

Workshop Proceedings

WAP - Interaktionsdesign und Benutzbarkeit

Workshop auf der Konferenz: Mensch und Computer 2001.

Einführung

Das Wireless Application Protocol (WAP) eröffnet neue Möglichkeiten, aber auch neue Herausforderungen für die Gestaltung und Entwicklung von mobilen, nomadischen Informationssystemen. In diesem Workshop sollen die Stärken und Schwächen von WAP in Bezug auf benutzerorientierte Gestaltung und technische Machbarkeit erörtert werden. Dazu sollen 15 Teilnehmer aus Wissenschaft und Praxis zu Vorträgen und zum Erfahrungs- und Ideenaustausch eingeladen werden. Insgesamt soll in diesem Workshop Wissen zu Interaktionsdesign und Benutzbarkeit von WAP gesammelt werden.

Workshop Moderatoren

Albrecht Schmidt, TecO, Universität Karlsruhe,
Tom Gross, GMD-FIT, Sankt Augustin,
Oliver Frick, SAP AG, CEC Karlsruhe,

albrecht@teco.uni-karlsruhe.de
tom.gross@gmd.de
o.frick@sap.com

Beiträge

Christian Feichtner, Siemens Austria

Approaches to develop services for wired and wireless networks.

Tom Gross, GMD-FIT

PRAVTA - A Light-Weight WAP Awareness Client

Frank Kargl, Torsten Illmann, Alexander Raschke, Stefan Schlott, Uni Ulm

WAPcam - eine multimediale WAP Anwendung in der Studentenausbildung.

Andreas Kramer, Markus Latzina, Jörg Mann, SAP AG, Corporate Research

Navigating Within Tables Used in SAP BW Mobile Reporting (WAP):
Results of Usability Tests

Carsten Magerkurth und Thorsten Prante, GMD-IPSI, Darmstadt

Towards a unifying Approach to Mobile Computing

Holger Nösekabel, Franz Lehner, Universität Regensburg

Bewertung von WAP-Anwendungen - Entwicklung von Kriterien und Erfahrung bei der Qualitätsbewertung in der Praxis

Albrecht Schmidt, Uni Karlsruhe

WAP - Eine offene Plattform die unsere Kommunikation verändert?
Context-Call - ein Ansatz

Michael Semrau, Dresdner Bank

m-Commerce für Finanzdienstleister - eine Chance mit vielen Hindernissen.

Approaches to develop services for wired and wireless networks.

By Christian Feichtner, Siemens Austria, PSE MCS CN4

The number of people who are using the Internet and the World Wide Web has grown dramatically during the past few years. Companies have made considerable efforts to create content and services for the web. However, since the advent of WAP, many of them face the problem of porting services to the WAP platform and will continue to do so with the new generations of mobile wireless devices. This paper focuses on the main problem, to develop services for a wide range of devices having different characteristics like screen size or navigation options and discusses two approaches on how to solve those problems.

1.1 The Dilemma

Developers creating services for the wired and wireless world have to deal with the problem of small displays, limited memory and computing power on the wireless platform versus high resolution displays, huge memory and efficient computing power on wired desktop computers just to name a few. If a service should be available on both platforms it has to be adapted for each possible device it will be used on. This requires developers to create several versions of a service, compatible with the device capabilities of each class of device or even each single device in the worst case. At present, there is no real solution that will enable developers to create a service for the wired and wireless world with a minimum of effort without rewriting the entire service for each platform or device.

1.1.1 HTML vs. WML

For a long time, HTML [1] has been the language which was used to generate services and to design navigation for web content. HTML is a standardized hypertext markup language, derived from SGML and is used to create the content and layout for a web page. Navigation is implemented by providing hyperlinks to the users. A browser takes the HTML file as input and will render the final layout on the screen based on the content and layout information contained in the document. Users then are able to navigate by using a graphical pointing device to click on hyperlinks. Services based on that, are made for rather big screens on powerful computers. One disadvantage of HTML is, that it is not XML compliant, which is becoming more and more important today as XML is standardized and allows a variety of automatic processing options.

Totally contrary to the web, services for mobile devices have to be small and must run on devices with limited screen capabilities and power supply, offering only a few keys for input. The WAP Forum [3] is responsible for developing the WAP architecture along with the markup language used to design services. This markup language, called wireless markup language (WML), though it offers quite similar elements to HTML, is incompatible with HTML. WML is fully XML [2] compliant and offers additional tags especially designed to ease the layout and the navigation on mobile devices.

