help systems is described, and three existing systems using CBR are analyzed.

3.2.2 Using Case Based-Reasoning in Context-Aware Help Systems

Strategies of machine learning and automated reasoning are one possible way to realize context-aware assistance in mobile and network-connected scenarios. Among the various strategies listed in the previous chapter, *Case-Base Reasoning (CBR)* was chosen to be used in the iManual prototype. How it was implemented and integrated into the iManual system is described in chapter 4.4.4.

3.2.2.1 What is CBR?

CBR is an analogy-based learning technique. It is rooted in early works of Schank (1982), Kolodner (1983) and Lebowitz (1983). Existing knowledge from a domain is used to solve a similar problem from the same domain. The problem is solved based on similarities to previous problems. The knowledge base in CBR is dynamic because new problems and their solutions are stored and influence how later problems are solved.

A CBR system consists of *cases*, i.e. a representation of a problem and its solution, a *case base* to store these cases, and *similarity measures*. These measures are a numerical representation modeling the similarity of two problems. Each case can have several attributes of various types, and *local similarity measures* are defined for each attribute type.

Learning in a CBR system is a four-step process called retrieve, reuse, revive and retain step as shown in Figure 11.

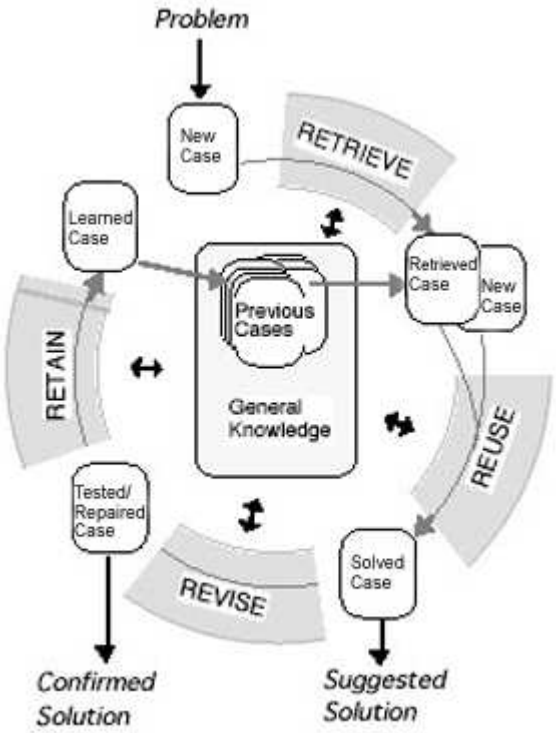

Figure 11: The CBR Cycle [Aamodt et al. (1994)].

The *retrieve step* finds a set of similar cases to the given problem. Similarity is measured by combining the local similarity measures of the problem's attributes. The combination can be done in several ways, e.g. using the Weighted Euclidean Distance or the Hamming Distance. Retrieval can also be implemented in several ways. *Sequential re-*

trieval is a simple way of walking through the complete case base and comparing the given problem to every case. This is an easy-to-implement and exact method. However, as case bases tend to become very large, comparing the given case with every other could be very costly. Ways of dealing with this are a two-step or an index-based retrieval. *Two-step retrieval* uses pre-selection to filter out cases that are probably not similar to the given case. The pre-selection is mostly based on one or few simple attributes, e.g. Boolean attributes. The remaining cases are then compared using a sequential retrieval. *Index-based retrieval* uses indexing and binary search trees to speed up the search for similar cases.

The *reuse step* then transforms the solution of the most similar cases to the given problem. There are several transformation strategies for that, e.g. rule-based, model-based, or user-based transformation. In some cases, no transformation is needed and a solution can be used in similar problem situations without adaption.

The transformed solution is then evaluated in an optional *revise step*.

The *retain step* is responsible for gaining new knowledge. A CBR system can learn in four ways: it can learn new cases, i.e. ways to solve problems, can improve its similarity measuring, reorganize its case base or improve the way of transforming existing solutions. Thus, knowledge in CBR cases is represented in four *knowledge containers*:

- The case base
- The similarity measures (including weighting)
- The case model (attributes of a case etc.)
- The transformation knowledge

CBR has several advantages that make implementing, maintaining and using it relatively easy, such as

- Easy modification and maintenance of the system
- Learning and using periods are not separated
- A complete solution is stored
- The similarity measures can easily be adjusted
- Knowledge is represented in different containers that each can learn.

On the other hand, the following challenges have to be met when realizing a CBR system:

- The case base has to offer a way of storing any kind of problem and solution
- Efficient search algorithms have to be found to ensure acceptable performance
- The most similar case doesn't have to be the one that can be best transformed to the current problem
- There is a dilemma between having as many cases as possible to cover the complete problem domain and having as few as possible cases to allow an efficient retrieval.

Based on these advantages when realizing a CBR system, but also keeping the challenges in mind, the next chapter points out why a case-based strategy was chosen.

3.2.2.2 Why CBR?

CBR was chosen in the iManual scenario because it has some promising advantages in comparison to other strategies of machine learning or automated reasoning such as Rule-Based-Reasoning or Neuronal Networks [compare Leake (1996)]:

- Rule-based strategies have to cover the complete problem domain previous to using the system. The problem domain often is just too large to be covered by such a rule set. Furthermore, setting up these rules could be quite expensive because experts would have to define them. Two examples in the following chapter show scenarios where CRB was chosen because the problem domain was too large to be covered by rules. CBR only handles problems that really occur instead of covering all possible problems. Furthermore, no experts are needed to set up the cases. Instead, the case base grows by learning new cases while the system is in use. In the scenario of a context-aware mobile help system, the problem situation is dependent on the values of various context-parameters that change continuously, which is hard to cover by a rule set. Therefore, CBR is better suited here than a rule-based problem-solving strategy.
- In many fields where strategies of machine learning are used, the problem domain is imperfectly understood at the moment when the system is implemented. Most other strategies require the main modeling of their knowledge to happen before the systems is used, which then leads to imperfect results. CBR on the

other hand offers an increasing quality of solutions during use because it continues to extend its knowledge. Furthermore, it can also learn to handle situations that were not predictable a priori. In the scenario of a context-aware help system, it is hard to predict which kind of help a user needs in which context. Different users will need different help tailored to their personal preferences, and the same user might need different help if the same situation occurs frequently. Therefore, a CBR system is valuable here as it can develop the knowledge to react appropriately in unpredictable situations.

- A static structure, such as a predefined set of rules, is not as flexible as a case base. While changing a rule set requires effort and the presence of an expert, a case base can be changed by the interaction of the user with the system. A user therefore can train a CBR system himself without needing an expert. A CBR system can be initialized with few seed cases and grow dynamically and independently while being used. In the iManual scenario, the CBR system learns from the user's interaction with the system. In the system, we have an ongoing cycle of interaction: The user is in a situation needing assistance, the system delivers context-aware help, the user reacts to that help, and the system tracks the users reaction to learn if its assistance was successful or not. Based on this cycle, the CBR system can learn to improve its future reactions by using past experience.

- A main difference of CBR systems to other inductive or deductive strategies is that it does not generalize what it observes. When a RBR system is initialized with training examples, it generalizes these examples to rules and when it observes a problem situation, it tries to find a general rule for the specific problem. CBR systems, however, store what they observe. They don't store the general rule that can be learned from a case, but the case itself. In the iManual scenario, generalizing cases, i.e. problem situations and its solutions, to general rules would be hard because a solution depends on rich contextual information that cannot serve as a general rule for future problem situation.

- CBR systems do not only learn from reusing previous solutions in current situations, but also from failure. For context-aware help systems, determining that the given help wasn't successful, i.e. didn't provide the assistance the user needed in the given situation, can be at least as helpful for future cases as successful assis-

tance. Most users rather judge a software system by the mistakes it makes than by the situations it provided helpful assistance.

- Another advantage of CBR systems lies in better explainability, and therefore increased understandability and higher user acceptance. A problem solution in CBR is always based on real happenings from the past. Whereas a RBR system could only try to explain parts of its general rule set and Neuronal Nets don't provide explanations of their decisions, a CBR system can relate its behavior to real cases. A context-aware help system could not only give assistance according to the current situation, but also explain why this kind of assistance is given. For example, when the system shows a context-related link "CLICK HERE TO LEARN HOW TO HANDLE TRAFFIC JAMS" in the iManual for a car's navigation system, another link "WHY DOES THIS LINK APPEAR?" could be added where the user would get information about the context-parameters and his previous interaction that the link presentation is based on. Therefore, the intelligent system would not react as a "black box" but explain its behavior to interested users. Modern HCI research identified understandability as a key to increased usability and higher user acceptance in adaptive software systems.

- The chosen CBR system in the iManual scenario also takes advantage of the system being network-connected and therefore allowing large groups of users to cooperate. The system does not only learn from a single user's interaction to improve adaption to that user's needs, but can also store cases from multiple users and therefore make the improvements learned from a single user available to all others. This can be seen as an implicit cooperation. Consider a CBR system that was initialized with a seed case to show a link "HOW TO USE YOUR ADDRESS BOOK TO DEFINE ANOTHER ROUTE TARGET" after the user has just defined his first target in his car's navigation system. If 90% of all users ignored that link and opened topics like "HOW TO START THE ROUTE CALCULATION" or "HOW TO DELETE A TARGET" instead, the system would learn from the other users' behavior and suggest these links, even if the current user is in that situation the first time. To assure personalization and let the system still adapt to the user's individual needs, the CBR retrieval step can be adjusted to give the attributes that identify the current user a higher weight, so that cases of the current user are considered more important when searching for similar cases.

3.2.2.3 CBR in Help Systems

To underline that CBR is a promising strategy to be used in context-aware mobile help systems, this chapters briefly describes typical systems that give assistance in some way and that are often realized using a CBR strategy.

Case-Based Aiding Systems are used to support human reasoners by using case bases. The system assists its user by reminding him of similar previous problem situation, suggesting prior successful solutions and warning in case of previous failures. Thus, they serve as a complement to human experience. These systems perform the retrieval and retain step of the CBR cycle described above, but leave the revise and reuse step to the user. As Leake (1996) adds, "Not only does this interaction provide practical advantages, by avoiding the need for automatic case adaptation and evaluation, but humans readily accept and appreciate the availability of advice."

Case-Based Help Desk Systems are a kind of Case-Based Aiding Systems that are used to support help desk employees. Those use the system as a knowledge base that they consult if they cannot solve a problem immediately and think they might find a similar problem in the case base. An example of a Case-Based Help Desk System will be presented in the next chapter.

Many authors use the term *Case-Based Education* to summarize the use of CBR systems in educational contexts [Leake (1996)]. CBR strategies can be used in many ways in training and teaching, e.g. by making experiences of other learners available when a learner is confronted with a typical problem situation.

Intelligent Tutoring Systems (ITS) are special kinds of educational systems that are not only focused on CBR [Patel et al. (2001), Patel et al. (1998), Martens (2003b)]. Case-based strategies are just one way to implement typical tasks of those systems. An ITS supports the teaching activity of a traditional human teacher. Martens (2003a) summarizes that "an ITS proposes the student a problem, follows him step by step, diagnoses his errors, acknowledging their causes and, if necessary, gives him hints and examples." For giving this form of educational help, the ITS tries to build user-models to analyze human behavior and predict future intentions. In an ITS, these models are the expert knowledge model (also called domain model), the pedagogical model (teaching model), the student model (learner model), and the model of the user interface. To realize these models, an ITS mostly uses strategies of machine learning such as CBR.

Adaptive Hypermedia Systems also use user-models to adapt to individual preferences. These user-models are applied to hypermedia systems to adapt the content and the links to the user's needs. Many Adaptive Hypermedia Systems are used in educational contexts. Most authors distinguish *adaptive navigation support* and *adaptive presentation* [Brusilovsky (1999), Peylo et al. (2000)]. The former helps the learner to find the optimal path through the hypermedia content and includes direct guidance, adaptive link hiding and adaptive link annotation. Direct guidance informs the learner which links on the page leads to the optimal content according to his individual context. Similar goals are also reached be annotating links to indicate if a link is useful in the current context and by hiding links that are not helpful yet according to the user's knowledge level. In adaptive presentation, the content itself is adapted to the user-model, e.g. by leaving out details for advanced users or by generation additional information for beginners. Adaptive Hypermedia Systems are often implemented using CBR or other machine learning strategies. The iManual system that is presented in this thesis uses features similar to an Adaptive Hypermedia System. It uses a direct guidance strategy to lead the user to the help content that he needs in a certain situation and also implements a simple form of link annotation when giving additional information about why certain links are presented. Adaptive presentation is also planned to be implemented in future versions of the system. However, it extends typical adaptive hypermedia scenarios because it does not only use an implicit user-model but also contextual information from various sources to adapt to the context and because it does not want to guide the user through the complete content for educational reasons but to instantly offer the reference information that is needed in a certain situation. Furthermore, the hypertext includes embedded control elements that allow interactive communication with the device and therefore integrate learning and using.

3.2.2.4 Examples of CBR in Existing Systems

This chapter briefly describes three systems that successfully used CBR strategies.

HOMER, a Case Based Help Desk System developed for DaimlerChrysler, is used to support help desk employees that serve over 1500 engineers designing cars [Bergmann et al. (1999)]. As a typical help desk system, HOMER stores experiences of help desk operators in a knowledge repository and therefore makes corporate knowledge available for help desk employees on all levels. It can also be used to identify common problems and weak spots in engineering that have to be improved. A Rule-Based Reasoning sys-

tem was considered impossible to realize because of the complexity and size of the problem domain. Initial tests showed that 32% of all occurring problem could be used with the HOMER system and average time to solve one of these problems decreased from 141 minutes to 9 minutes. Evaluating the system also showed that important challenges for the success of such a system is to motivate help desk operators to input new cases into the case base and to maintain the knowledge base to ensure that the case base's cases are helpful and well-specified.

The *GM Variation-Reduction Adviser* (VRA) is a CBR-inspired system used in automobile plants to support employees that supervise the quality of vehicles and try to find reasons when a insufficient quality is detected [Morgan et al. (2003), Morgan et al. (2002)]. The VRA system is used as a knowledge repository as well as a problem solving and communication tool. A case-based approach was chosen because it was considered impossible to cover the whole domain with rule sets. The challenge of motivating users to input new cases was solved by combing the knowledge base and problem-solving function with a communication feature that allows workers in different shifts to communicate with each other. New CBR cases are mostly based on these messages. Attributes of a case include pre-selected symbols and free text, while similarity of cases is measured by searching for keywords in free text or comparing symbols with an ontology. The combination of a communication system and a CBR system is called a "success story" by the authors.

Both previous examples show that knowledge acquisition is much harder in CBR systems where users have to input new cases themselves than in systems where CBR cases can be generated automatically, e.g. from user interaction.

The *LISTEN* project, developed by the Fraunhofer Institute for Applied Information Technology FIT (now shortly called Fraunhofer FIT), is a system that provides audio information for humans moving in physical space [Zimmermann (2003)]. A prototype is planned to be used by visitors of a museum who are offered audio sequences regarding the user's spatial position, head orientation and personal interests. E.g., a user wearing a headphone might stop in front of a painting and the LISTEN environment would emit audio sequences giving additional information to that painting. The LISTEN project combines context-aware computing and CBR in a way that contextual information is represented in cases, and CBR can be used to find similar situations to the user's current context. The problem description of a case is mapped to the parameters of the four context dimensions (see chapter 2.1.2) and the solution to a recommendation that is given

in a certain context. Furthermore, a static context that holds values that do not change often is distinguished from a dynamic context that stores snapshots of contextual values that change constantly, e.g. the spatial location.

The LISTEN project is similar to the iManual scenario described here because it combines context-aware adaption with the use of a CBR strategy in a mobile situation and because it maps contextual information to cases in a case base. However, it concentrates on the context dimension of location while the iManual system concentrates on environment or activity and identity. Furthermore, LISTEN provides audio information to augment the physical space while the iManual system provides mostly textual information for assistive purposes. Third, iManual integrates functions to control the device into the contextualized information.

3.3 Integrating Learning and Using

As said in the previous chapters, a context-aware help system running on a PDA brings the opportunity that the PDA and the device can communicate via wireless links. This chapter describes that this opportunity leads to a completely new learning and using situation.

Consumer electronics, office devices, machines in the industrial domain, and devices in various other domains can be complex and therefore need help systems. Users have to learn to use those devices – some from the start and others only in reference situations and for unusual tasks.

The opportunity to embed control functionality into help content is based on a wireless communication channel. This communication between help systems and devices is bidirectional. First, the device can send its internal state and recent interaction with the user to the help system. Therefore, as described before, the help system is constantly updated on relevant contextual information and therefore can provide context-aware help that takes this information into account. Second, the help system can send commands to the device. Hence, any kind of function could be invoked both at the device directly and with the help system PDA. Any device that enables wireless communication could therefore be remotely controlled [compare Feldbusch et al. (2002), Nichols et al. (2002b), Myers et al. (2002), Zimmermann et al. (2003)]. However, what is promising is neither the remote control function nor the context-aware help system – but the integration of both.

By embedding control elements into the help system, the entire UI of the device can be projected onto the help system PDA. This is part of a scenario that is called *distributed* and *disappearing interfaces* in ubiquitous computing [Dey et al. (2001), Hu et al. (2003)]. Interfaces of modern devices, e.g. of the car's navigation system, can be projected onto any other computerized device with an input and output facility. A single device can therefore have multiple distributed and interconnected interfaces which are each suitable for a particular situation or an individual preference [see Burkey (2000), Landay (2002), Yates et al. (2003), Nichols et al. (2002a)]. Information systems can also be distributed on multiple devices, while some of these devices can be so integrated into the environment that the user doesn't recognize it as a computerized artifact.