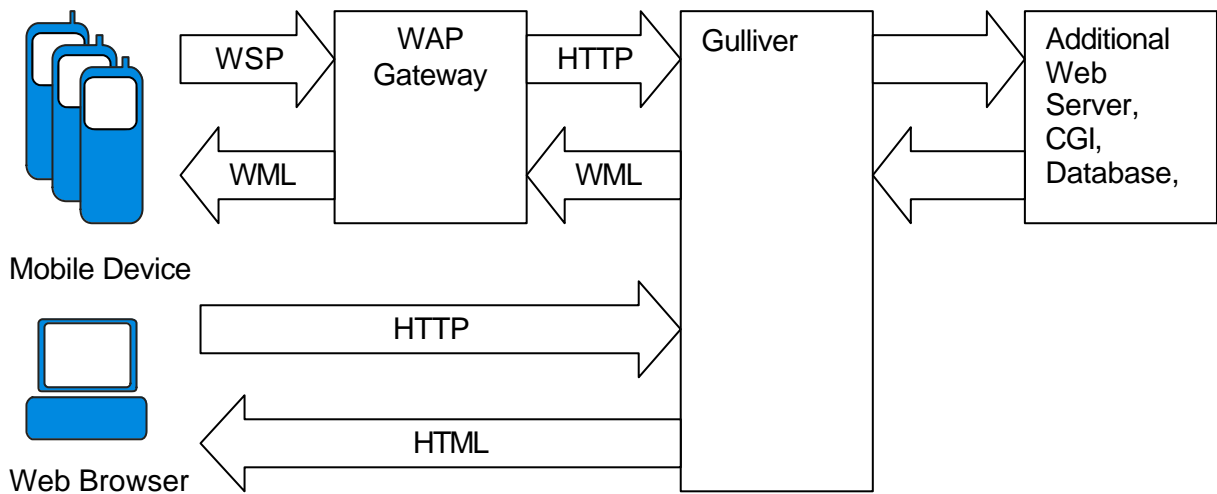
The problem faced by service developers is how to reuse existing services and content in both worlds, as the wireless world undoubtedly will become more and more important. Currently developers have to re-write services in order to work on small, wireless devices. It is also not recommendable to match navigation patterns from the wired world like the world wide web to the wireless WAP world. So a solution is needed, which will allow developers to create services for both worlds with a minimum of effort.

1.2 An approach for device-independent service provisioning: The Gulliver Project

"Gulliver" is a research project cooperation, currently being in it's final phase, between Siemens PSE and the University of Linz, Austria. Gulliver, which is an open platform addresses the following issues:

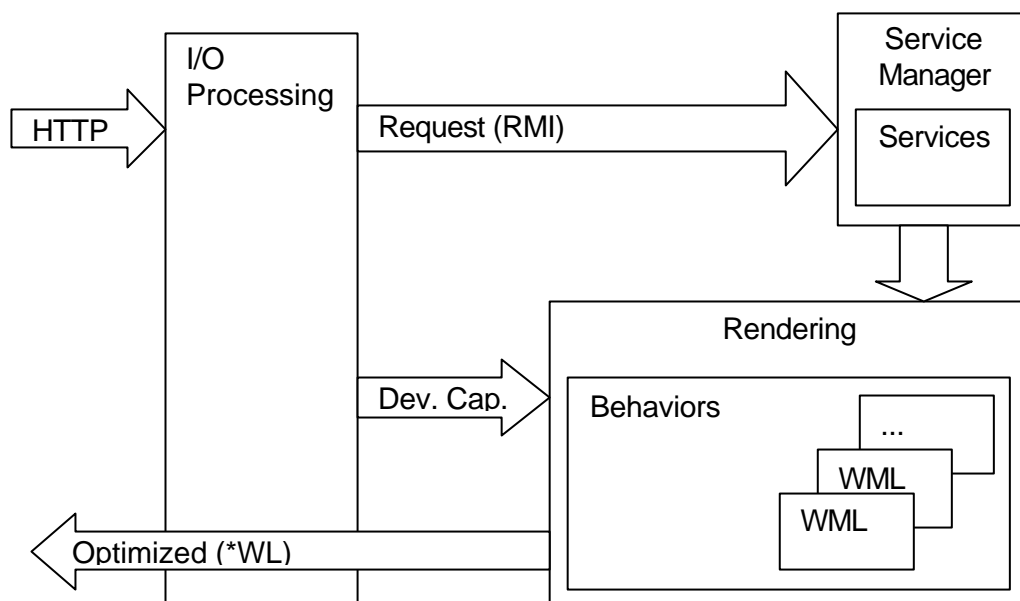
- Man-Machine-Interaction is device specific
 - Hardware (Graphical, Keys only, ...), Software (Browser Version, ...)
- Context aware content (time and location)
- User Profiles
 - Personal Interests, Ad-Hoc Meetings, Age, Language
- The data transmission medium is taken into account
- QoS Parameters

The Gulliver platform allows to design and to integrate services using a common language. The following graphics shows the generic Gulliver architecture.



Picture 1: Gulliver overall architecture

Devices using a Gulliver service, communicate with the Gulliver platform the standardized HTTP protocol. The output being sent to the device is dependent on the device characteristics, which have been determined. The following picture is a closer look into the Gulliver box from the above picture.



A HTTP request for a service is received by the I/O processing unit. This unit forwards the request via RMI to the service manager and the service. The service generates output which is passed to the rendering unit, which itself contains behaviors for several devices. Using these, the renderer generates optimized *ML (WML, HTML, XHTML, ...) output, which is then sent to the device. The I/O processing part passes the optimized *ML response through to the client.