Of course, to exploit these opportunities, the help system content has to be designed in a new way. Descriptions of how to fulfill a task on the device can be complemented with links that invoke the function directly. The help system no longer instructs "PRESS THE GRAY BUTTON TO…" but "PRESS HERE." A help system that both offers descriptions on how to operate the device and functions to remote control it is also a usability challenge. The user might expect a description to complete a task at the device when he finds a page that allows directly invoking the functions from the PDA or vice versa. A usable interface has to make clear which parts *describe* and which can be used to *control*. This can be done in many ways, for example by using metaphors or consistent icons. The icon used in the link above () is taken from the help of windows operating systems and used for the same purpose, i.e. indicating that the link in the help system invokes a command in the system.

As help systems and devices intertwine, learning to operate a device is coupled with using it. Integration of help system and device leads to integration of learning and using. Learning a system and being able to use it the same time improves learnability and learning motivation for several reasons:

First, integrating learning and using supports interactive discovery of the system. Whatever the learner reads about the device – he can always try that function out directly from the help system. Interactive discovery supports learning by doing, which is an effective way to gain new knowledge (see chapter 2.2.2.1). It was therefore identified by the minimalist approach as a key factor to design better manuals (see chapter 2.2.4.2). Therefore, minimalism demands to allow "training on real tasks." Interactive discovery is also included in many design guidelines for handheld computers to improve the handheld system's usability (see chapter 2.3.3.3).

Second, a help system wirelessly connected to the device supports coordination of help content and device. This "coordination of system and training" is also part of the minimalist approach. If a task is described by a tutorial or a step-by-step instruction, learnability is improved by coordinating or synchronizing the state of the device with the current step in the help system. The help system therefore assists the learner in keeping the device up-to-date so he can concentrate on learning the task itself instead of having to control that the device is in the state the instruction step requires.

Third, a user in a reference situation who is stuck in the middle of a task will appreciate the direct control functionality because he is not interested in learning as much as possible about the system but to get his task done. A context-aware help system that is connected to the device can be aware of user's recent interaction with the device. If he gets stuck then, the help system analyzes several contextual parameters including the recent interaction and offers context-aware assistance. These help topics could typically include parts like this: "IF YOU WANT TO DO THIS, PRESS THE GREEN BUTTON OR CLICK HERE." Many users would appreciate this kind of help because it is a fast way of solving problems and getting tasks done, which is what most users want when they interact with a device.

All in all, this kind of integration of help system and device could realize many features that most help systems for software already have. Many help topics for software systems today are closely integrated into the software itself and for example contain direct links that invoke actions in the software. As shown in chapter 2.2.4, strategies to bring a system and its help closer together have been shown to improve the usability of the systems. Realizing these strategies for physical devices could achieve similar success.

3.4 Organizing Content for Contextual Use

This chapter describes common challenges a context-aware mobile help system has to face as far as the representation and storage of the help content are concerned. The basic challenges for designing content have already been identified in chapter 2.3.3. It was described how content has to be written to be usable on handheld devices with limited screen size and which UIs are needed to cope with that screen size and the limited input facilities.

A main challenge for storing the context is the granularity problem. Any help content in context-aware systems might consist of sentences or words that are equal for all users,

and other sentences or words that are only shown in certain situations. But storing very small text fragments and combining them as needed is as unfeasible a solution as is storing various versions of longer text fragments that only differ slightly. Therefore, the help system must help the content provider in finding a good trade-off, e.g. by splitting the content into units of several sentences and choosing from several versions of the same content unit based on meta-information that is tagged to these versions. Russell (2001) describes the granularity problem related to web-based systems that have to decide which pieces of the information should be presented on the screen at a time and how to structure this information internally. In addition to solve the granularity problem, a context-aware mobile help system has to find a representation of the content that is suitable for individual presentation of the help on various devices with different output facilities. This could for example be done with XML or a database schema.

Another important challenge to ensure that the content can be used in a context-aware system is to provide an authoring system that offers the content provider an easy way to write and edit the content so that it can directly be used in the help system - without requiring too much technical knowledge from the authors. Typically, a context-aware help system would provide a kind of content-management system (CMS) similar to those CMSs that allow authors to edit content on web-pages without knowing about underlying web-design technologies. However, compared to a CMS for the web, a CMS for a context-aware help system has to offer more: The authors have to be able to write several versions of the content for each situation that requires a different representation of the help information. Furthermore, authors have to provide a kind of information tagged to each chunk of content that describes for which contextual situation this part should be presented. This could for example be achieved by defining a metadata structure that is tagged to each chunk, together with a comfortable way for authors to edit this meta-information.

Chapter 4.4.5 describes how the design guidelines for mobile content have been realized in the iManual prototype, how content is represented and stored in the system and how the granularity problem was addressed. The chapter then describes which kind of CMS was used to provide the appropriate authoring environment for content providers.

3.5 Summary – Chances and Challenges

This chapter summarizes the chances and challenges of context-aware mobile help systems by reviewing the concepts that have been described in this section and adding related aspects.

Chances

A basic functionality of a context-aware mobile help system is to analyze contextual information and adapt the presentation and the help contents to that context. This provides assistance or instructions that are tailored to the user's individual need and therefore improves the usability and increases user satisfaction. This function of a context-aware help system was described in chapter 3.1.

User satisfaction can further be improved by enhancing automatic adaptivity with explicit adaptability. If not enough contextual information is available to make a decision about which content or which presentation mode is most appropriate in the current situation, the missing information could be provided by explicitly stated user preferences. In addition to specifying their preferences, users should be given the possibility to actively tailor the layout, presentation, or functionality to their individual needs.

The portability and mobility of the help system and the opportunity to communicate with the device wirelessly brings further possibilities. Most important, it enables integration of help system and device and therefore – as described in chapter 3.3 – leads to a new learning and using situation where both activities are integrated.

If context-aware mobile help systems become available for many devices that surround us in our daily life, this could support the vision of ubiquitous and pervasive computing [Weiser (1991), Gerstner (1998)]. When "technologies disappear into the background" [Weiser (1991)] and millions of smart computerized devices will be "interwoven in the global fabric of computing and communications" [Gerstner (1998)], some computerized devices will be so invisible that users do not recognize that they are actually using a technological system and therefore might need no help system. On the other side, as this network of interconnected intelligent devices gets more powerful, even in gentle-slope systems complex tasks might still be complex to fulfill, and therefore, intelligent, context-aware, and personalized help might still be needed. Furthermore, like users might not notice that they are actually using computerized devices, they might not notice either that they are receiving help by integrated intelligent help systems.

The handheld's connectivity to the Internet also enables communication and cooperation among users as well as between users, manufacturers and content provides. Users can give various kinds of feedback to help pages, for example annotations or ratings, which are then available to all other users. This can help users to learn from other users' experiences, find solutions for problems that are not covered by the help system, or get additional information if help pages are not understood. Furthermore, it can help the content provider to improve his content and the manufacturer to identify badly-designed product interfaces or common errors. This Internet-based cooperation can create user communities that could evolve into a large product-related knowledge base and that would benefit users, content providers and manufacturers.

Challenges

Developing a context-aware mobile help system for a physical device also brings challenges that have to be faced.

As said in chapter 2.1.5 about the challenges of context-aware computing, context-aware help system have to cope with the frame problem. That means that the real world is infinitely complex and constantly changing, so a context-aware help system has to model the world by deciding which aspects are relevant to provide intelligent help, and to keep the model up-to-date in that constantly changing environment.

Along with the frame problem, a context-aware help system has to develop a way to cope with the multitude of possible contextual constellations. Intelligent help systems that adapt the given help to various contextual parameters have to implement a strategy to show an intelligent reaction to every possible situation. In this section, I showed that machine learning strategies are promising approaches because they are based on a knowledge base and are able to extend their knowledge by learning from observations. I also explained why CBR, among the various machine learning strategies, best fits to the requirement of a context-aware mobile help system. CBR does not require to cover all possible situations with a rule set but is able to start with few seed cases and extend its knowledge base dynamically by learning from experiences – even from failure (see chapter 3.2.2.2).

Organizing the help content for a mobile and contextualized use is a further challenge. As said in chapter 3.4, a context-aware mobile help system has to find a way to store and represent the content so it can be adapted to the current context. Is also has to offer

an authoring system for content providers to add and edit their content and to provide the meta-information that is needed for contextual adaption.

Along with that, the mobile help content has to be designed to meet the specific requirements of a mobile, portable handheld computer with small screen size and limited input facilities. Guidelines for writing and designing text and for designing the UI of the mobile system have been identified in chapter 2.3.3.

The help system's design also has to support the integration of learning and using. While this is a main benefit for a context-aware mobile help system, as described in chapter 3.3, it can also be a design challenge. Control elements that invoke functions on the device being integrated into help content that only describes the device can also be confusing for users. A well-designed help system has to make clear which parts describe functions of the device, which parts invoke these functions, and where the border between both parts lies.

The following section 4 describes the realization of a context-aware help system – the iManual-System. It will describe in which ways the chances that were described in this sections are put into action and in which ways the challenges are met.

4 iManual - the System Prototype

This section describes the iManual system – a prototypical realization of a context-aware mobile help system. It describes which concepts of such a system are realized in the iManual prototype, how these concepts were implemented and why the particular implementation techniques were chosen.

First, I will explain which problem domain was chosen for the prototype. The subsequent chapter describes the main features of the iManual system: context-triggered help, contextual links, embedded control, and a design optimized for mobile scenarios. The following chapters then describe how these features have been implemented: This starts with a functional overview of the iManual architecture and its components. Then, I will provide details of the design and implementation steps of the iManual system. This also includes the discussion of basic design decisions that influenced the architecture and the implementation of the system. First, I will describe which platforms and which programming languages have been used and why they were chosen. Second, I will describe Bluetooth™ and web services, the main communication techniques that have been used, and justify why they were taken. Then, it is shown how the Fraunhofer FIT CBR package was integrated to implement CBR functionality and how this is used to provide help that adapts to the current context. After that, I will explain how the help content of the system is stored, represented and handled and how the Adaptive Learning Environment (ALE), developed by Fraunhofer FIT, was used to support these tasks. Finally, I will discuss further work that is needed to improve the iManual system and to realize more concepts of a context-aware mobile help system.

4.1 A Help System for the Automotive Domain – BMW 7

A context-aware mobile help system could be used for many kinds of devices. For the iManual-System one of the most complex, most computerized and most expensive devices that end users typical own was chosen – a car. As described in chapter 2.4, a context-aware help system developed for a car has several advantages. Cars have a quite high complexity, offer a great number of functions, and the help system has a low relative price compared to the car. Furthermore, cars are already equipped with thousands of sensors to deliver contextual information, and many of them offer integrated IT sys-

tems that often can even be used with PDAs. Thus, due to the complexity, the many functions and the integrated IT systems, there is a need for an intelligent help system and due to the many sensors, the low relative price, and a possibly existing PDA interface there is the chance to implement a useful context-aware help system.

For presentations and evaluations of the system, parts of the manual for a BMW 7 were redesigned and integrated into the system. Figure 12 shows that the description of the navigation system of the BMW 7 and related chapters were chosen for the prototype.

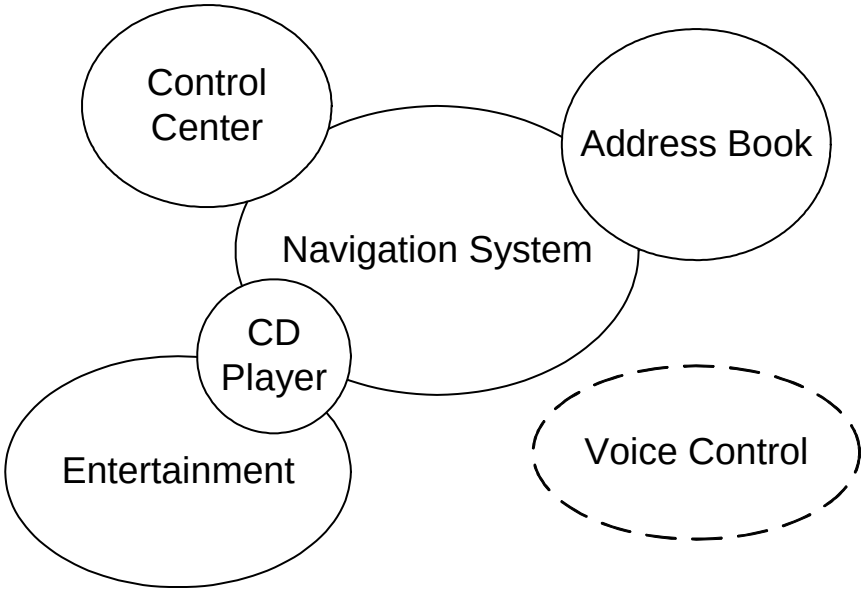

Figure 12: Help system content of the prototype.

The internal Address Book of the BMW 7 is related to the navigation system because addresses can be used as targets for the navigation system and current targets can be saved into the address book. The entertainment system is related to the navigation system in a way that both use the CD player – the former to play music, the latter to load map CDs. The description of the control center (i.e. the main display and the controller – a button that is used as the main input facility) was added because its display is used to present the route and its controller to input targets, select options etc. The description of how to use voice control was included because most parts of the navigation system can also be operated by voice.

4.2 System Features

This chapter summarizes the main features of the iManual system. By enabling these features, the concept of a context-aware mobile help system is put into practice. The

chapter only described features that were realized in the prototype, leaving features that might be implemented in future to chapter 4.5 – Further Work.

4.2.1 Context-Triggered Help

Figure 13: Context-triggered help.

##Hinweis: Platzhalter. Richtiges Bild wird später eingefügt. Kann zur Zeit nicht erstellt werden, weil das PDA-Kabel defekt ist. Das gilt auch für die weiteren Bilder dieses chapters.##

The PDA that the iManual help system runs on is constantly updated on relevant contextual data, including contextual parameters that are measured by the car. Based on this constant communication of car and PDA, the PDA shows important alerts or warnings about the car's internal state or important information about current actions. Hence, this presentation of assistive information is triggered by context changes. For example, if the car detects an important defect at one of its sensors, the help system immediately shows a warning that includes ways to solve the problem, related background information, and links to local car dealers if necessary (see Figure 13).

Another example in which a changed context triggers a help message on the PDA is a traffic jam that the car's navigation system might detect. If that jam is on the way that is currently shown in the navigation system, an alert would come up showing detailed traffic information followed by instructions on how to let the navigation system com-

pute alternative routes or to let it find the target again if the driver bypasses the jam himself (see Figure 14).

Figure 14: A traffic jam triggers contextual help.

4.2.2 Contextual Links

The iManual system also offers individual help according to the current context. Because the help system is always up-to-date about relevant contextual information, it can offer a function that analyzes the current context and shows links to the most helpful topics in that situation. These links are called *contextual links* (see Figure 15). The links are self-describing and the number of links that are shown in each situation can be adjusted by the user, so he can decide himself if that information is relevant and helpful in that situation or if he wants to ignore it. In the prototype, the presentation of contextual links is triggered by the user clicking on an “iHelp” button. For future versions, empiric evaluation is needed to find out if this kind of context-aware help is better triggered by the user or by the system, e.g. on system startup or when opening the start page.

Figure 15: Contextual links.

Based on the reaction of the user, the system learns to improve the given help in similar situations in the future. Chapter 4.4.4 will describe which steps are needed to determine the most relevant help, how the system learns from previous experiences and how this functionality is implemented.

4.2.3 Embedded Control

The communication channel between the car and the PDA allows transmitting several kinds of messages in both directions. Based on that communication, an integration of using and learning is realized. Messages sent from the PDA to the car include commands that invoke functions of the car, and other kinds of user input. For security reasons, the commands only invoke functions that are not related to driving the car because these functions are better operated with the more familiar interface of the car. But for many secondary systems of the car, this PDA-based remote control is promising. Functions of the navigation system might easier be invoked by the PDA when currently reading about that function in the help system. Inserting addresses into the car's address book might be easier on the PDA, because text input is more familiar and faster on the PDA than by using the BMW's controller. Furthermore, the addresses can even be taken from the PDA's address book to minimize necessary user input. Of course, due to security reasons the PDA's pages and the embedded control functions can only be used if the car is standing, or by passengers who are not driving. If the car is driving and there

are no passengers except the driver, voice control and audio output have to be used – yet another interface that is most appropriate in a certain situation. Offering the physical interface of the car, its projection onto the iManual PDA and an additional audio interface is an example for distributed interfaces as described in chapter 3.3.

In the iManual system, a target for the navigation system or a new address for the address book can be inserted in many ways, including with the PDA, which is probably the fastest and easiest way (see Figure 16).

Figure 16: The PDA as an alternative input device.

Furthermore, many pages of the help system have embedded links that invoke the function of the car that is currently described (see Figure 17). Thus, remote control features are embedded into the help content, which makes coordination of the car and the help system much easier and therefore supports integration of learning and using, as described in chapter 3.3.

Figure 17: Remote control functions integrated into help content.

4.2.4 Design for Mobile Use

The iManual system presents assistive and instructional information designed for a handheld device with a limited display size. Therefore, the guidelines to design UIs and to write text for small displays that were developed in the mobile HCI field and presented in chapter 2.3 were translated into action.

The iManual system provides multiple and direct ways to access information because this has been identified as a key factor to support information access on handheld devices. In addition to providing a table of contents, a keyword index and a search facility, the contextual links described above are also a way of information access. The contextual links and the table of contents are always accessible from the toolbar, which further supports direct access (see Figure 18).

Figure 18: The help system's home page

Furthermore, the iManual system tries to provide a consistent and easy-to-remember navigation through the system. The same icons are used on all pages to navigate to the previous page or the menu page. Furthermore, step-by-step instructions consisting of two or more pages have a consistent navigation bar on top of the page (see Figure 19).