Gulliver has successfully been presented using a demo service called "Vienna Guided Tour", which also demonstrated a suspend/resume feature allowing a user to suspend the tour at will and to resume it at a later time and incorporates user preferences, which allow the user to specify if he wants to see museums or churches for example.

1.3 Separating Content, Layout and Navigation

A service designed to run with a web browser will probably run, but be difficult to use on a wireless device. To solve this problem, developers should clearly separate a service's content, layout and navigation. This can be achieved by utilizing standards like XML and XML style sheets (XSL) [8]. The content only incorporates the structure, making no implications or definitions on how it should be displayed on a device. XML is utilized to structure the content. The layout defines how the content should be displayed, which may be different for each class of devices like powerful desktop computers or thin wireless devices. A combination of content and layout is what the user actually sees on the screen. The navigation is implemented using hyperlinks, but is also to be regarded separately from the content and layout, which makes it possible to define different navigation patterns for different devices. By combining the content, layout and navigation in a way suitable for a specific device a service becomes truly device independent.

1.3.1 Content: XML

The Extensible Markup Language (XML) [2] is a universal format for structured documents. It allows to structure a document and to store it in a plain text file. This way developers can mark special parts of a document which should be rendered in a special font or color or which should be used for navigation. Using an XML parser, the document can be parsed and the output can be adopted to the capabilities of any device. Several web sites already demonstrate this by having the content stored as XML and layout it differently for screen and print.

1.3.2 Layout: XSL

The Extensible Stylesheet Language (XSL) [8] allows to transform a XML document into an other XML compliant document. XSL hereby contains the definitions on how the transformation has to take place. A XML/XSL processor takes both, the XML and the XSL file as an input and transforms them into a new document, depending on the transformations defined in the XSL.

1.3.3 Navigation

Unfortunately there is no standardized mechanism to store navigation for a service similar to XML and XSL. However, the navigation can be seen as a frame around the service; a frame around each page of a service if we talk about Web and WAP services. . This view would suggest to also store the navigation separately in order to become optimized for several device classes. At present navigation is also defined using XML and XSL is used to transform the navigation into a proper format for a device class.

1.4 Device capabilities

If content, layout and navigation are separated as proposed in the previous chapter, then we also need a way to determine how to combine them according to the capabilities of the

device. Currently a service can determine the device capabilities by a user-agent string which is sent along with each request and typically contains the a user-agent name, version number and manufacturer. But the service has to know or to retrieve the device capabilities from somewhere based on that user-agent string. The browscap method allows a service to use a common file managed by a single institution to determine the characteristics based on the user agent string. The second method is using a CC/PP profile which allows a device to directly inform the service about it's capabilities.

1.4.1 Browscap File

This file, maintained by CyScape [6], is a plain text file, listing the capabilities for almost any known web browser, in a way known from windows initialization (.ini) files. The file is used together with web servers or CGI scripting languages and can be used to determine the capabilities based on the HTTP_USER_AGENT [7] field, which is transmitted with every HTTP GET request – if it is not filtered by a firewall or similar. This approach requires web server administrators or developers to always ensure they have the latest and correct version of this file, or services may not function correctly.

1.4.2 CC/PP

This approach, which is currently being developed by a working group at the world wide web consortium, is XML based and tries to define a RDF [5] based framework for device profile information. The goal of the CC/PP framework [4] is to specify how client devices express their capabilities and preferences (the user agent profile) to the server that originates content (the origin server). The origin server uses the "user agent profile" to produce and deliver content appropriate to the client device. In addition to computer-based client devices, particular attention is being paid to other kinds of devices such as mobile phones. CC/PP is currently a working draft.

1.5 Conclusion

As discussed in this paper, developers have to separate content from layout and navigation to create services device independently for the wired and wireless world. A standardized markup language along with the mechanisms to define a layout already exist which will aid the user in this task. However, in order to reassemble the content, layout ad navigation according to the device's capabilities, a standardized way to determine the device capabilities would be required. The browscap file method is proprietary and CC/PP is currently in the process of being standardized. Future effort should be put into a common, standardized way, which allows a developer to easily determine the capabilities of a device. Along with new technologies like XML and XSL and the proposed separation of content, layout and navigation and finally the mechanisms to provide device capability negotiation it will be easy to develop cross network services in the near future.