Figure 19: Consistent navigation.

Another strategy to make navigation as easy as possible is to use a similar look than on the device. For instance, the overview page that leads to the help about the main menus of the car's information system looks like the main menu itself.

Figure 20: Comparison of the main menu in the help system and in the car.

An important factor for handheld information systems is to keep the information short and compact. iManual uses a layering strategy to focus on the main information on a single page and make additional information available via self-describing hyperlinks.

Figure 21: Layering with hyperlinks.

Finally, the iManual system's text is "designed for scannability." That means that long uniform text blocks are avoided and the text is structured with well-places headlines, consistent and easy-to-remember icons, bullet lists, or pictures.

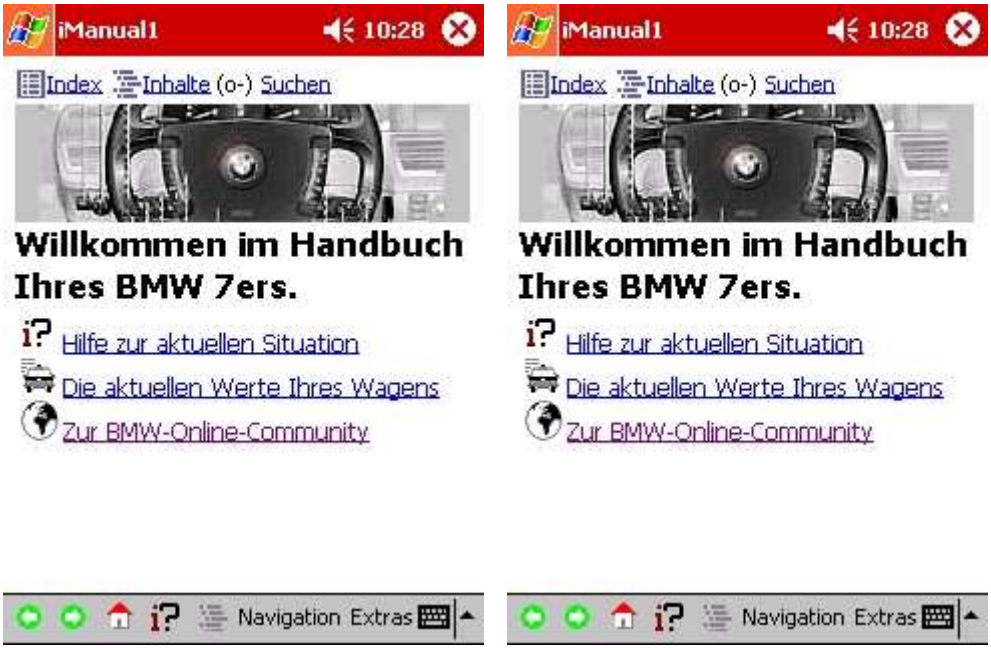

Figure 22: Design for scannability.

4.3 Architecture

This chapter will give an overview of the main software and hardware components that were used for the iManual prototype. It tries to describe the components of the iManual system on a functional level. Hence, it will focus on *what* the components do and *what* they communicate rather than *how* they do it and *how* they communicate. The following chapter 4.4 then provides technical details of the software artifacts, of how they were implemented, and how they communicate.

4.3.1 The Big Picture

Error! Reference source not found. shows that the iManual system basically consists of three logical parts:

- The iManual-compatible car
- The iManual PDA
- The iManual Server

The PDA and the car communicate over Bluetooth™ while the PDA and the iManual Server use web services.

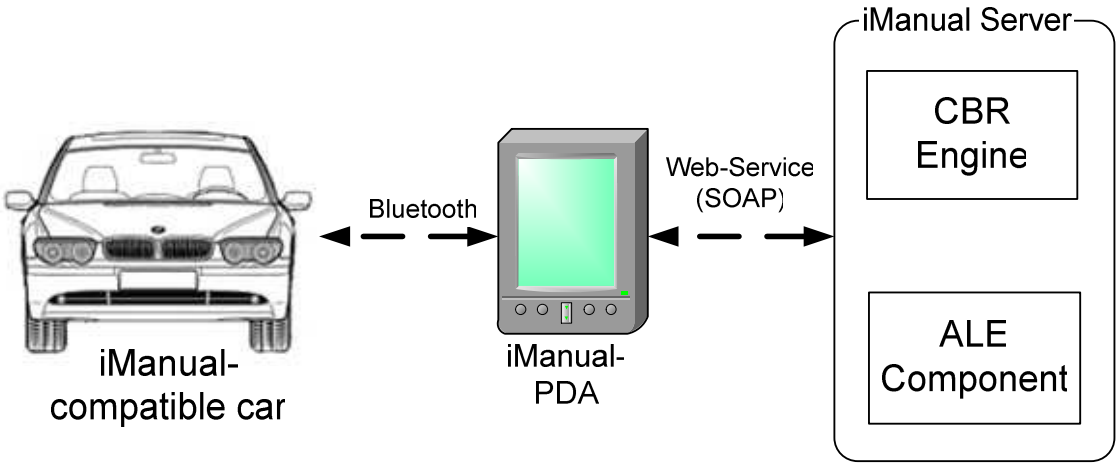

Figure 23: Main components of the iManual system.

4.3.2 Functional Overview

To describe the components of the iManual system and their functions, a more detailed overview is provided in Figure **24**.

4.3.2.1 iManual PDA

The iManual PDA is the main input and output device in the iManual architecture and has several responsibilities. First, it presents the help content and allows interactive navigation through the help system. Second, the PDA collects the contextual information. In the prototype, this information includes user interaction, internal state changes and sensor data sent from the car, as well as the user's identity and his preferences that are stored on the PDA itself. Third, the PDA requests content from the iManual server. This either happens when the user clicks on a (static or contextual) link or uses the search facility. The iHelp function determines which content is most helpful for the user in the current context and displays appropriate contextual links. For this, the PDA sends CBR-requests including a representation of the contextual parameters to the iManual server and receives a description of the links to most relevant topics. Another responsibility of the PDA related to CBR is to trigger the learning process on the iManual server. Because learning is based on the success of previously provided help, the PDA determines if current help was successful and sends new cases to the server that represent the context, the provided help and the success.

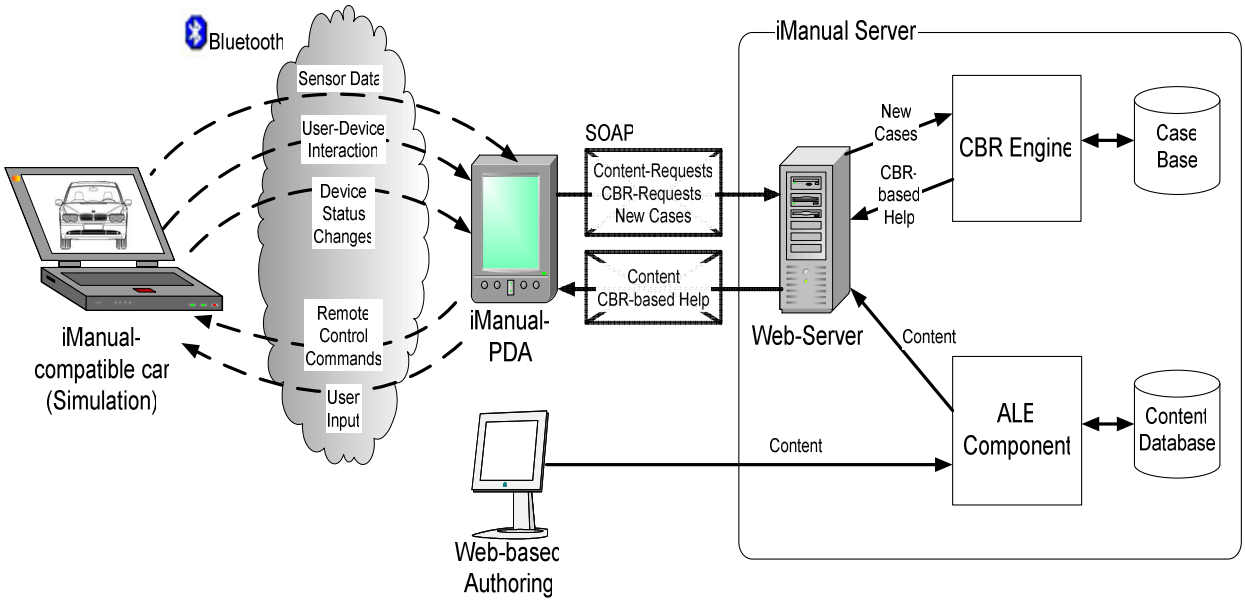

Figure 24: Detailed architecture of the iManual system.

How the content requests of the PDA are handled is described in chapter 4.4.5, while the CBR functions including the learning process are explained in chapter 4.4.4. Implementation details of the PDA components are given in chapter 4.4.1.

4.3.2.2 iManual-compatible Car

The car in the iManual architecture is responsible for determining contextual information and sending it to the PDA over the wireless Bluetooth™ link. The car sends three kinds of messages to the PDA. First, it determines contextual data from its various sensors and sends relevant information to the PDA, e.g. the remaining gas in the tank or the presence of any kind of defects. Second, the car monitors the user interaction and sends relevant actions to the PDA. Third, the car informs the PDA about changes of its internal state, e.g. the number of entries in the address book or the currently selected viewing option in the navigation system. Hence, the car keeps the PDA updated on all relevant contextual information.

In the other direction, the car receives messages that enable the mentioned integration of learning and using and the embedded control on the PDA. These messages include commands that invoke functions of the car's system and other user input.

As shown in the overview, the iManual system did not use a real car but implemented a simulation of the relevant properties running on a laptop. The simulation includes a graphical representation of BMW's cockpit with an imitation of the navigation system and some related functions. All the menus of the navigation system and the address

book are rebuilt, so the graphical output is close to the look of the real system. The usage of the controller to select items from the menu and to insert text was simulated by buttons for each menu item and each letter of the alphabet (see Figure 25 to Figure 27). The user clicks those buttons with his mouse instead of turning the controller until the menu item is marked and then pressing the controller. The application on the laptop also simulates the Bluetooth™ communication between the iManual PDA and the car, which is described in chapter 4.4.3. In the rest of this section, when I talk about "the car," I mean the simulation of the car.

Figure 25: Screenshot of the simulation of the navigation system of the BMW 7

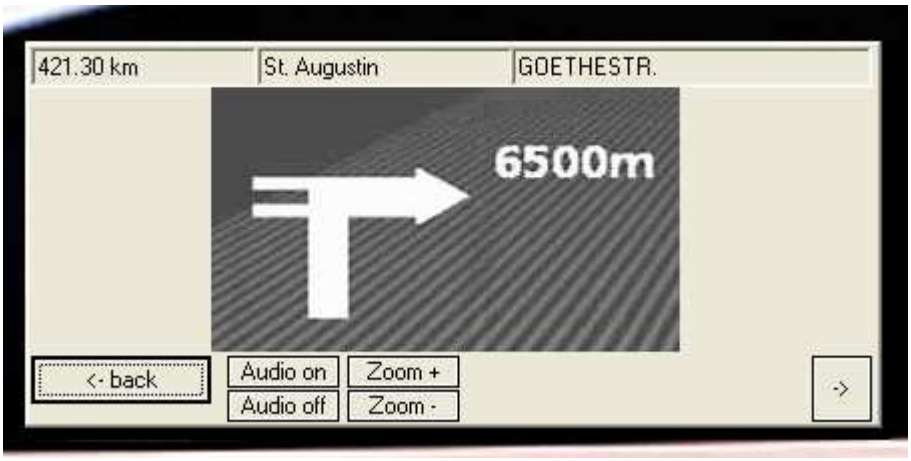

Figure 26: Simulation of a typical screen of the navigation system

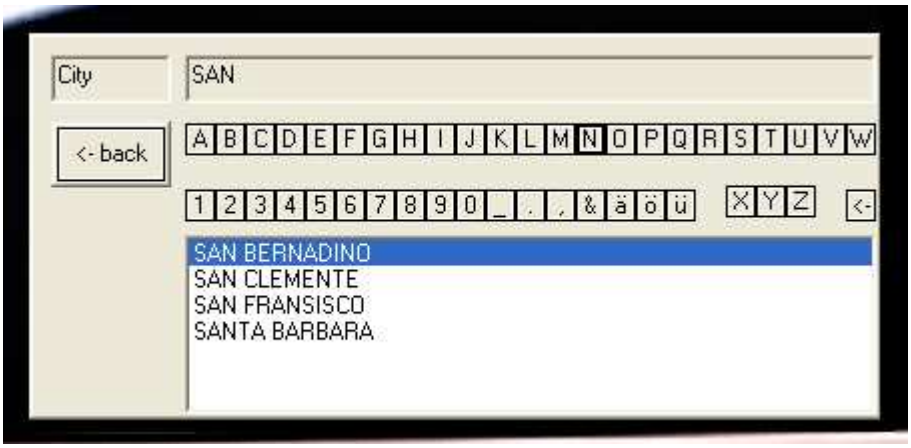

Figure 27: Simulation of text input into the navigation system

4.3.2.3 iManual Server

The iManual Server basically consists of three components:

- The *web server* handles web service requests from the PDA and forwards them to the other two components. How this was implemented is described in chapter 4.4.2.
- The *CBR engine* implements the CBR functionality of the iManual system. The engine receives CBR requests from the PDA forwarded by the web server. These requests include a representation of the current contextual parameters. Bases on these requests, the CBR engine identifies which help topics are most relevant and sends a representation of contextual links back to the PDA. It also receives new cases from the PDA which represent the success of the provided help in a certain context. These cases are used to improve future help in similar situations. They are only added to the case base if they improve the knowledge of the CBR system. To obtain the content related to a contextual link, the ALE system is used. How the contextual links-feature is realized with CBR and how the Fraunhofer FIT CBR package was used to implement this is described in chapter 4.4.4.
- The *Adaptive Learning Environment (ALE)* is used as the Content Management System (CMS) of the iManual system. It stores the help content in a database and offers an interface that the web server can use to obtain the desired contents. Furthermore, is allows web based authoring so that any authorized writer can insert and edit the help system's content from any Internet-connected computer

through an easy-to-use graphical interface. How ALE was integrated into the iManual server and how content is stored, represented, and managed is described in chapter 4.4.5.

4.4 iManual in Detail – Design and Implementation

This chapter gives details of the development and the implementation of the iManual system. It also reports about important decisions that influenced the architecture and the implementation of the system. Alternative implementations that were taken into account are compared and reasons for choosing the existing solutions are given.

First, it is described which programming languages and which libraries were used for the main components of the system and why they were chosen. Next, this chapter elucidates how the components communicate with each other and why the particular communication techniques have been selected. This includes wireless Bluetooth™ communication and web service communication transmitted over a wireless Internet link. Then, the question of how to store, represent, and deliver content is answered. In the iManual system, content is stored on a server and transmitted over wireless Internet connections. Last, the chapter describes how CBR techniques were implemented to realize the context-aware adaption to the current situation. This was done by integrating and extending the Fraunhofer FIT CBR package.

Readers with limited technical background who are not interested in learning how the system features described above were realized might also skip this chapter and continue with the discussion section. Understanding the discussion and conclusion section will not depend on the technical details given here.

4.4.1 Platforms and Programming Languages

As said above, a Fujitsu Siemens Pocket LOOX 600 was used as the iManual PDA. The PDA runs on Microsoft Pocket PC 2002, which is based on Microsoft Windows CE 3.0. The decision of which programming language, which IDE and which libraries should be used for the PDA depended on three main requirements: the support for Bluetooth™ (or virtual serial port) communication, the availability of libraries for web service communication and the support of rendering HTML on the device. Based on these requirements, Embedded Visual C++ (eVC) was chosen, because using hardware components such as Bluetooth™ devices and virtual serial ports is relatively easy with C++, free

web service libraries are available and an HTML component is integrated [Microsoft MSDN (2004)]. HtmlView, the HTML component integrated into eVC offers similar functionality and the same familiar look than Pocket Internet Explorer, the web browser that is integrated into Pocket PC. Alternative languages such as Java™ were also taken into consideration but had some crucial disadvantages. J2ME, the Java™ version designed for small devices [Sun Microsystems (2004)] depends on third-party Java™ Virtual Machines (JVMs), such as SuperWaba [SuperWaba (2004)] or IBM's J9, which is part of an extensive IDE called WebSphere Device Developer (WSDD) [IBM (2003)]. J2ME has no integrated support for HTML, and third-party libraries to render HTML are quite costly. Furthermore, Bluetooth™ or virtual-serial-port communication is harder to implement because Java™ is designed to be platform independent.

For the simulation of the car simulation, also C++ was chosen because Bluetooth™ communication with the PDA was easier to implement and to maintain using the same programming language on both sides. Of course, also any other programming language could be used for the simulation. If future versions wanted to offer a better graphical simulation, possibly even in 3D, other languages that better support those graphical requirements should be considered.

For the iManual-Server, Java™ was chosen for two reasons: First, several components were used in the iManual server, such as a web server, a web service toolkit, and the CBR engine. Integrating the chosen components into a Java™ application was much easier because they were all designed to be used with Java™. Second, the Java™ language is platform independent and therefore the iManual server could easier be transferred to other platforms or operating systems.

4.4.2 Web Service Communication

The communication between the PDA and the iManual server was implemented with web services. Web services are modular service applications that can be invoked from anywhere in the internet by exchanging messages that are independent from any platform or programming language [W3C Consortium (2003b)]. Therefore, web service are seen as a „new paradigm to the process of creating highly distributed applications" [Chopra et al. (2001)]. Web services are self-describing and can be published, located and invoked from any network-connected platform. The messaging protocol used for web services is SOAP (Simple Object Access Protocol), an XML-based open

Internet standard [W3C Consortium (2003a)]. Web service messages can be transmitted over several standard Internet protocols such as HTTP or SMTP.