2 References

- [1] World Wide Web Consortium: The Markup Language: <http://www.w3.org/MarkUp/>
- [2] World Wide Web Consortium: Extensible Markup Language: <http://www.w3.org/XML/>
- [3] The WAP Forum: <http://www.wapforum.org>
- [4] World Wide Web Consortium: CC/PP Architecture <http://www.w3.org/Mobile/CCPP/>
- [5] World Wide Web Consortium: Resource Description Format: <http://www.w3.org/RDF/>
- [6] Cyscape: Browscap File: <http://www.cyscape.com/browscap/>
- [7] HTTP 1.1: <http://www.rfc-editor.org/rfc/rfc2616.txt>
- [8] World Wide Web Consortium: Extensible Stylesheet Language <http://www.w3.org/Style/XSL/>

3 Abbreviations

CC/PP	Composite Capabilities/Preference Profiles
CGI	Common Gateway Interface
HTTP	Hypertext Transfer Protocol
RDF	Resource Description Format
SGML	Standard Generalized Markup Language
W3C	World Wide Web Consortium
WAP	Wireless Application Protocol
WWW	World Wide Web
XHTML	Extended Hypertext Markup Language
XML	Extensible Markup Language
XSL	Extensible Style Language

PRAVTA—A Light-Weight WAP Awareness Client

Tom Gross

*GMD—German National Research Center for Information Technology
(tom.gross@gmd.de)*

Abstract. Despite huge progress in information and communication technology it is often difficult to spontaneously contact persons who are at other locations. This is often due to the fact that important information about the persons at the other sites is missing. Users need to know if the potential communication or cooperation partners are present in the system, if they are available, how busy they are, and so forth. Furthermore, users need this information independently of their current location and adapted to their current context. In this paper we introduce the PRAVTA prototype that provides users with awareness about presence, availability, and tasks of other users anytime and anyplace.

1 Introduction

Huge progress in information and communication technology makes it increasingly easy to communicate and cooperate with other persons over distance. Nevertheless, it is sometimes still difficult to spontaneously reach the persons that are needed. This is often due to the fact that important information about persons at other sites is missing. Users need information whether the person to be contacted is present in the system (e.g., does the person have a mobile phone and is it on; or is the person currently working on the PC and equipped for video conferencing), whether the person is available for communication (e.g., does the person want to have a conversation), and whether the topic of the intended conversation does fit the current task context of the person. We, therefore, argue that users need information about the presence, availability, and tasks of their colleagues.

In this paper we present a WAP prototype that allows users to query for these types of information and to enter it about themselves. We will briefly introduce awareness. Then we will present the implementation and functionality of the PRAVTA Prototype.

2 Awareness

In the computer-supported cooperative work (CSCW) community it has been emphasised for years that efficient and effective cooperation requires that the cooperating individuals are well informed [3]. They require awareness—information about the other persons they are cooperating with, about their actions, about shared artefacts, and so forth. If the cooperating individuals are at the same place this information is obvious and can be gathered easily. If the individuals who are at different places have to cooperate technological support is essential. In order to develop technological support we first have to analyse what we exactly mean with awareness, what types of awareness are relevant, and how they can be supported technologically.

Awareness is the up-to-the-minute knowledge of a user about other users and shared artefacts and includes information about the other participants' locations in the shared space, their present and past actions, and their intentions and possible future actions [5]. *Presence awareness* is the most basic information and refers to the pervasive experience of who is around. In the literature it is sometimes called informal awareness or general awareness [1]. It is a prerequisite for spontaneous interaction. *Availability awareness* is more specific information about the current disposition of other users—that is, it informs about the current occupation of the user, the mood, the attention, and emotions of the user. Some authors [e.g., 6] call it social awareness. *Task awareness* refers to knowledge about other users' work contexts. In collaborative settings it is often useful to know if somebody else is working in the same work context at the same time. For instance, if I know that

one or more of the co-authors for a paper are currently also working on the paper, I can easily coordinate with them about our changes to the shared document.

3 The PRAVTA Prototype

The PRAVTA prototype is an example implementation of a more general concept of ubiquitous and context-specific support for presence, availability, and task awareness.

3.1 Concept

The PRAVTA prototype constitutes an elaborated client for an awareness information environment called NESSIE [9]. The concept of NESSIE is based on sensors, events, and indicators. Sensors are associated with actors, shared artefacts, or any other object and capture events related to them. The generated events are sent to the NESSIE server, which stores the events in an event database. Indicators are used to present the events to interested users with adequate access rights. A range of stationary indicators is available for NESSIE ranging from presentations on a computer monitor like pop-up windows, ticker tapes, and 3D graphical presentations in a multi-user environment to presentations in some ambient displays like balloons and a magic bowl. These indicators are very powerful and allow the users to capture the information in the foreground, but also in the background of their attention. However, they are all stationary and have so far only been installed in office environments.

The aim of the PRAVTA prototype was to develop a system that provides users with adequate awareness information anytime and anyplace. The PRAVTA prototype covers the full range of potentially relevant awareness information as discussed above—it can provide users with presence awareness, availability awareness, and task awareness about potential communication and cooperation partners. Users can customise the system in various ways. In particular, they can specify the type of information they want to receive (e.g., logins or full user names) and the type of presentation (e.g., lists, tables, icons). The PRAVTA prototype was implemented for WAP. It runs on small, mobile devices and offers an easy to use interface, where users can receive awareness information and enter information about their current status with only a few clicks.

3.2 Implementation

The PRAVTA prototype is implemented on two layers: the PRAVTA Client that realises the user interface by means of the wireless application protocol (WAP) technology and the PRAVTA Communication Layer. Being based on WAP, the PRAVTA prototype can be accessed from any mobile device that supports WAP such as mobile phones, palmtops, or smart phones. Figure 1 shows the software architecture of the whole system consisting of the PRAVTA Client, the remaining NESSIE components, and the PRAVTA Communication Layer that connects the two.

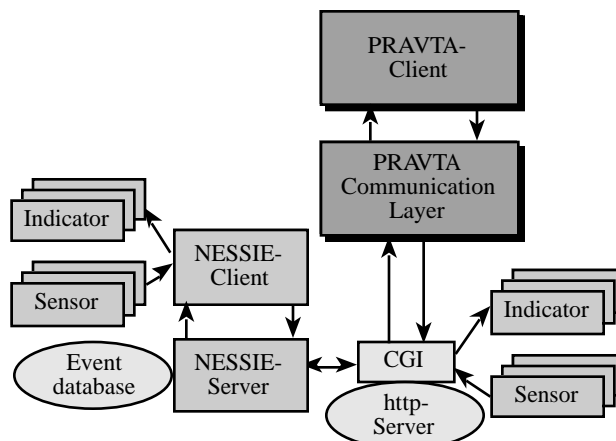


Figure 1. The PRAVTA software architecture.

The user interface at the *PRAVTA Client* is implemented in the wireless markup language (WML) and in the wireless markup script language (WMLScript). The wireless application protocol connects the user interface to the PRAVTA communication layer. The *PRAVTA Communication Layer* translates the data from the NESSIE server into PRAVTA format (i.e., WML and WMLScript) and translates the data from the PRAVTA client into NESSIE format. It provides mechanisms for login and access control, and so forth. It is implemented in Tcl/Tk. The PRAVTA Communication Layer uses of the common gateway interface provided by NESSIE in order to communicate with the remaining parts of NESSIE.

3.3 Functionality

The first page of the PRAVTA prototype shows the front page and a menu from which users can choose various actions. Figure 2 shows screenshots of the front page, the action menu list, and the login dialogue.



Figure 2. The PRAVTA prototype user interface: (a) front page; (b) action menu list; (c) login dialogue.

The user needs to be logged in before she can query for information about other users. This is important to maintain privacy of users. We will discuss the issue of privacy and related issues later on. The user then can check who is online. Figure 3 shows screenshots of the result of the action menu items ‘Who is online?’, ‘Check availability’, and ‘Checktasks’, as well as the actual state in the 3D multi-user environment.

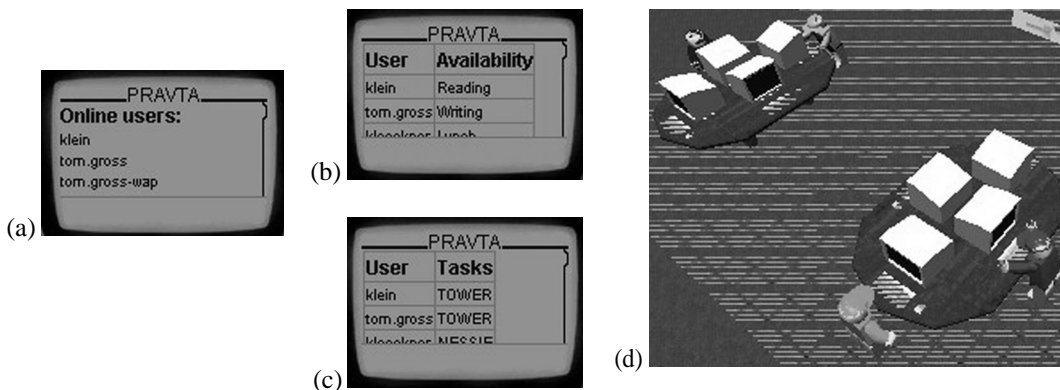


Figure 3. The PRAVTA prototype user interface: (a) result of ‘Who is online?’; (b) result of ‘Check availability’; (c) result of ‘Check tasks’; (d) corresponding 3D multi-user environment.

A simple list resulting from the query *Who is online* shows that currently the users ‘klein’, ‘tom.gross’, and ‘tom.gross-wap’, and so forth are logged in. In fact, ‘tom.gross-wap’ is the user who in our example performed the query using a mobile phone. That is, the user can see herself in the list of online users. The fact that anybody can be seen by anybody else is called reciprocity of awareness. The list shows all users who are logged in on the NESSIE Server no matter what system they are using. Currently, mobile users can log in from the PRAVTA prototype, and stationary users can log in from a Java-based NESSIE Client, and from a 3D multi-user environment [2]. The result of *Check availability* shows a table with the current availability of the users. Currently the PRAVTA prototype offers availability states such as Reading, Writing, Meeting, and so forth. The result of *Check tasks* shows a table with the current tasks (i.e., task

contexts) of the users. The table shows that when the screenshot was made user ‘klein’ and user ‘tom.gross-wap’ were working in the context of the TOWER project, and so forth.

In the 3D multi-user environment presence awareness is provided by the presence of user avatars; availability awareness is provided by a visualisation of the current activity of the respective user; and task awareness is symbolised by the room the avatar is in and the table the avatar is using.