Web services were chosen for the iManual system because they allow communication of distributed components over the Internet regardless of the platform, programming language or physical location. In the iManual system, the C++ component on a Pocket PC platform communicated seamlessly with the Java™ component running on a Windows XP platform. Therefore, replacing the Pocket PC PDA with another platform, e.g. Palm OS, would not require any changes on the iManual server.

On the PDA, a web service client was supported by using an open source library called PocketSoap [PocketSOAP (2003)]. Because web services are a rather new technology, PocketSOAP is one of only few tools available for the Pocket PC 2002 platform. On the iManual server, web service functionality was realized with Apache Axis, an open source web services toolkit for Java™ [The Apache Software Foundation (2003)]. To realize server functionality, i.e. to offer a service that clients can call over the Internet, Axis was integrated into an Apache Tomcat web server [The Apache Software Foundation (2004)]. Axis is a web application that can easily be integrated into web servers and Java™ applications. It automatically generates web services for existing Java™ methods and maps requests to those services to the appropriate Java™ classes. Axis was preferred over other web service toolkits such as Systinet WASP [Systinet Corporation (2004)] because a better interoperability with the PocketSoap library on the PDA side was experienced.

4.4.3 Bluetooth™ Communication

Bluetooth™ is used as the communication channel between PDA and iManual-compatible car (i.e. the car simulation on the laptop). Bluetooth™ is a connectivity standard that allows wireless communication over short distances. It is a standardized technology developed by an industry consortium [Bluetooth SIG (2004)], uses a freely available radio frequency (2.4 GHz), allows communication over distances of up to ten meters, and offers a maximum bandwidth of 723.2 kb/s [Stajanao (2002), Weiss (2002)]. It is particularly designed for small devices with limited battery power and is available in most modern PDAs today.

Although the car is only simulated in the prototype, Bluetooth™ was considered the most likely communication channel between a real car and a handheld computer be-

cause it is an industry standard, designed for small devices and available on most handhelds.

In the prototype, both PDA and laptop have integrated Bluetooth™ hard- and software. Bluetooth™ communication can possibly be realized in many ways but often depends on the availability of Application Programming Interfaces (APIs) and toolkits developed by the Bluetooth™ hardware manufacturers. Due to the lack of such a toolkit, a virtual serial port was established over Bluetooth™. A virtual serial port is a constant Bluetooth™ link that can be used as if it was a serial connection [compare Barr (2000)]. Therefore, the communication partners do not have to know about the specifics of Bluetooth™ communication, but send messages to the other side as if connected by a physical serial cable.

4.4.4 Context-Awareness With Case-Based Reasoning

Chapter 3.2 already justified why CBR is a reasonable approach to realize adaptive and intelligent help features in a context-aware mobile help system. This chapter describes how CBR helps the iManual system to provide this kind of help and how the Fraunhofer FIT CBR package was used to implement this functionality.

In the iManual prototype, CBR is used to provide contextual links in a reference situation. As mentioned in chapter 2.2.3.3, the reference situation is characterized by a user who is stuck in the middle of using the device and needs help fulfilling his task. In this situation the iManual system provides help by displaying contextual links that lead to topics that are helpful for the user to fulfill his task. They are determined by analyzing contextual parameters and using CBR techniques. These techniques relate the current situation to similar situations of the past and reuse successful solutions of those previous situations.

This is not the only situation where the use of CBR techniques is helpful in this iManual scenario. For example, another usage that is planned to be implemented is connecting a link with the appropriate content according to the user profile. In the first prototype, a link always leads to the same target. In later version, a link could possibly lead to various versions of the same content, and a CBR strategy could evaluate which version is appropriate for the current user based on experiences of the past.

4.4.4.1 Case Model

A *case* in CBR consists of a problem situation and a solution to that problem. The *problem situation* in the iManual scenario is characterized by the user in a certain context requesting help. An iManual *solution* is the presentation of a contextual link and a rating of that solution. The *rating* is a numerical representation of the success of the provided help. It is determined by analyzing the user interaction that follows the representation of the contextual link. A solution is successful if its rating is high and unsuccessful if its rating is low. Possible success factors to rate a solution are:

- Did the user click on the contextual link?
- How long did he watch the related topic and possible follow-up pages?
- Did he add an explicit rating for that page and what was the rating?
- Did he leave the page by clicking “cancel,” “back,” “next page,” or did he close the iManual application?

Combining these factors leads to a general rating of the current solution. In the system prototype, the first three factors above are considered. Based on these factors and a weighting of them, a numerical rating value is determined by a rating algorithm. An important task for further development is a usability study to find out which factors make up a successful solution and how they have to be considered. Then, the rating algorithm has to be evaluated, adjusted and possibly a learning strategy has to be implemented that can adjust the rating algorithm dynamically.

Thus, an iManual case consists of a problem situation (i.e. attributes that represent the current context), a solution (i.e. a representation of the contextual link), and a rating of that solution. The contextual parameters that are considered by the iManual prototype are descried now.

4.4.4.2 Considered Context Parameters

In chapter 2.1.2.1, I already identified the context dimensions relevant to a context-aware mobile help system. I also identified in chapter 3.1 which aspects of these dimensions could possibly be used in such a system. Among these context parameters, the iManual system considers the identity of the user, the user’s preferences, the last user interaction with the iManual-PDA, the last user action with the car, and various statuses of the car. A possible interaction with the iManual could be the user using the search

function, an interaction with the device could be the user opening the “target” menu of the navigation system and possible statuses of the device could be number of targets in the target list of the navigation system, the number of entries in the address book, the remaining fuel or the fact that the car is moving or not.

To determine contextual links based on these context parameters, the iManual system concentrates on the retrieval and retain step of the CBR cycle (see chapter 3.2.2.1).

4.4.4.3 Retrieval

The retrieval step is the search for similar cases compared to the current problem situation. This is done in by a two-step retrieval. A pre-selection query filters out cases that are probably not relevant and then a sequential retrieval is executed on the remaining cases using the attributes’ local similarity measures.

The system prototype uses different kinds of local similarity measures depending on the data type and semantic characteristics of the attributes. Examples are given in chapter 4.4.4.5. Combining these local similarities to a global similarity is done using the weighted sum. The weights of the attributes were specified after short test with few cases in the case base. In future development, these have to be evaluated and adjusted, or a learning strategy has to be implemented so that these weights can change dynamically. CBR strategies could be used to implement this kind of learning, but other strategies such as Neuronal Networks should also be considered.

While most CBR systems then transform the solution of the most similar case to the given problem, the iManual CBR system considers a *set of cases* that are most similar to the problem. Among this set of cases, all cases with identical solutions (i.e. with the same contextual link that lead to a certain topic) are grouped and the system computes the average rating for each group. The retrieval then is finished by returning the solutions with the highest ratings and by presenting the contextual links on the PDA’s display.

For example, in a certain contextual situation, the most helpful contextual links are searched in a case base of 1000 cases. The situation is also called query case and a contextual link is a solution of a case. First, the pre-selection step might filter out 900 cases because they are not similar to the query case based on the simple pre-selection query. From the remaining 100 cases, 50 might be very similar to the query case. Each case has a solutions and a rating, but two cases can of course have the same solution. So the

50 cases might have 10 different solutions. Then, for each of the 10 solutions, the average rating is computed and the solutions with the highest ratings are returned.

4.4.4.4 Retain

As said above, the retain step is responsible for gaining new knowledge. A CBR system can learn in four ways: it can learn new cases, improve its similarity measuring, reorganize its case base or improve the way of transforming existing solutions. In the iManual system's retain step, the system gains knowledge by adding new cases to the case base. Other forms of learning mentioned above are not implemented in the current version. However, future versions might integrate learning strategies to adjust the case model, the weights of the local similarity measures, or the way a rating for a solution is determined. The system stores new cases consisting of the problem situation, the solution and the solution's rating. To prevent the case base from 'exploding' (i.e. to become too large to allow an efficient retrieval), only cases are added that are not too similar to any existing case. Checking if there is such an existing case is a similar process than the retrieval step. If a new case is similar to an existing one, the rating of the old case is replaces by the arithmetic mean of the old and new rating and the new case is dismissed then.

This is only a first strategy for limiting the growth of the case base. Further development is needed to evaluate other strategies for this aim, especially because the iManual prototype is going to be used by multiple users that each can generate new cases. For example, a case could have a kind of expiry date that a process could check frequently to *forget* cases that are too old. Another strategy would be to replace several similar cases in the case base with a single *generalized case* [Bergmann (1996)].

Not only strategies to limit the growth of the case base, but also to initialize the case base have to be developed [Leake (1996)]. CBR systems only learn from previous experiences and in the iManual system the learning process is only triggered by the reaction of the user to the provided help. Therefore, if the case base was empty, the retrieval step would have no cases to search for and could not provide any contextual links. Consequently, the learning process would not be triggered and the case base would remain empty. While there are many strategies to cope with this problem, the iManual prototype uses two strategies. First, the case base is initialized manually with a basic set of cases when the server is installed. Second, the learning process is also triggered in a situation that is not dependent on the presentation of contextual links: In the system,

users can add annotations and ratings to each page. As said before, the explicit rating is also considered when determining the rating of a CBR solution. In order to trigger the learning process not only after contextual links were shown, a new case is also generated after adding or modifying an explicit rating. The case then consists of the contextual parameters (problem situation), the currently rated page (solution) and the explicit rating. This helps to generate a basic set of cases that future retrieval steps can use.

4.4.4.5 Integration of the Fraunhofer FIT CBR Package

To realize CBR functionality in the iManual system, a CBR package developed by Fraunhofer FIT was integrated into the system. The package is a collection of Java™ classes that can be integrated into other Java™ systems. It can be extended easily because it built highly modular. The package is a part of a bigger package that is currently being developed at Fraunhofer FIT. It can easily be configured via XML files. Cases can either be stored in active memory, XML files or relational databases. For iManual, a mysql database was chosen [MySQL AB (2004)]. Communication with the PDA that triggers the retrieval process and generates new cases is done via web services that are forwarded by the iManual web server (see chapter 4.4.2).

The CBR functionality was implemented by integrating and extending the Fraunhofer Fit CBR package. The package already offers basic CBR functionality such as loading and saving cases, organizing cases in a linear case base, local and global similarity measures for various types and sequential retrieval of cases. Additionally, I extended the package in four ways:

First, I added local similarity measures for iManual-specific types. In the prototype, a case has a `DeviceActionId` attribute that represents the user's interaction with the car. Each `DeviceActionId` has a set of related keywords and therefore, two user actions are compared by comparing their keyword sets. Therefore, a new local similarity measure named `KeywordSetMeasure` was added to the CBR package:

A user action of case c_1 that has n related keywords has a similarity of m/n to the user action of case c_2, if m keywords of c_1 are equal to one of c_2's keywords.

Due to the modular structure of the CBR package, the class could easily be added by deriving from an abstract super class and implementing the similarity function. It is then integrated by specifying this kind of similarity measure in queries of the retrieval or

retain step. Appendix B shows the class diagram of local similarity measures in the CBR package.

Second, I added a two-step retrieval to the package. The CBR package offers a framework that allows seamlessly plugging in new retrieval methods. Furthermore, it already includes a sequential retrieval class that walks through the case base sequentially to find similar cases to a given case. The `TwoStepRetrieval` class that was added to the package uses a pre-selection query to filter out cases that are probably not similar to the given case. The remaining cases are then compared using the sequential retrieval. A detailed description of the two-step retrieval process if given in the UML sequence diagram given in Appendix C .

Third, a special way of modeling cases needed for the iManual scenario was added to the system. As said above, an iManual case has a solution and a rating. Therefore, classes to deal with these solutions and ratings were added (see Appendix D for details). These classes seamlessly work together with the retrieval classes like `SequentialRetrieval` and `TwoStepRetrieval` mentioned above.

Fourth, a basic retain step was added to the CBR system. It is essential for a CBR system to prevent the case base from growing too large. While there are various strategies available for this problem, in iManual a `RetainManager` was implemented that adds new cases only if they are not too similar to existing cases and *forgets* cases otherwise. Before forgetting such a new case, an iManual-specific `iManualRetainManager` replaces the rating of the existing case by the arithmetic mean of the old and new rating.

4.4.5 Content Management

This chapter describes how the help content of the iManual system is organized. First, I will describe why the iManual architecture uses a central server to store and deliver the content. Second, I will present how the help content is stored in the prototype system and how this representation might be improved in a future version. Last, I will describe how the ALE system was integrated into the iManual system.

4.4.5.1 Content Storage

When designing a context-aware mobile help system, the content can basically be stored at three different places: on the PDA, on a central server, or on the device. After comparing advantages and disadvantages, it was chosen to place the content on a server that

can be accessed over the Internet. A main advantage is having only one version of the content that can easier by maintained by the content provider. Furthermore, content can better be complemented by annotations, ratings or comments from the user community, which are then available to all other users. Third, this solution is also feasible for handheld devices with minimal storage capacity because hardly any data has to be saved on the handheld. On the other side, transmitting the help content over the Internet could be a bottleneck if the handheld bandwidth is limited. However, considering that acceptable bandwidth is already available today via WLAN and excepting that greater bandwidth at lower cost will be available in future (e.g. via GPRS or UMTS), this disadvantage is acceptable. Furthermore, the car might also have an Internet connection in the future that the PDA could use. To overcome the bottleneck of a slow or unavailable Internet connection, the help system could use several strategies: An easy way that is already implemented in the system is to offer a setting to switch to an offline mode if no acceptable connection is available. For that, of course, a copy of the complete content has to be transferred to the handheld when the help system is installed or when a fast connection is available. Furthermore, the help system could offer buffering strategies that would only transmit content if the user requests a new page or a page that was changed since the last visit.

4.4.5.2 Content Representation

The help content that is presented to the user on the PDA contains formatted text and images, i.e. the content types that are also usually found on web pages. Therefore, the content displayed on the PDA is encoded in HTML, the standard way of representing web content. This has the advantage that standard components on the PDA can be used to handle the HTML, render it and present it on the display. As said above, choosing an HTML representation allowed using HtmlView, a standard HTML component for Pocket PC. Of course, choosing to use an HTML component on the PDA does not mean that the content has to be stored as HTML on the server, too. Nevertheless, for the prototype, exactly this was done. Hence, content providers create their content in HTML on the server (with a graphical editor) and the same format is transmitted to and displayed on the PDA. The granularity problem has been solved by storing the content in units that typically fit on one PDA page. This ensures that the effort to combine the content of a page from various units is not too high and that the editorial work to write several versions of the content is not too much.

However, the HTML representation and units of the size of a PDA page have some disadvantages that have to be addressed in future versions of the system. HTML does not separate content and layout, so HTML encoded content will look very similar on each device. Modifying content always requires thinking about the layout, too. Cascading Style Sheets (CSS) are one way to define consistent layout rules for large sets of HTML pages, but still, the page size of the content would not be adaptable to different screen sizes and, even more important, the content is limited to devices that are able to display HTML. A possible solution to these disadvantages is to define a XML representation for mobile help content. XML is a flexible, platform independent way to represent many types of information. The XML content could then be transformed into various output formats, e.g. HTML, WAP or PDF, that would each be optimized for a particular device or a certain user preference. XML allows strict separation of layout and content and would not require splitting the content into a predefined unit size. Furthermore, different versions of the content related to different contextual situations could be included in the XML representation as well a meta-information to define which content is intended for which situation.

4.4.5.3 Integration of the Fraunhofer FIT Adaptive Learning Environment

The help content of the iManual system is stored and managed by the ALE system developed by Fraunhofer FIT. ALE is a learning management system that allows creating educational content that consists of reusable learning objects. It presents individualized courseware for learners depending on user preferences, learning style etc. [Specht et al. (2002a), Specht et al. (2002b)]. ALE also supports cooperation among learners by bringing together learners with similar interest and supporting cooperative work.

In the iManual prototype, ALE is only used as a kind of CMS. ALE enables web-based authoring and modifying content or layout from anywhere in the Internet by multiple authors. The features of ALE intended for learners, such as adapting content presentation to the user's preferences, were not used because the special situation of a context aware mobile help system required implementing other strategies to adapt the provided help to contextual parameters. Therefore, a new interface was developed to deliver the contents of the ALE system to the mobile user via web services.

The ALE system is currently developed further at Fraunhofer FIT. For future versions of the iManual system, the existing and upcoming features of the ALE system will be

evaluated to decide if more tasks of a context-aware mobile help system can be realized with ALE.

4.5 Further Work

The iManual system is a prototypical realization of the concept of context-ware mobile help systems. This chapter showed the main features of the system and explained how they were realized. However, as already said in some chapters, the system has to develop further to realize other concepts or improve existing features in order to improve the given help. Therefore, this chapter summarizes possible improvements and extensions to the current system. Needed further work that was already mentioned in the previous chapters of this sections will also be summarized.

In general, the iManual prototype was implemented as a basis for evaluating the feasibility of the concept of a context-aware mobile help system. Therefore, the iManual prototype has to be tested empirically to identify conceptual strengths and usability weaknesses.

New Domains

The prototypical help system was implemented for a BMW 7 and especially for its navigation system. This is considered as a promising domain because of several reasons mentioned above. However, the iManual system was designed to be as independent as possible from any domain and the concept of context-aware mobile help systems is promising for a large range of devices. Therefore, future versions have to be designed for other devices such as washing machines, copy machines, TVs or VCRs, mobile phones, drill machines, or industrial machines in many branches.