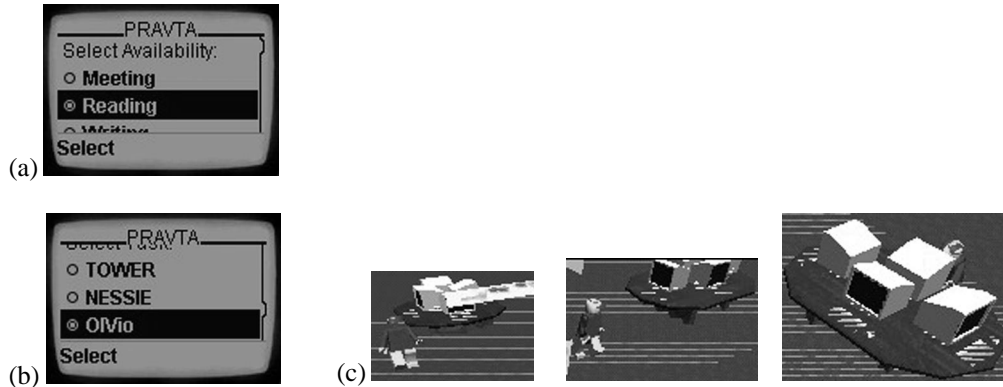


Figure 4. The PRAVTA prototype user interface: (a) update of availability; (b) update of task; (c) corresponding update of the user task (i.e., location) in the 3D multi-user environment (showing the user tom.gross-wap: leaving the TOWER task context, walking, working in the new OIVio task context).

Users can not only query information, but also change their own status. Figure 4 shows screenshots of the dialogue for changing the availability information and the task information as well as the corresponding update in the 3D multi-user environment. On the top of the dialogue the user can choose if she is currently in a Meeting, Reading, Writing, having Lunch, and so forth (cf. Figure 4a). On the bottom of the dialogue the user can enter information about her current task context (cf. Figure 4b). The status of the user is then immediately updated in the NESSIE server and the various clients also receive the updated information. Figure 4c shows the update in the 3D world as an example—the avatar representing user ‘tom.gross-wap’ leaves the TOWER project space and moves towards the OIVio project space, where it starts reading. It can also be seen that this user is currently the only person working in the context of the OIVio project.

Finally, Figure 5 shows screenshots of the broadcasting functionality of the PRAVTA prototype: a screenshot with a dialogue for entering a broadcast message, the feedback of the executed command, and the resulting tickertape on the PCs of the colleagues in the offices.

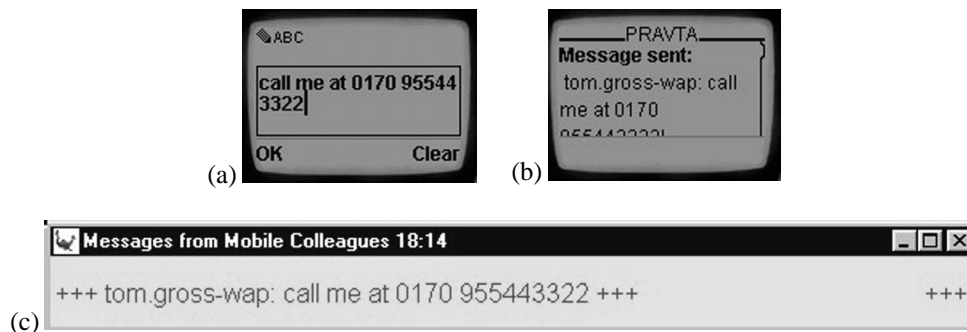


Figure 5. The PRAVTA prototype user interface: (a) send message dialogue; (b) feedback; (c) resulting tickertape on the PCs of the colleagues in the offices.

The tickertape runs on the monitors of the PCs of all users who currently run any kind of NESSIE client. So far, the tickertape is only used to show the concept of broadcasting in PRAVTA. Of course, the broadcasting functionality could be extended. For instance, in the 3D world the broadcast message could be visualised as a bubble attached to the sender like in cartoons. For other mobile users who use PRAVTA on a mobile phone it could be translated into a short

message (SMS). For users who should be reached, but who are currently not logged into the system at all, the broadcast message could be translated into an email message.

4 Related Work

Only few systems and prototypes provide awareness for mobile and nomadic users. Examples are the Audio Aura and Nomadic Radio. The Audio Aura [8] provides mobile digital audio via portable, wireless headphones. The audio information that is provided depends on the physical actions and position of the user, which are traced by active badges. Users can choose sound designs like voice only, music only, sound effects only, or a mixture. The overall goal of Audio Aura is to provide serendipitous information (i.e., information that is relevant, but not essential) by use of background auditory cues.