Conceptual Enhancements

As said above, the Internet connection of the PDA (or possibly the device) enables various features of a context-aware mobile help system. In the current version of the system, users can add annotations and ratings to each page of the help system. However, at the moment, this kind of user feedback is not available to other users. The annotations have to be stored in the CMS. This would enable cooperation among users and therefore be a basis to build a user community. For example, if a user does not understand a certain topic in the help system, he can check if other users attached annotations to the topic that might clarify the issue or link to other helpful information.

This kind of community-building is also valuable for the device manufacturer and help content provider. The former can use community feedback to find usability weaknesses or identify common errors. Both could be used to improve future versions of the product. Furthermore, building a user community would increase customer satisfaction and loyalty. The latter could use that feedback to improve the help content itself.

The iManual prototype shows an example to integrate machine learning strategies into a context-aware help system. CBR – the chosen strategy – as well as other learning strategies could be used for other functions, too. As said above, not only contextual links, but the description of a topic itself should be adapted to the user's context, especially his knowledge and learning preferences. Determining the most appropriate version of the help content when clicking on a link therefore could be supported by CBR in future versions.

The ALE system that is currently only used as a CMS for the iManual prototype also supports adaption of the presentation of learning content to the user's knowledge, preferences and learning style. Therefore, ALE can be used to provide personalized content based on a user-model and an appropriate representation of the contents. This feature, which is currently used for web-based e-learning in other systems, has to be evaluated and possibly be adapted to the requirements of a context-aware help system.

The basic ability to tag metadata to the learning objects and to analyze it to present appropriate content could be used in the iManual scenario. Advanced authoring facilities, such as metadata-tagging and writing different versions of the help content for specific situations should therefore be supported in future versions of iManual (and ALE).

Functional Enhancements

In this chapter, several design and implementation decisions were discussed and in some chapters, it was already mentioned that alternative realizations would improve existing or enable new features.

As already said, the current iManual system demands an Internet connection or a local copy of the entire content of the help system. To provide other alternatives for situation, in which the Internet connection is not available, to slow, or too expensive, buffering strategies could be implemented in future versions.

The existing CBR system could also be extended in several ways. As said in chapter 3.2.2.1, a CBR system can learn in many ways. For the presented realization of CBR in

a context-aware help system, two ways are relevant. First, in the retrieve step, the weighting of the attributes of a case could be managed by a machine learning strategy. Determining which attribute is more and which is less important when comparing two cases could be supported by a CBR strategy, by a rule-based approach or by a Neuronal Network. Second, to decide when two cases are similar and when not should also be supported by a kind of dynamic strategy.

A very important step for further work on the iManual system it to evaluate the rating algorithm. This algorithm determines if the provided contextual link was successful help or not. Usability tests have to find out which factors make the user satisfied with the provided help and how to measure this satisfaction. Based on these tests, the rating algorithm has to be adjusted, possibly even dynamically at runtime.

Furthermore, as said above, additional strategies to initialize the case base and to prevent the case base from growing too large have to be evaluated. Additionally, user tests also have to find out if the retrieval and presentation of contextual link should be triggered by the user or the system.

Another task for future work on the iManual system that was mentioned above is to improve the representation of the content and its presentation on the device. A first task is to find a more flexible format for representing the content. Defining an XML format and transforming it into multiple output formats would be a more flexible solution than using HTML and optimizing the layout for a fixed screen size.

The way the content is presented on the handheld device could also be improved, but that would of course depend on the problem domain and the content itself. For example, instructions to change parts of a digital camera or to repair a laser printer should better be presented in interactive 3D animations. Interesting web-based examples are shown in ParallelGraphics (2004).

All in all, many concept of a context-aware mobile help system have been realized in the iManual prototype. Therefore, the prototype is a valuable basis for evaluating the concept, while it could still be extended in several ways.

5 Discussion

Questions of the Thesis

This thesis analyzed how human competence can be supported by context-aware mobile help systems running on handheld computers such as PDAs. Based on the main characteristics of a handheld – mobility, connectivity, interactivity and personality – it was examined which new opportunities arise to assist humans in learning and operating surrounding physical devices. It was further asked how context-aware mobile help system can adapt the help content and its presentation to different contextual parameters such as the user's knowledge, his previous interaction with the device, or the internal state of the device.

Expectations

It was expected that users like such help systems better than traditional paper-based manuals and were more successful in operating devices when supported by a context-aware mobile help system. It was further prognosed that the help provided by such an adaptive help system is what users expect in the current situation and what they need to fulfill their task. Finally, it was expected that help systems based on handheld computers enable many of the strategies that successfully improved the help of software systems when it evolved from paper-based manuals to integrated and embedded help.

Opportunities of the Concept

Based on these expectations, the concept of a context-aware mobile help system was introduced. These systems present the help system of a physical device on a portable, interactive and personal handheld computer that is wirelessly connected to the device and the Internet. It was shown that the personal and connected handheld device can be used to adapt the provided help to the internal state of the device, the user's interaction with the device, and the help system, as well as his level of expertise, knowledge, experiences and preferences. The quality of the assistance can further be improved by enhancing automatic adaptivity with individual adaptability. It was further shown that mobility, connectivity, and interactivity enable embedding control elements into the help content. Therefore, the device can be operated from the familiar interface of one's personal handheld computer over a wireless link, e.g. a Bluetooth™ connection. This leads to a new learning situation where learning and using are integrated. Third, it was

exposed that Internet connectivity enables linking the help content to external information sources and supports communication and cooperation among help system users, content providers, and manufacturers.

Challenges of the Concept

The analysis of the opportunities of the concept also identified main challenges for any context-aware mobile help system. First, the system has to develop a way to cope with the multitude of possible constellations of contextual parameters in an infinitely complex and constantly changing world. Second, it has to find a strategy to store and represent the help content and to offer an appropriate authoring system for content providers. Third, content has to be designed for a handheld device with a small screen size and limited input facilities following mobile HCI guidelines.

Realizing Opportunities

The concept of a context-aware mobile help system was realized in the iManual system, a prototypical implementation of a context-aware mobile help system for the navigation system of a BMW 7.

The main opportunities that transferring a help system onto a handheld computer brings are realized in the iManual system. Based on analyzing contextual information from the device and the handheld, the iManual system provides context-triggered help. Also based on this contextual information, the system suggests contextual links that provides assistive information that is most relevant to the current situation. The opportunity to integrate learning and using is realized by embedding control elements and input facilities into the help content. The benefits of this integration can be enhanced by exploiting the specific opportunities of the PDA being a personal information and communication device. For example, inserting a target for the navigation system with the PDA can be facilitated by taking these addresses from the PDAs address book.

It was also recognized that the prototype can be extended in several ways to exploit further opportunities of a context-aware mobile help system. The Internet connection has to be exploited in two ways. First, cooperation and collaboration has to be supported by providing feedback channels to rate pages and add annotations. This creates user communities and lets them share their experiences and knowledge. Second, the help content has to be linked to external information sources to offer related information.

Furthermore, adaptivity should be better supported by adaptability. Users should be able to add information that can be used to improve the adaptive reaction of the system. Along with that, users should have extended possibilities to personalize the presentation and tailor the help system to their needs.

Meeting Challenges

The main challenges of a context-aware help system have been addressed. To cope with the multitude of different contextual situations, a CBR strategy was used in the iManual system because of several advantages. CBR systems store situations that really occur and do not require to cover the while problem domain with rule sets. Furthermore, they have a dynamic knowledge base that learns from success *and* failure, do not require experts to set up rule sets, offer an increasing quality of solutions, and support explainability and understandability. The CBR system is used to offer contextual links based on the current context. The system learns from the user's reaction to the provided help to improve future reactions in similar situation. It was found that only few seed cases are necessary to initialize the knowledge base, and that the provided help constantly improves while the system is in use. However, it was also found that the rating algorithm that determines if the given help was successful or not is crucial in such a CBR system. Future work is needed to empirically test what makes up successful assistance and how to identify a satisfied user. Based on these tests, the rating algorithm has to be improved, and it has to be evaluated if machine learning strategies can be used to dynamically adjust the algorithm. Similar empirical tests have to show when two situations are similar enough to reuse previous contextual links and which attributes of the case are most relevant for this comparison. At the moment, contextual links are only reused if two situations are very similar, but empirical research will probably show that reusing can also be reasonable in less similar situations.

Further development to improve learning is not only needed within the CBR system. CBR or other machine learning strategies can also be used in other situations of a context-aware mobile help system, for example when determining which version of the same content is most appropriate according to the user's learning level and previous knowledge.

The CBR functionality was implemented with a CBR package developed at Fraunhofer FIT. It was found that its XML-based configuration makes seamless integration into

larger systems very easy and that its highly modular structure greatly supports extending the package and adding new functionality.

The challenge to organize, store, and represent content for context-aware mobile use has been met by using ALE as the content server and the system's CMS. It was found that using a central server to store the content makes modifying and maintaining the content very easy for the content provider and enables cooperation among users. However, as long as the Internet connections of some PDAs are too slow, too expensive, or not available, alternatives or enhancements to that architecture have to be considered. It was found that offering an offline mode is practical because it guarantees the availability of the help system in such situations.

ALE provides an easy-to-use graphical user interface to insert or modify help content. However, other systems developed at Fraunhofer FIT show that the ALE system can be used for more than just content management. Because ALE also supports adaption of the learning objects to the user's knowledge and preferences, future work has to evaluate if iManual's adaptive features, which are currently provided by the CBR system, can also be provided or enhanced by ALE.

The chosen solution to store the content as HTML pages on the server needs to be viewed critically. This representation was chosen because the prototypical development concentrated on adapting suggested help to contextual parameters and integrating learning and using, and not on finding flexible representations for various environments. Although ALE provides a graphical HTML editor that allows authors to edit content without knowing about HTML, this representation is still quite inflexible. First, providing help for different devices with varying screen sizes, that might not even be able to display HTML, would be awkward with HTML and second, showing different personalized and contextualized versions of the same content would be impeded by such a representation. Therefore, future work will have to find more flexible representations applicable for a broader range of mobile devices.

The challenge to design the UI and write the help content optimized for handheld devices was met by providing direct and multiple access to the information, offering a consistent navigation through the hypertext, keeping the content short and compact and designing it for scannability. These guidelines in general have been empirically evaluated by other authors, but the realization of these guidelines in the iManual system has to be analyzed empirically.

Lessons Learned From Development

The development of the iManual system has given some significant insights into implementing mobile, wireless, PDA-based applications. First, it was found that eVC is better suited to developing applications for the Pocket PC platform than Java because a significantly better performance was experienced and better toolkits and libraries were available for the iManual requirements. Second, it was experienced that web service communication is a very flexible and easy-to-maintain way of developing distributed systems. With web services, different programming languages on different platforms seamlessly communicate with each other. Furthermore, more and more powerful tools are available that transform abstract service definitions into ready-to-use programs and therefore shorten implementation time. Third, compared to implementing web services that communicate over standard Internet connections, implementing Bluetooth™ communication requires significant development effort. The Bluetooth™ technology is quite new and most applications that use Bluetooth™ are developed by OEMs. The virtual serial port solution that was chosen for the iManual prototype provides a flexible communication channel between two programmable devices, but for other scenarios that need to use advanced Bluetooth™ features, alternative implementations that are dependent on the availability of APIs and toolkits provided by hardware manufacturers or driver suppliers have to be found.

Evaluating Expectations

Initial tests with the iManual system showed that the expectations that were expressed prior to the development of the system and the writing of this thesis seem to be fulfilled. However, detailed empirical evaluations have to be made to completely verify or falsify these expectations. The iManual system is a valuable basis for such empirical user tests.

Those kinds of evaluations have to show if users like context-aware mobile help systems better than traditional paper-based manuals and if they are more or less successful in fulfilling tasks with the device. The iManual system's implementation is relatively independent from any domain, so the system could also easily be developed and tested for other devices than a car.

The question whether the provided help is what users expect and what they need to fulfill their current task needs further evaluation, too. Initial tests showed that the system gave helpful advice in various situations and that the given help improved based on a growing knowledge base of the CBR system.

The iManual system transfers many strategies that are used in help systems for modern software applications to help systems for physical devices. Several software-related approaches that were described in the state of the art can also be found in the iManual prototype:

The context-triggered help feature of the system is a kind of *active help*, which has been shown to be useful in some software systems. Evaluations – e.g. with help systems used in real cars – have to verify if this is also true for physical devices.

The kind of *context-aware help* that is found in software system can also be found in the iManual system. However, help in iManual goes beyond the context-aware help that is included in some software applications. The iManual system considers a wider range of contextual parameters, especially personal attributes of the user, and offers a stronger integration of learning and using, because it does not only enable to invoke functions from the help system but can be used as an alternative input device and even an alternative interface for the device's functions. Furthermore, the personal and familiar handheld computer can include the help systems of numerous physical devices at any location.

Interactive Help is realized in several ways. Using hypertext in general lets user navigate interactively through the system. The layering strategy further supports interactive and individual help. Finally, the mentioned integration of learning and using takes interactivity to another level by offering interactive navigation through the help system combined with embedded control elements to operate the device.

The concept of *embedded help* is not always directly transferable to physical devices. The iManual system takes a contrary approach. Instead of embedding help into the device, it embeds the device's interface into the help system.

Usability tests have to show if these strategies have similar success on help systems for physical devices than on help systems for software systems.

6 Conclusions and Outlook

This thesis discussed the concept of context-aware mobile help systems and presented a prototypical implementation – the iManual system. These systems complement human competence by providing help that adapts to the user's individual needs and his current contextual situation. The help is presented on a mobile, interactive, and personal hand-held computer that is wirelessly connected to the device and the Internet. By embedding remote control elements into the help content, these systems create a new learning situation that integrates learning and using. Furthermore, they support Internet-based communication and cooperation among users.

Machine learning strategies can be used to implement functions of context-aware help systems and to improve provided help based on a knowledge base. This thesis showed how CBR can be used to achieve context-aware adaption and to improve future help based on current user interaction.

The ubiquitous availability of situation-aware help that is wirelessly transferred to a single personal and familiar device is one step towards a scenario that offers appropriate, personalized and unobtrusive help whenever we need it. In this scenario, PDAs will be personal devices that accompany us all day, carry various personal information, enable many kinds of wireless communication, and offer assistance for the all the devices that surround us or are embedded into our environment.

The projection of the device's interface into the help system is one step towards distributed interfaces. Devices of the future will no longer have a single interface that is physically attached to it. According to the current situation and his individual preferences, the user can choose between multiple interfaces on various devices.

The iManual system will be developed further at Fraunhofer FIT. Usability tests will show if users appreciate help that adapts to contextual parameters and if they are more successful in operating devices when supported by the iManual system, also for other devices in different domains. Further work will evaluate if contextual adaptivity should be enhanced by individual adaptability, how cooperation among users can be supported by the system and how advanced machine learning strategies can be used to support the provided help.

7 Acknowledgements

This thesis is dedicated to my mother who always supported my universal studies and my father who would be proud to see this final result.

I would like to thank Prof. Dr. Volker Wulf at the University of Siegen and Dipl.-Inf. Markus Klann at the Fraunhofer Institute for Applied Information Technology for supporting this thesis, answering numerous questions and giving helpful advice.

Special thanks go to Markus Klann, Stefan Apelt, Andreas Zimmermann, Marcus Specht and many more at the Fraunhofer Institute for Applied Information Technology who supported me in designing or developing the iManual system. I also would like to thank the developers of several open source software systems that I used to develop the system: Simon Fell, who developed PocketSOAP, the developers of Axis and Tomcat at the Apache Software Foundation, the developers of the Eclipse IDE and the developer community at The Code Project and the Pocket PC Developer Network.

In addition to those mentioned above, I want to thank Gregor Müller at the University of Tulsa and Oliver Klee at the University of Bonn for reviewing my thesis.

Appendix A

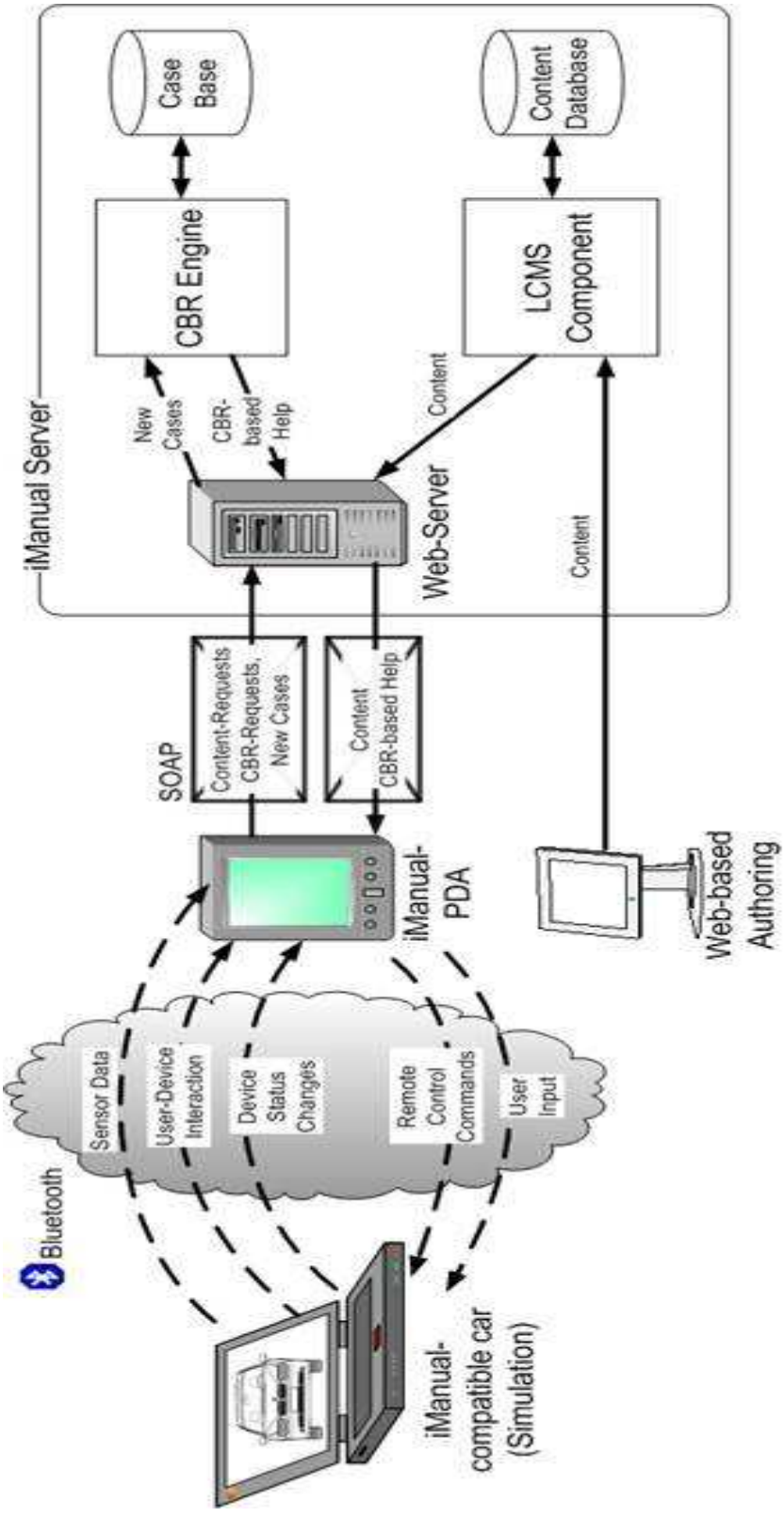

Figure 28: The iManual Architecture (detailed).##Hinweis: Diese größere Version ist u.u. nicht nötig, wenn das kleinere bild auch lesbar ist##

Appendix B

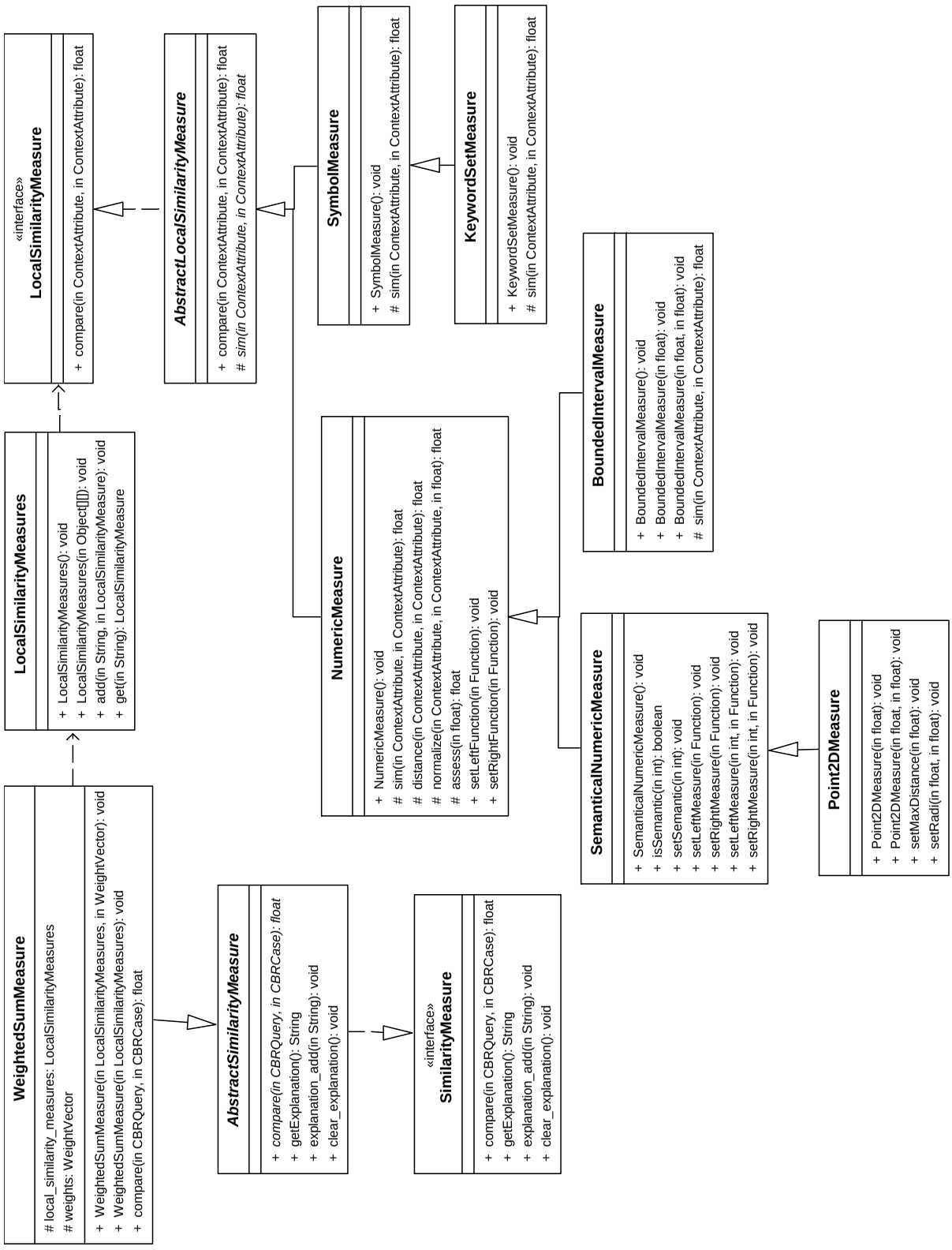

Figure 29: UML Class diagram of local similarity measures in CBR system.

Appendix C

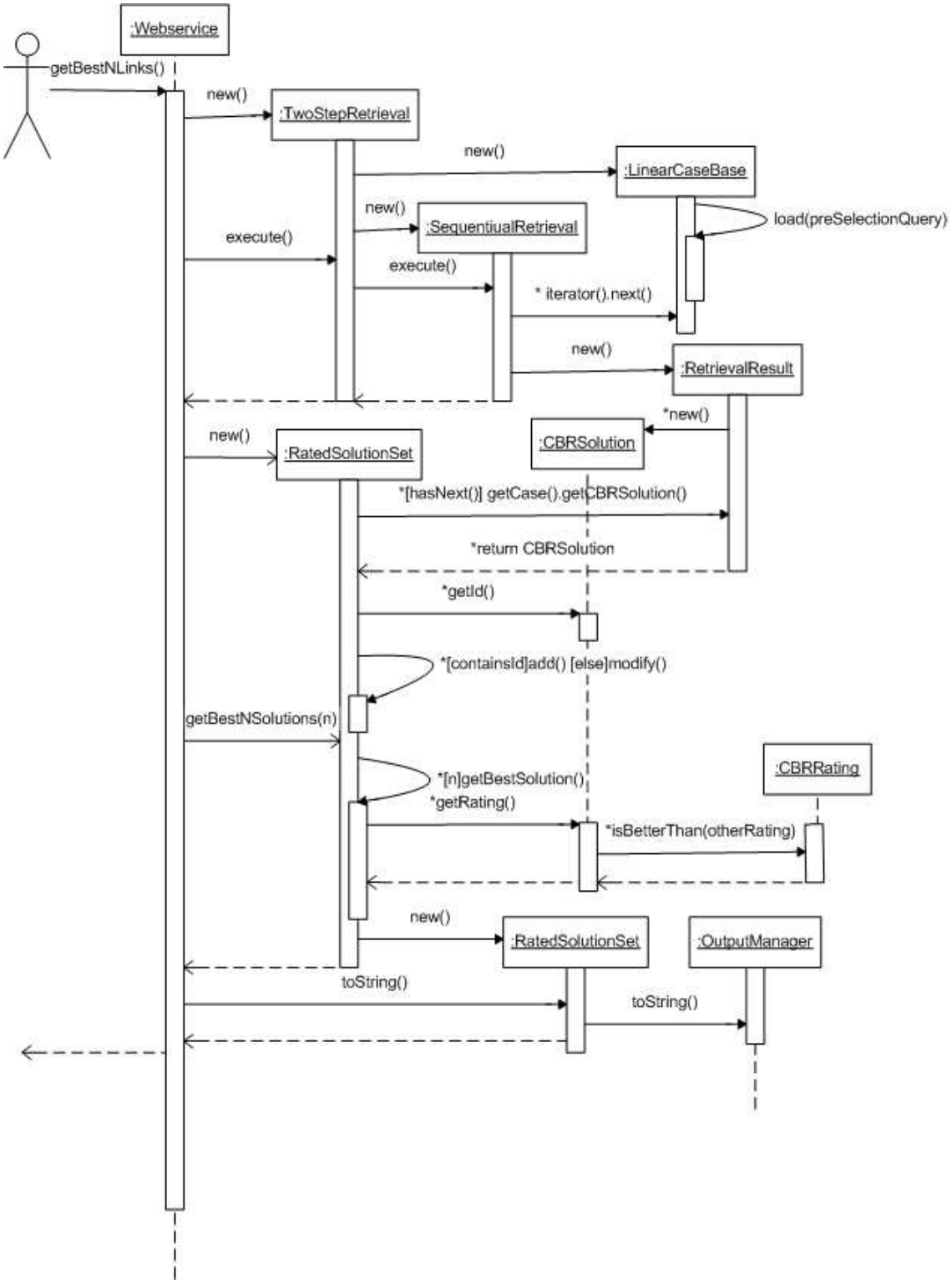

Figure 30: UML sequence diagram of the two-step retrieval process.

Appendix D

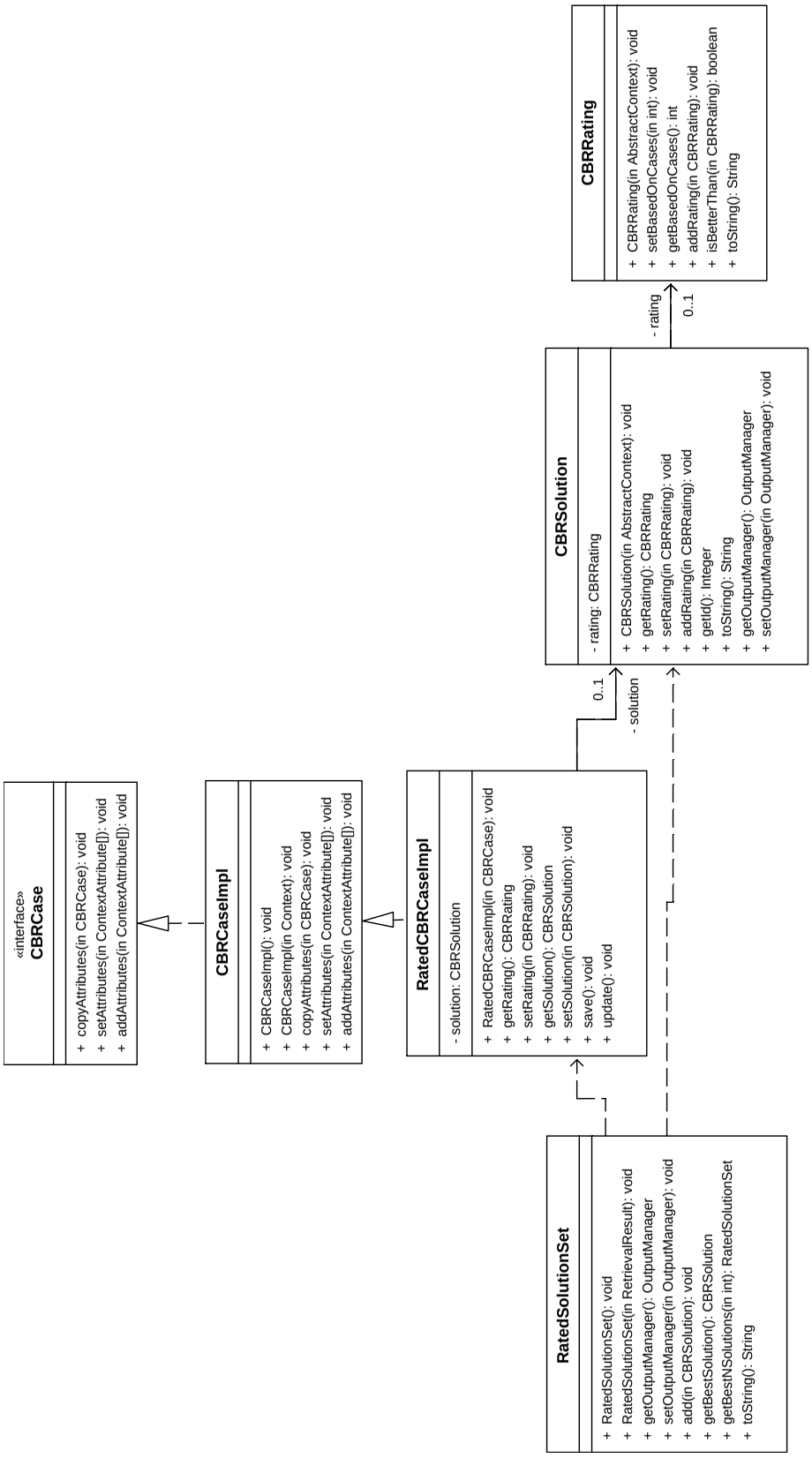

Figure 31: UML Class diagram of a `RatedCBRCase` and its solution and rating.

8 References

Aamodt et al. (1994)

Aamodt, A.; Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. In: AI Communications 7(1), 1994, p.39-59.

Abowd et al. (2001)

Abowd, G. D.; Mynatt, E. D.: Charting Past, Present, and Future Research in Ubiquitous Computing. In: *Carroll, J.M.(Ed.)*: Human Computer Interaction in the new Millennium, p.513-535, ACM Press, New York, USA 2001

America Online (2003)

America Online. T9 Text Input. http://www.t9.com/, 2003, Accessed on 30.11.2003

Amor (2002)

Amor, D.: Das Handy gegen Zahnschmerzen und andere Geschäftsmodelle für die Dienstleister von morgen. Galileo Press, Bonn 2002

Andrews et al. (1980)

Andrews, D. H.; Goodson, L. A.: A comparative analysis of models of instructional design. In: Journal of Instructional Development 3(4), 1980, p.2-16.

Anson (1998)

Anson, P. A. H.: Exploring minimalism today: a view from the practitioner's window. In: *Carroll, J.M.(Ed.)*: Minimalism Beyond the Nurnberg Funnel, The MIT Press, Cambridge, MA, USA 1998

Apple Computer (1996)

Apple Computer. Balloon Help. http://developer.apple.com/documentation/mac/HIGuidelines/HIGuidelines-240.html, 1996, Accessed on 30.11.2003

Aristotle Rhetoric (1990)

Aristotle Rhetoric: Trans. W. Rhys Roberts. In: *Bizzell, P.; Herzberg, B.(Ed.)*: The Rhetorical Tradition: Readings from Classical Times to the Present, Bedford Books, Boston 1990

ATX North America (2003a)

ATX North America. ATX. http://www.atxna.com/, 2003a, Accessed on 02.07.2003

ATX North America (2003b)

ATX North America. ATX Unveils First Voice-Activated Vehicle Owner's Manual. http://www.atxna.com/news/pr_2003-04-22_vehicleownersman.asp, 2003b, Accessed on 02.07.2003

Audi AG (2003)

Audi AG. Audi telematics. http://www.audi.com/de/de/neuwagen/elektronik/info/audi_telematics/audi_telematics.jsp, 2003, Accessed on 27.06.2003

Baecker et al. (1990)

Baecker, R.; Small, I.: Animation at the interface. In: *Laurel, B.(Ed.)*: The Art of Human-Computer Interface Design, p.251-267, Addison-Wesley, Reading, USA 1990

Baecker (2003)

Baecker, R. M. The Design of Interactive Computational Media. http://www.dgp.utoronto.ca/people/RMB/318S2003/lecture/lect12.pdf, 2003, Accessed on 30.11.2003

Barr (2000)

Barr, M.: Virtual Serial Ports. In: Embedded Systems Programming 13(2), 2000, p.33-37.

Bauer et al. (1988)

Bauer, J.; Schwab, T.: Anforderungen an Hilfesysteme. In: *Balzert, H.; Hoppe, H.U.; Oppermann, R.; Peschke, H.; Rohr, G.; Streiz, N.(Ed.)*: Einführung in die Software-Ergonomie, p.197-214, de Gruyter Verlag, Berlin 1988

Bergmann (1996)

Bergmann, R.: Effizientes Problemlösen durch flexible Wiederverwendung von Fällen auf verschiedenen Abstraktionsebenen. Ph.D., University of Kaiserslautern 1996

Bergmann (2002)

Bergmann, R. Wissensentdeckung und Maschinelles Lernen. http://www.dwm.uni-hilde-sheim.de/dwm/cms/lehre/Wintersemester20022003/VorlesungWissensentdeckung/weml1.pdf, 2002, Accessed on 05.01.2004

Bergmann et al. (1999)

Bergmann, R.; Breen, S.; Göker, M.; Manago, M.; Wess, S.: Developing industrial case based reasoning applications: the INRECA methodology. Springer, Berlin 1999

Bibliographisches Institut & F.A. Brockhaus AG (2001)

Bibliographisches Institut & F.A. Brockhaus AG: Brockhaus multimedial 2002. Mannheim 2001

Billsus et al. (2002)

Billsus, D.; Brunk, C. A.; Evans, C.; Gladish, B.; Pazzani, M.: Adaptive Interfaces for Ubiquitous Web Access. In: Communications of the ACM 45(5), 2002, p.34-38.