The Nomadic Radio [10] provides mobile digital audio via the SoundBeam Neckset. The particular strengths of the system are a contextual notification model and scalable auditory techniques for providing timely information and minimising interruptions. Whereas in Audio Aura only the position of the user in the physical environment is taken into account, in Nomadic Radio several sensors measure the environment of the user and try to infer her current context. For instance, an audio sensor captures whether the user is currently in a conversation. The auditory techniques can be scaled from silence for least interruption and conversation, to ambient cues for peripheral awareness, to auditory cues for notification about task completion, and so forth. On a whole the system offers sophisticated mechanisms for the subtle presentation of information. However, it primarily provides an interface for communication systems such as email and voice mail. It does not provide users with information about other users and it does not allow users to enter information about themselves.

Event notification services provide similar information as the PRAVTA prototype. For instance, Elvin [4] and Khronika [7] are systems that use sensors and daemons to capture information from various sources in the form of events and that allow users subscribe to events or to browse events. Once, the user has subscribed she is provided with related information automatically. These systems are similar to the core NESSIE system, but they do not offer the advantages of PRAVTA.

6 Conclusions

The PRAVTA prototype provides ubiquitous awareness about presence, availability, and tasks of other users. It allows users to query various types of information anytime and anywhere. Furthermore, users can flexibly update information about their presence, availability, and tasks.

Some features that have not yet been implemented for the PRAVTA prototype are a push-principle and the automatic capturing of data about the mobile users (especially their current location). So far, the WAP technology only supports queries by users. Currently, there are many discussions about the advantages of support for push services with WAP. With push services the user would not actively have to query the awareness information needed, but rather the system could send the information to the user immediately after it receives it. Automatic capturing of information about mobile users is also being discussed currently. For instance, the application of GPS systems in combination with WAP devices would allow to capture the current position of the users. This information could then be used for inferring information about the user's presence, availability, and task context, and for adapting the contents of the awareness information and its presentation to the current context of the user.

Acknowledgements

The research presented here was carried out and financed by the IST-10846 project TOWER, partly funded by the EC. I would like to thank all my colleagues from the TOWER team.

References

1. Borning, A. and Travers, M. Two Approaches to Casual Interaction Over Computer and Video Networks. In *Proceedings of the Conference on Human Factors in Computing Systems - CHI'91* (Apr. 27-May 2, New Orleans, LO). ACM, N.Y., 1991. pp. 13-20.
2. Broll, W. and Prinz, W. Using 3D to Support Awareness in Virtual Teams on the Web. In *World Conference on the WWW and Internet - AACE WebNET'99* (Oct. 25-30, Honolulu, Hawaii). 1999. pp. to appear.
3. Dourish, P. and Belotti, V. Awareness and Coordination in Shared Workspaces. In *Proceedings of the Conference on Computer-Supported Cooperative Work - CSCW'92* (Oct. 31-Nov. 4, Toronto, Canada). ACM, N.Y., 1992. pp. 107-114.
4. Fitzpatrick, G., Mansfield, T., Kaplan, S., Arnold, D., Phelps, T. and Segall, B. Augmenting the Workaday World with Elvin. In *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work - ECSCW'99* (Sept. 12-16, Copenhagen, Denmark). Kluwer Academic Publishers, Dordrecht, NL, 1999. pp. 431-450.
5. Gutwin, C. and Greenberg, S. Support for Group Awareness in Real-Time Desktop Conferences. In *Proceedings of the Second New Zealand Computer Science Research Students' Conference* (Apr. 18-21, Waikato, Hamilton, NZ). 1995.
6. Gutwin, C., Greenberg, S. and Roseman, M. Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation. In *Proceedings of the Conference on Human-Computer Interaction: People and Computers - HCI'96* (Aug. 20-23, London, UK). Springer-Verlag, Heidelberg, 1996. pp. 281-298.
7. Loevstrand, L. Being Selectively Aware with the Khronika System. In *Proceedings of the Second European Conference on Computer-Supported Cooperative Work - ECSCW'91* (Sept. 24-27, Amsterdam, NL). Kluwer Academic Publishers, Dordrecht, NL, 1991. pp. 265-278.
8. Mynatt, E.D., Back, M. and Want, R. Designing Audio Aura. In *Proceedings of the Conference on Human Factors in Computing Systems - CHI'98* (Apr. 18-23, Los Angeles, CA). ACM, N.Y., 1998. pp. 566-573.
9. Prinz, W. NESSIE: An Awareness Environment for Cooperative Settings. In *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work - ECSCW'99* (Sept. 12-16, Copenhagen, Denmark). Kluwer Academic Publishers, Dordrecht, NL, 1999. pp. 391-410.
10. Sawhney, N. and Schmandt, C. Nomadic Radio: Scalable and Contextual Notification for Wearable Audio Messaging. In *Proceedings of the Conference on Human Factors in Computing Systems - CHI'99* (May 15-20, Philadelphia, PE). ACM, 1999. pp. 96-103.