Bluetooth SIG (2004)

Bluetooth SIG. Bluetooth - The Official Website. http://www.bluetooth.com/, 2004, Accessed on

BMW AG (2002)

BMW AG: Owner's Manual for Vehicle BMW. 2002

BMW AG (2003a)

BMW AG. BMW Online. http://www.bmw-telematik.de/bmw_online.html, 2003a, Accessed on 27.06.2003

BMW AG (2003b)

BMW AG. Connected Service. http://www.bmw.com/e65/id15/5_ti_cbs.jsp, 2003b, Accessed on 27.06.2003

Boedicker (1990)

Boedicker, D.: Handbuch-Knigge: Software-Handbücher schreiben und beurteilen. Wissenschaftsverlag, Mannheim; Wien; Zürich 1990

Bourguin et al. (2001)

Bourguin, G.; Derycke, A.; Tarby, J. C.: Beyond the Interface: Co-evolution inside Interactive Systems - A Proposal Founded on Activity Theory. In: Proceedings of IHM-HCI'01, Lille, France, 2001, p.297-310

Brockmann (1990)

Brockmann, R. J.: The why, where and how of minimalism. In: ACM SIGDOC Asterisk Journal of Computer Documentation 14(4), 1990, p.111 - 119.

Brunnberg (2002)

Brunnberg, L.: Backseat Gaming: exploration of mobile properties for fun. In: CHI 2002, Minneapolis, USA, 2002,

Brusilovsky (1999)

Brusilovsky, P.: Adaptive and Intelligent Technologies for Web-based Education. In: KI - Künstliche Intelligenz 13(4), 1999, p.19-25.

Burkey (2000)

Burkey, C.: Environmental Interfaces: HomeLab. In: Conference on Human Factors in Computing Systems - CHI'02, The Hague, Netherlands, 2000, p.47-48

Buyukkokten et al. (2000a)

Buyukkokten, O.; Garcia-Molina, H.; Paepcke, A.: Focused Web Searching with PDAs. In: Proceedings of the 9th International WWW Conference(WWW9), Amsterdam, Netherlands, 2000a, p.213-230

Buyukkokten et al. (2000b)

Buyukkokten, O.; Molina, H. G.; Paepcke, A.; Winograd, T.: Power Browser: Efficient Web Browsing for PDAs. In: Proceedings of the Conference on Human Factors in Computing Systems - CHI'00, The Hague, Netherlands, 2000b,

Byrne et al. (1999)

Byrne, M. D.; John, B. E.; Wehrle, N. S.; Crow, D. C.: The tangled Web we wove: a taskonomy of WWW use. In: Proceedings of the Conference on Human Factors in Computing Systems - CHI'99, Pittsburgh, USA, 1999, p.544-551

Carroll et al. (1988)

Carroll, J.; Aaronson, A.: Learning by doing with simulated intelligent help. In: Communications of the ACM 31(9), 1988, p.1064 - 1079.

Carroll (1990)

Carroll, J. M.: The Nurnberg Funnel: Designing Minimalist Instructions for Practical Computer Skill. MIT Press, Cambridge, USA 1990

Carroll et al. (1984)

Carroll, J. M.; Mack, R. L.: Learning to use a word processor: By doing, by thinking, and by knowing. In: *Thomas, J.C.; Schneider, M.L.(Ed.)*: Human factors in computer systems, p.13-51, Ablex Publishing, Norwood, USA 1984

Cas (2002)

Cas, J.: UC - Ubiquitous Computing oder Ubiquitous Control? In: *Britzelmaier, B.(Ed.)*: Der Mensch im Netz - Ubiquitous Computing, Teubner, Stuttgart 2002

Charney et al. (1988)

Charney, D.; Reder, L. M.; Wells, G.: Studies of elaboration in instructional text. In: *Doheney-Farina, S.(Ed.)*: Effective documentation: what we have learned from research, p.47-72, MIT Press, Cambridge, USA 1988

Chen et al. (2000)

Chen, G.; Kotz, D.: A Survey of Context-Aware Mobile Computing Research. Dartmouth Computer Science Technical Report TR2000-381, Hanover, USA 2000

Choi (1993)

Choi, J.: Experience-Based Learning In Deductive Reasoning Systems. Dissertation at State University of New York at Buffalo, 1993

Chopra et al. (2001)

Chopra, V.; Zoran, Z.; Damschen, G.; Chris Dix; Cauldwell, P.; Chawla, R.; Saunders, K.; Olander, G.; Norton, F.; Hong, T.; Ogbuji, U.; Richman, M. A.: Professional XML Web Services. Wrox Press, Birmingham, Great Britain 2001

Costabile (2002)

Costabile, M. F.: End-User Development - Empowering people to flexibly employ advanced information and communication technology. In: 1st EUD-Net Workshop, Pisa, Italy, 2002,

Costabile et al. (2003)

Costabile, M. F.; Piccinno, A.; Fogli, D.; Mussio, P.: Software Shaping Workshops: Environments to Support End-User Development. In: Proceedings of the CHI'03 Workshop on End-User Development, Fort Lauderdale, USA, 2003, p.19-22

Covi et al. (1995)

Covi, L. M.; Ackerman, M. S.: Such easy-to-use systems!: How organizations shape the design and use of online help systems. In: Proceedings of the Conference on Organizational Computing Systems, Milpitas, USA, 1995,

c't (2003)

c't. Rad am Draht - Innovationslawine in der Autotechnik. http://www.heise.de/ct/03/14/170/default.shtml, 2003, Accessed on 12.12.2003

Daimler-Chrysler (2002)

Daimler-Chrysler. Mercedes-Benz Portal. http://www.mercedes-benz.t-online.de, 2002, Accessed on 27.06.2003

Daimler-Chrysler (2003)

Daimler-Chrysler. Mercedes Benz TeleAid. http://www.mercedes-benz.com/d/innovation/rd/teleaid.htm, 2003, Accessed on 27.06.2003

Dey et al. (1999)

Dey, A. K.; Abowd, G. D.: Towards a better understanding of Context and Context-Awareness. In: Proceedings of Handheld and Ubiquitous Computing (HUC'99), Karlsruhe, Germany, 1999, p.304-308

Dey et al. (2001)

Dey, A. K.; Ljungstrand, P.; Schmidtg, A.: Distributed and Disappearing User Interfaces in Ubiquitous Computing. In: Conference on Human Factors in Computing Systems - CHI'01, Seattle, USA, 2001, p.487 - 488

Dick et al. (1996)

Dick, W.; Carey, L.: The systematic design of instruction. 4th ed. Harper Collins College Publishers, New York, NY, USA 1996

Dillon et al. (1990)

Dillon, A.; Richardson, J.; McKnight, C.: The Effect of Display Size and Text Splitting on Reading Lengthy Text from the Screen. In: Behaviour and Information Technology 9(3), 1990, p.215-227.

Duchnicky et al. (1983)

Duchnicky, R. L.; Kolers, P.: Readability of text scrolled on visual display terminals as a function of window size. In: Human Factors 25(6), 1983, p.683-692.

Dunlop et al. (2002)

Dunlop, M.; Brewster, S.: The Challenge of Mobile Devices for Human Computer Interaction. In: Personal and Ubiquitous Computing 6(4), 2002, p.235-236.

Duri et al. (2002)

Duri, S.; Gruteser, M.; Liu, X.; Moskowitz, P.; Perez, R.; Singh, M.; Tang, J.-M.: Framework for Security and Privacy in Automotive Telematics. In: International Conference on Mobile Computing and Networking. Proceedings of the 2nd international workshop on Mobile commerce, Atlanta, USA, 2002, p.25-32

ExtremeTech.com (2003)

ExtremeTech.com. Visual Studio .NET Beta 2. http://www.extremetech.com/article2/0,3973,1153298,00.asp, 2003, Accessed on 30.11.2003

Farkas (1998)

Farkas, D. K.: Layering as a Safety Net for Minimalist Documentation. In: *Carroll, J.M.(Ed.)*: Minimalism Beyond the Nurnberg Funnel, p.247-274, MIT Press, Cambridge, USA 1998

Feldbusch et al. (2002)

Feldbusch, F.; Paar, A.; Odendahl, M.; Ivanov, I.: A Bluetooth Remote Control System. In: Proceedings of ARCS, 2002, p.241-255

FleetNet (2003)

FleetNet. inter-vehicle communication. www.fleetnet.de, 2003, Accessed on 27.06.2003

Fraunhofer FOKUS (2003)

Fraunhofer FOKUS. Interc@r. http://www.fokus.fraunhofer.de/research/cc/cats/projects/content.html.orig#INTERCAR, 2003, Accessed on 27.06.2003

Friedrich (1990)

Friedrich, J.: Adaptivität und Adaptierbarkeit informationstechnischer Systeme in der Arbeitswelt - zur Sozialverträglichkeit zweier Paradigmen. In: Proceedings der 20. GI-Jahrestagung, Stuttgart, Germany, 1990, p.178 - 191

Fujitsu Siemens Computers (2004)

Fujitsu Siemens Computers. Pocket LOOX. http://www.fujitsu-siemens.com/rl/products/handhelds/pocketloox_detail.html, 2004, Accessed on 05.01.2004

Gagne et al. (1979)

Gagne, R. M.; Briggs, L. J.: Principles of Instructional Design. Holt, Rinehart and Winston, New York 1979

Gerstner (1998)

Gerstner, L. Cebit'98 Keynote address. http://www.ibm.com/lvg/keynote.phtml, 1998, Accessed on 10.10.2003

Gery (1995)

Gery, G.: Attributes and Behaviors of Performance-Centered Systems. In: Performance Improvement Quarterly 8(1), 1995, p.47-93.

GM Corporation (2003a)

GM Corporation. OnStar to Increase Product Availability Of Its Safety, Security and Information Services. http://onstar.internetpressroom.com/PressReleaseDetail.cfm?id=222, 2003a, Accessed on 02.07.2003

GM Corporation (2003b)

GM Corporation. OnStar.com. http://www.onstar.com/, 2003b, Accessed on 27.06.2003

Goodwin et al. (1992)

Goodwin, C.; Duranti, A.: Rethinking context: an introduction. In: *Duranti, A.; Goodwin, C.(Ed.)*: Rethinking context: language as an interactive phenomenon, Cambridge University Press, 1992

Gray et al. (2001)

Gray, P.; Salber, D.: Modelling and Using Sensed Context Information in the Design of Interactive Applications. In: *Reed, M.; Nigay, L.(Ed.)*: Proceedings of 8th IFIP International Conference - EHCI 2001, Toronto, Canada, 2001, p.317-335

Haniff et al. (1999)

Haniff, D. J.; Baber, C.; Edmondson, W.: Human Factors of Multi-modal Ubiquitous Computing. In: Handheld and Ubiquitous Computing, First International Symposium - HUC'99, Karlsruhe, Germany, 1999, p.346-348

Hansmann (2001)

Hansmann, U.: Pervasive computing handbook. Springer, Berlin u.a. 2001

Hirschmann (2003)

Hirschmann. Car Communication Systems. http://www.hirschmann.com/deutsch/bereiche/ccs/index.html, 2003, Accessed on 27.06.2003

Hu et al. (2003)

Hu, J.; Feijs, L.: An Agent-based Architecture for Distributed Interfaces and Timed Media in a Storytelling Application. In: Proceedings of AAMAS'03, Melbourne, Australia, 2003,

Huber et al. (2001)

Huber, W.; Lädke, M.; Ogger, R.: Extended Floating-Car Data For The Acquisistion Of Traffic Information. In: 6th ITS World Congress, Toronto, 2001,

IBM (2003)

IBM. WebSphere Studio Device Developer. http://www.ibm.com/software/wireless/wsdd/, 2003, Accessed on 15.02.2004

Jonassen et al. (1993)

Jonassen, D. H.; Wang, S.: Acquiring Structural Knowledge from Semantically Structured Hypertext. In: Journal of Computer-Based Instruction 20(1), 1993, p.1-8.

Jones et al. (1999)

Jones, M.; Marsden, G.; Mohd-Nasir, N.; Booner, K.: Improving Web Interaction on Small Display. In: Proceeding of the 8th International Conference on World Wide Web, Toronto, Canada, 1999, p.51-59

Kearsley (1988)

Kearsley, G.: Online Help Systems: Design and Implementation. Norwood, USA, Ablex Pub. Corp 1988

Kim et al. (2001)

Kim, L.; Albers, M. J.: Web Design Issues when Searching for Information in a Small Screen Display. In: Proceedings of the 19th annual international conference on Computer documentation - SIGDOC'01, Santa Fe, USA, 2001,

Kirste et al. (2001)

Kirste, T.; Rapp, S.: Architecture for Multimodal Interactive Assistant Systems. In: Statustagung der Leitprojekte "Mensch-Technik-Interaktion", Saarbrücken, Germany, 2001, p.111-115

Klann et al. (2004)

Klann, M.; ????: ???? In: *Lieberman, H.; Paternó, F.; Wulf, V.(Ed.)*: End User Development, Kluwer, in Press, 2004

Klann et al. (2003)

Klann, M.; Eisenhauer, M.; Oppermann, R.; Wulf, V.: Shared initiative: Cross-fertilisation between system adaptivity and adaptability. 2003

Kolodner (1983)

Kolodner, J.: Reconstructive Memory: A Computer Model. In: Cognitive Science 7(4), 1983,

Köster (2000)

Köster, A.: Modellbildung und Simulation unter Verwendung wissensbasierter Konzepte. Fakultät für Informatik, Universität der Bundeswehr, Munich, Germany. Technischer Bericht Nr. 2000-04 2000

Koyani et al. (2003)

Koyani, S. J.; Balley, R. W.; Nall, J. R. Research-Based Web design & Usability Guidelines. http://usability.gov/pdfs/, 2003, Accessed on 02.02.2003

Laarni (2002)

Laarni, J.: Searching for Optimal Methods of Presenting Dynamic Text on Different Types of Screens. In: NordicCHI, Århus, Denmark, 2002,

Lacey (1996)

Lacey, D. W. How to Create Manuals. http://users.zoominternet.net/~dwlacey/man-bro.htm, 1996, Accessed on 15.01.2003

Landay (2002)

Landay, J. A.: Informal Tools for Designing Anywhere, Anytime, Anydevice User Interfaces. In: Diagrams 2002 - Second International Conference on Theory and Application of Diagrams, Callaway Gardens, USA, 2002, p.359

Leake (1996)

Leake, D. B.: CBR in Context: The Present and Future. In: *Leake, D.B.(Ed.)*: Case-Based Reasoning: Experiences, Lessons, and Future Directions, AAAI Press/MIT Press, Menlo Park, USA 1996

Lebowitz (1983)

Lebowitz, M.: Memory-Based Parsing. In: Artificial Intelligence 21(4), 1983, p.363-404.

Lieberman et al. (2003)

Lieberman, H.; Paternò, F.; Repenning, A.; Wulf, V.: Perspectives on end user development. In: Conference on Human Factors in Computing Systems - CHI'03, Ft. Lauderdale, USA, 2003, p.1048-1049

Lieberman et al. (2000)

Lieberman, H.; Selker, T.: Out of Context: Computer Systems that Adapt to, and Learn from, Context. In: IBM Systems Journal 39(3), 2000, p.617-631.

Lueg (2002)

Lueg, C.: On the Gap between Vision and Feasibility. In: Proceedings of the International Conference on Pervasive Computing - Pervasive 2002, Zurich, Switzerland, 2002, p.45-57

Marcus et al. (2002)

Marcus, A.; Chen, E.: Designing the PDA of the Future. In: interactions 9(1), 2002, p.34-44.

Marsden et al. (2001)

Marsden, G.; Jones, M.: Ubiquitous Computing and Cellular Handset Interfaces – are menus the best way forward? In: Proceedings of South African Institute of Computer Scientists and Information Technologists Annual Conference 2001, Pretoria, South Africa, 2001, p.111-119

Martens (2003a)

Martens, A.: Discussing the ITS Architecture - A Proposal for Case-Based ITS. In: GI-Workshop "Expressive Media and Intelligent Tools for Learning", 26th German Conference on Artificial Intelligence KI-2003., Hamburg, Germany, 2003a, p.3-10

Martens (2003b)

Martens, A.: Discussing the Support of Cognitive Processes in Case-Based ITS. In: GI-Workshop "Expressive Media and Intelligent Tools for Learning", 26th German Conference on Artificial Intelligence KI-2003, Hamburg, Germany, 2003b, p.49-56

Meij et al. (1998)

Meij, H. V. d.; Carroll, J. M.: Principles and heuristics for designing minimalist instruction. In: *Carroll, J.M.(Ed.)*: Minimalism beyond the Nurnberg funnel, p.19 -53, MIT Press, Cambridge, USA 1998

Michalski et al. (1983)

Michalski, R.; Carbonell, J.; Mitchell, T.: Machine Learning: An Artificial Intelligence Approach. Tioga Publishing Company, California, USA 1983

Microsoft (1995)

Microsoft: The Windows Interface Guidelines for Software Design. Microsoft Press, Redmond, WA, USA 1995

Microsoft (2001a)

Microsoft. Microsoft Inductive User Interface Guidelines. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwui/html/iuiguidelines.asp, 2001a, Accessed on 05.01.2004

Microsoft (2001b)

Microsoft. Windows XP Visual Guide. http://www.microsoft.com/whdc/hwdev/windowsxp/downloads/default.mspx, 2001b, Accessed on 12.12.2003

Microsoft (2002)

Microsoft. Case Study: computerjobs.com - Realizing the Benefits of .NET. http://www.asp.net/faq/ComputerJobsCFtoASPNET.pdf, 2002, Accessed on 30.11.2003

Microsoft (2003a)

Microsoft. Microsoft Visual Studio. http://msdn.microsoft.com/vstudio/, 2003a, Accessed on 30.11.2003

Microsoft (2003b)

Microsoft. Microsoft Windows CE 3.0 - Designing a User Interface for Windows CE. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wcedesgn/htm/_wcesdk_Designing_a_User_Interface_for_Windows_CE.asp, 2003b, Accessed on 05.01.2004

Microsoft MSDN (2004)

Microsoft MSDN. eMbedded Visual Tools. http://msdn.microsoft.com/vstudio/device/embedded/, 2004, Accessed on 15.02.2004

Microsoft MSDN News (2001)

Microsoft MSDN News. Talking to Rob Brigham. http://msdn.microsoft.com/msdnnews/2001/july/TalkingTo/TalkingTo.asp, 2001, Accessed on 30.11.2003

Mirel (1998)

Mirel, B.: Minimalism for complex tasks. In: *Carroll, J.M.(Ed.)*: Minimalism beyond the Nurnberg Funnel, p.179-218, MIT Press, Cambridge, USA 1998

MobileAria Inc. (2003)

MobileAria Inc. Automotive Telematics. http://www.mobilearia.com/auto/index.shtml, 2003, Accessed on 27.03.2003

Mørch et al. (2004)

Mørch, A.; Stevens, G.; Won, M.; Klann, M.; Dittrich, Y.; Wulf, V.: Component-Based Technologies for End-User Development: Vision, Reality, and New Directions. In: CACM Special Issue on EUD????(2004,

Morgan et al. (2002)

Morgan, A. P.; Cafeo, J. A.; Gibbons, D. I.; Lesperance, R. M.; Sengir, G. H.; Simon, A. M.: The General Motors Variation-Reduction Adviser: An Example of Grassroots Knowledge Management Development. In: Practical Aspects of Knowledge Management. 4th International Conference, PAKM 2002, Vienna, Austria, 2002, p.137-143

Morgan et al. (2003)

Morgan, A. P.; Cafeo, J. A.; Gibbons, D. I.; Lesperance, R. M.; Sengir, G. H.; Simon, A. M.: The General Motors Variation-Reduction Adviser: Evolution of a CBR System. In: Case Based Reasoning Research and Development. 5th Inter-

national Conference on Case-Based Reasoning, ICCBR 2003, Trondheim, Norway, 2003, p.306-318

Morkes et al. (1997)

Morkes, J.; Nielsen, J. Concise, SCANNABLE, and Objective: How to write for the Web. http://www.useit.com/papers/webwriting/writing.html, 1997, Accessed on 01.10.2003

Myers et al. (2002)

Myers, B. A.; Nichols, J.: Communication Ubiquity Enables Ubiquitous Control. In: Boaster for Human-Computer Interaction Consortium - HCIC'2002, Winter Park, USA, 2002,

MySQL AB (2004)

MySQL AB. www.mysql.com, 2004, Accessed on 05.01.2004

Naranyanan et al. (1998)

Naranyanan, N. H.; Hegarthy, M.: On designing comprehensible interactive hypermedia manuals. In: International Journal of Human-Computer Studies 48(2), 1998, p.267-301.

Nichols (2001)

Nichols, J. W.: Using Handhelds as Controls for Everyday Appliances: A Paper Prototype Study. In: Proceedings of CHI'2001, Seattle, USA, 2001,

Nichols et al. (2002a)

Nichols, J. W.; Myers, B.; Harris, T. K.: Requirements for Generating Multi-Modal Interfaces for Complex Appliances. In: IEEE Fourth International Conference on Multimodal Interfaces, Pittsburgh, USA, 2002a, p.377-382

Nichols et al. (2002b)

Nichols, J. W.; Myers, B.; Higgins, M.; Hughes, J.; Harris, T. K.; Rosenfeld, R.; Pignol, M.: Generating Remote Interfaces for Complex Appliances. In: CHI Letters: ACM Symposium on User Interface Software and Technology - UIST'02, Paris, France, 2002b, p.161-170

Nielsen (1993)

Nielsen, J.: Usability Engineering. Academic Press, San Diego 1993

Nielsen (2000)

Nielsen, J.: Designing Web Usability. New Riders Publishing, Indianapolis, USA 2000

Nielsen et al. (1994)

Nielsen, J.; Sano, D.: SunWeb: User Interface Design for Sun Microsystem's Internal Web. In: Proceedings of the 2nd World Wide Web Conference '94: Mosaic and the Web, Chicago, USA, 1994,

Nitschke et al. (2001)

Nitschke, J.; Wandke, H.: Human Support as a Model for Assistive Technology. In: *Heuer, A.; Kirste, T.(Ed.)*: Intelligent Interactive assistance and Mobile Multimedia Computing, p.130-136, Neuer Hochschulschriftenverlag, Rostock 2001

Norman (1998)

Norman, D.: The Invisible Computer: Why Good Products Can Fail, the Personal Computer Is So Complex, and Information Appliances Are the Solution. MIT Press, 1998

Oppermann (1994a)

Oppermann, R.: Adaptive User Support. Ergonomic Design of Manually and Automatically Adaptable Software. Lawrence Erlbaum Associates, New Jersey, USA 1994a

Oppermann (1994b)

Oppermann, R.: Adaptively supported Adaptability. In: International Journal of Human-Computer Studies 40(3), 1994b, p.554-472.

Oppermann (2001)

Oppermann, R. Einführung in die Software-Ergonomie - Hilfesysteme. http://www.uni-koblenz.de/~oppi/SE-EinfuehrungInPortionen/SE08-Hil.pdf, 2001, Accessed on 31.11.2003

Oppermann et al. (2001)

Oppermann, R.; Specht, M.: Contextualized Information Systems for an Information Society for All. In: Universal Access in HCI: Towards an Information Society for All. HCI International, New Orleans, USA, 2001, p.850 - 853

ParallelGraphics (2004)

ParallelGraphics. Virtual Manuals. http://www.parallelgraphics.com/showroom/virtual-manuals/, 2004, Accessed on 15.02.2004

Pascoe (1998)

Pascoe, J.: Adding Generic Contextual Capabilities to Wearable Computers. In: Proceedings of 2nd International Symposium on Wearable Computers, Pittsburgh, USA, 1998, p.92-99

Patel et al. (2001)

Patel, A.; Kinshuk; Russell, D.; Oppermann, R.: Intelligent Tutoring Systems: Confluence of Information Science and Cognitive Science. In: Proceedings of Informing Science Conference 2001, Krakow, Poland, 2001,

Patel et al. (1998)

Patel, A.; Russell, D.; Kinshuk; Oppermann, R.; Rashev, R.: An initial framework of contexts for designing usable intelligent tutoring systems. In: Fraunhofer Institut für Angewandte Informationstechnik FIT, Information Services and Use 18(1-2), 1998, p.65-76.

Peylo et al. (2000)

Peylo, C.; Thelen, T.: Skills und Konzepte als Grundlage für Benutzermodellierung in einem Prolog-ITS. In: *Müller, M.(Ed.)*: 8. GI-Workshop Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen - ABIS 2000, Osnabrück, Germany 2000

Pham (2001)

Pham, T. L.: Composite device computing environment. Shaker, Aachen 2001

PocketSOAP (2003)

PocketSOAP. PocketSOAP - Web Services on the move. http://www.pocketsoap.com/pocketsoap/, 2003, Accessed on 05.01.2004

Price (1984)

Price, J.: How to write a computer manual. The Benjamin/Cummings Publishing Company, Menla Park, USA 1984

Randall et al. (1998)

Randall, N.; Pedersen, I.: Who exactly is trying to help us? The ethos of help systems in popular computer applications. In: Proceedings of the 16th annual international conference on Computer documentation, Quebec, Canada, 1998,

Raskin (1994)

Raskin, J.: Intuitive equals Familiar. In: Communications of the ACM 9(37), 1994, p.17.

Reeves et al. (1996)

Reeves, B.; Nass, C.: The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places. CSLI Publications and Cambridge University Press, Cambridge, Great Britain 1996

Rettig (1991)

Rettig, M.: Nobody reads documentation. In: Communications of the ACM 34(7), 1991, p.19-24.

Richter (1992)

Richter, M.: Prinzipien der künstlichen Intelligenz: Wissensrepräsentation, Inferenz und Expertensysteme. Teubner, Stuttgart 1992

Russell (2001)

Russell, J.: Book Metaphor: Friend or Foe? In: Proceedings of the 19th annual international conference on Computer documentation - SIGDOC'01, Santa Fe, USA, 2001,

Satyanarayanan (2001)

Satyanarayanan, M.: Pervasive Computing: Vision and challenges. In: IEEE Personal Communications 10(17), 2001,

Schank (1982)

Schank, R.: Dynamic Memory: A Theory of Learning in Computers and People. Cambridge University Press, New York, USA 1982

Schilit et al. (1994a)

Schilit, B. N.; Adams, N.; Want, R.: Context-Aware Computing Applications. In: Proceedings of the IEEE Computer Society Workshop on Mobile Computing Systems and Applications, Santa Cruz, USA, 1994a,

Schilit et al. (1994b)

Schilit, B. N.; Theimer, M.: Disseminating Active Map Information to Mobile Hosts. In: IEEE Network 8(5), 1994b, p.22-32.

Schmidt et al. (1998)

Schmidt, A.; Beigl, M.; Gellersen, H.-W.: There is more to Context than Location. In: Proceedings on International Workshop on Interactive Applications of Mobile Computing - IMC '98, Rostock, Germany, 1998,

Schulenburg (1995)

Schulenburg, D. A.: Learning and Using Context in a Connectionist Model of NLU. In: IJCAI-95 Workshop on Context in Natural Language Processing, Montreal, Canada, 1995,

Selker (1994)

Selker, T.: COACH: a teaching agent that learns. In: Communications of the ACM 37(7), 1994, p.92 - 99.

Siemens (2002)

Siemens. SIESaR - Ferndiagnose und -wartung für Fahrzeuge aller Art. http://w4.siemens.de/ct/de/technologies/ib/siesar.html, 2002, Accessed on 2.7.2003

Siemens (2003)

Siemens. Siemens S55 Handbuch. http://www.join-mms.de/handbuch/siemens/s55.pdf, 2003, Accessed on 30.11.2003

Silveira et al. (2001)

Silveira, M. S.; Souza, C. S. d.; Barbosa:, S. D. J.: Semiotic engineering contributions for designing online help systems. In: Proceedings of the 19th annual international conference on Computer documentation - SIGDOC'01, Santa Fe, USA, 2001,

Simon (1983)

Simon, H. A.: Why should machines learn? In: *Michalski, R.; Carbonnel, J.; Mitchell, T.(Ed.)*: Machine Learning: An Artificial Intelligence Approach, Palo Alto, USA 1983

Smart et al. (1998)

Smart, K. L.; DeTienne, K. B.; Whiting, M.: Customers' use of documentation: the enduring legacy of print. In: Proceedings of the 16th annual international conference on Computer documentation, Quebec, Canada, 1998,

Solem (1986)

Solem, A.: Designing computer documentation that will be used: Understanding computer user attitudes. In: Proceedings of the 4th annual international conference on Systems documentation, Ithaca, USA, 1986, p.55-56

Specht et al. (2002a)

Specht, M.; Kravcik, M.; Klemke, R.; Pesin, L.: Information Brokering for the Adaptive Learning Environment. In: Proceedings of the e-Learn 2002 Conference, Montreal, Canada, 2002a,

Specht et al. (2002b)

Specht, M.; Kravcik, M.; Klemke, R.; Pesin, L.; Hüttenhain, R.: Adaptive Learning Environment for Teaching and Learning in WINDS. In: Proceedings of the

2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems - AH 2002, Malaga, Spain, 2002b, p.572-575

Spyridakis (2000)

Spyridakis, J. H.: Guidelines for Authoring Comprehensible Web Pages and Evaluating Their Success. In: Technical Communication 47(4), 2000, p.355-367.

Stajanao (2002)

Stajanao, F.: Security for ubiquitious computing. Wiley, Chichester 2002

Steehouder et al. (2000)

Steehouder, M.; Karremann, J.; Ummelen, N.: Making Sense of Step-by-Step Procedures. In: Proceedings of the 18th annual ACM international conference on Computer documentation: technology & teamwork, Cambridge, USA, 2000,

Stevens (2002a)

Stevens, G.: Komponentenbasierte Anpassbarkeit - FlexiBeans zur Realisierung einer erweiterten Zugrifsskontrolle. Diploma Thesis, Department of Computer Science, University of Bonn, Germany 2002a

Stevens (2002b)

Stevens, G.: Komponentenbasierte Anpassbarkeit - FlexiBeans zur Realisierung einer erweiterten Zugrifsskontrolle. Diploma Thesis, Institut für Informatik, Abteilung III, Rheinische Friedrich-Wilhelms-Universität Bonn 2002b

Stiemerling (2000)

Stiemerling, O.: Component-based Tailorability. Ph.D. Thesis, Department of Computer Science, University of Bonn, Germany 2000

Sun Microsystems (2004)

Sun Microsystems. Java 2 Platform, Micro Edition (J2ME). http://java.sun.com/j2me/, 2004, Accessed on 15.02.2004

SuperWaba (2004)

SuperWaba. superwaba. http://www.superwaba.com.br/en, 2004, Accessed on 15.02.2004

Systinet Corporation (2004)

Systinet Corporation. WASP Product Suite. http://www.systinet.com/products/overview, 2004, Accessed on 15.02.2004

Telematics Research Group (2003a)

Telematics Research Group. Building Telematics in North America. http://www.telematicsresearch.com/, 2003a, Accessed on 03.07.2003

Telematics Research Group (2003b)

Telematics Research Group. Enabling Wireless Devices and the Impact of Automotive Telematics. http://www.telematicsresearch.com/, 2003b, Accessed on 3.7.2003

The Apache Software Foundation (2003)

The Apache Software Foundation. Apache Web Services Project. http://ws.apache.org/axis/, 2003, Accessed on 05.01.2004

The Apache Software Foundation (2004)

The Apache Software Foundation. The Apache Jakarta Project. http://jakarta.apache.org/tomcat/, 2004, Accessed on 05.02.2004

uidesign.net (2000)

uidesign.net. The Myth of a Big Screen. http://www.uidesign.net/2000/opinion/bigscreen.html, 2000, Accessed on 05.01.2004

Ummelen (1997)

Ummelen, N.: Declarative information in software manuals: what's the use? In: Proceedings of the 15th annual international conference on Computer documentation, Salt Lake City, USA, 1997,

Uther (2002)

Uther, M.: Mobile Internet usability: What can 'Mobile Learning' learn from the past? In: Proceedings of the IEEE International Workshop on Wireless and Mobile Technologies in Education - WMTE'02, Växjö, Sweden, 2002,

Volkswagen AG (2003)

Volkswagen AG. Volkswagen Service. http://www.vw-service.de/index.htm, 2003, Accessed on 27.06.2003

W3C Consortium (2003a)

W3C Consortium. SOAP Version 1.2. http://www.w3.org/TR/soap12-part1/, 2003a, Accessed on 05.01.2004

W3C Consortium (2003b)

W3C Consortium. Web Services Activity. http://www.w3.org/2002/ws/, 2003b, Accessed on 05.01.2003

Want et al. (1992)

Want, R.; Hopper, A.; Falcao, V.; Gibbons, J.: The Active Badge Location System. In: ACM Transactions on Information Systems 10(1), 1992, p.91-102.

Weiser (1991)

Weiser, M.: The Computer for the 21st Century. In: Mobile Computing and Communications Review 3(3), 1991,

Weiser et al. (1995)

Weiser, M.; Brown, J. S.: Designing Calm Technology. In: ?? 1995, p.??

Weiss (2002)

Weiss, S.: Handheld Usability. John Wiley, Chichester 2002

Wheeler (2002)

Wheeler, S. How the Auto Industry Should Embrace CRM. http://www.strategy-business.com/press/enewsarticle/?art=330283&pg=0, 2002, Accessed on 2.7.2003

Wulf (2000)

Wulf, V.: Exploration Environments: Supporting Users to Learn Groupware Functions. In: Interacting with Computers 13(2), 2000, p.265-299.

Wulf et al. (2001)

Wulf, V.; Golombek, B.: Direct Activation: A Concept to Encourage Tailoring Activities. In: Behaviour & Information Technology 20(4), 2001, p.249 - 263.

Yates et al. (2003)

Yates, A.; Etzioni, O.; Weld, D.: A Reliable Natural Language Interface to Household Appliances. In: Proceedings of International Conference on Intelligent User Interfaces - IUI'03, Miami, USA, 2003, p.189 - 196

Zimmermann (2003)

Zimmermann, A.: Context-Awareness in User Modelling: Requirement Analysis for a Case-Based Reasoning Application. In: Case Based Reasoning Research and Development. 5th International Conference on Case-Based Reasoning - ICCBR 2003, Trondheim, Norway, 2003, p.718-732

Zimmermann et al. (2003)

Zimmermann, G.; Nixon, T.; Beard, M.; Sitnik, E.; LaPlant, B.; Trewin, S.; Laskowski, S.; Vanderheiden, G.: Toward a Unified Universal Remote Console Standard. In: Conference on Human Factors in Computing Systems - CHI'03, Ft. Lauderdale, USA, 2003,

Eidesstattliche Erklärung

Hiermit erkläre ich eidesstattlich, dass ich die vorliegende Arbeit selbständig angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht. Ich bin mir bewusst, dass eine unwahre Erklärung rechtliche Folgen haben kann.

Siegen, 24. Februar 2004 Daniel Humberg