WAPcam - eine multimediale WAP Anwendung in der Studentenausbildung

Frank Kargl (frank.kargl@informatik.uni-ulm.de)
Torsten Illmann (torsten.illmann@informatik.uni-ulm.de)
Alexander Raschke (alexander.raschke@informatik.uni-ulm.de)
Stefan Schlott (stefan.schlott@informatik.uni-ulm.de)

Universität Ulm
Abteilung Medieninformatik

17.01.2001

1 Einleitung

Im Rahmen eines geplanten Praktikums mit dem Titel “Web-Engineering” entstand im Frühjahr 2000 an unserem Lehrstuhl die Idee, auch das Themengebiet WAP aufzugreifen. Das Praktikum selbst stellt ein Ergänzungsangebot zu einer gleichnamigen Vorlesung dar und vertieft in Themen wie Dokumenten- und Layoutsprachen im Web, Server- und Klienten-seitige Applikationsentwicklung fürs Web, Interaktionsformen und vieles mehr.

Von unseren Kollegen am TECO in Karlsruhe, mit denen sowohl die Vorlesung als auch das Praktikum in Kooperation entwickelt konzipiert und weiterentwickelt wird, übernahmen wir die Idee, auch WAP und insbesondere dem Thema “Messen/Steuern/Regeln mit Web und WAP” einen eigenen Platz einzuräumen. Während im vorausgegangenen Semester in Karlsruhe das Praktikum sich mit einfachen Schalt- und Regelvorgängen wie dem Ein- und Ausschalten einzelner Dioden beschäftigte, wollten wir allerdings einen Schritt weitergehen. Unsere Idee war es, eine interaktiv steuerbare Kamera über WAP zugänglich zu machen: die *WAPcam*.

2 Anforderungen

Bevor wir uns an eine Umsetzung heran wagten, wollten wir zunächst klären, unter welchen Rahmenbedingungen ein solches Unternehmen realisiert werden

konnte.

Aus technischer Sicht war zunächst zu definieren, welche Funktionalität die Anwendung haben sollte. Die Kamera sollte dreh- und schwenkbar gelagert sein und sich über das Handy intuitiv bedienen lassen. Wegen des kleinen Displays sollte weiterhin eine Zoom-Funktion integriert sein, um Details besser erkennen zu können. Da die Displays bei den gängigen Mobiltelefonen keine Graustufen- oder gar Farbdarstellung unterstützen, war weiterhin eine Kontrast- bzw. Helligkeitsregulierung vorgesehen.

Neben den technischen Aspekten, die unmittelbar für die Implementierung eines Prototypen interessant waren, mussten wir selbstverständlich auch die organisatorischen Aspekte im Praktikum berücksichtigen. Schon aus Kostengründen war klar, dass die Studenten primär mittels eines WAP Simulators [1] entwickeln und testen würden. Das Praktikum beinhaltete bereits eine Vielzahl unterschiedlichster Technologien (HTML, CGI, Perl, PHP, Java, Servlets, Applets, XML, XSL, ...), die den Studenten zumindest in der praktische Anwendung teilweise völlig neu waren. Von daher war der Arbeitsaufwand nicht zu unterschätzen und wir hatten zunächst Bedenken, einen weiteren, komplett neuen Technologiebereich (WAP, WML [2], WMLS [3]) mit einzubinden. Wenn überhaupt, dann musste diese Aufgabe, die am Semesterende auch den letzten Aufgabenbereich bilden sollte, mit Sicherheit entsprechend spannend und ansprechend gestaltet werden, um die Motivation deutlich über die Aussicht auf einen Praktikumsschein zu heben. Wir wollten also in jedem Fall neben der Simulationsumgebung auch die Arbeit an richtigen Mobiltelefonen ermöglichen. Auf unsere Anfrage hin stellte uns das Entwicklungszentrum der Firma Siemens in Ulm ausreichend wap-fähige Geräte des Typs Siemens S35i zur Verfügung. Die zentrale Universitätsverwaltung versorgte uns mit notwendigen SIM-Karten, ohne die ein Mobiltelefon nur eine nutzlose Hülle bleibt. Um die Motivation weiter zu steigern, bauten wir um die Kamera herum eine Art Marslandschaft auf, die ein verstecktes Rätsel enthielt, welches es zu entdecken und zu enträtseln galt.

3 Systemarchitektur

Der grobe Aufbau der WAPcam Steuerung wird in Abbildung 1 dargestellt. Die Kamera ist dabei auf einen dreh- und schwenkbaren Fuß [4] montiert. Das Auslesen der Kamera sowie die Steuerung des Kamerafusses erfolgt über einen separaten Kontrollrechner. Dieser stellt die Funktionalität über einen Java RMI Dienst [5], welcher einfach von anderen Rechnern angesprochen werden kann, zur Verfügung. Die RMI Schnittstelle beinhaltet Methoden, um den Kameraarm entweder absolut auf bestimmte Winkel zu positionieren oder aber relativ um einen Winkel zu drehen und zu schwenken. Darüber hinaus läßt sich der Kamerafuß in die Ausgangsstellung zurücksetzen und es kann ein einzelnes Kamerabild ausgelesen werden. Die RMI Schnittstelle wurde von uns entwickelt und den Studenten zusammen mit den zugehörigen Stub-Klassen und Testdateien zur Verfügung gestellt.

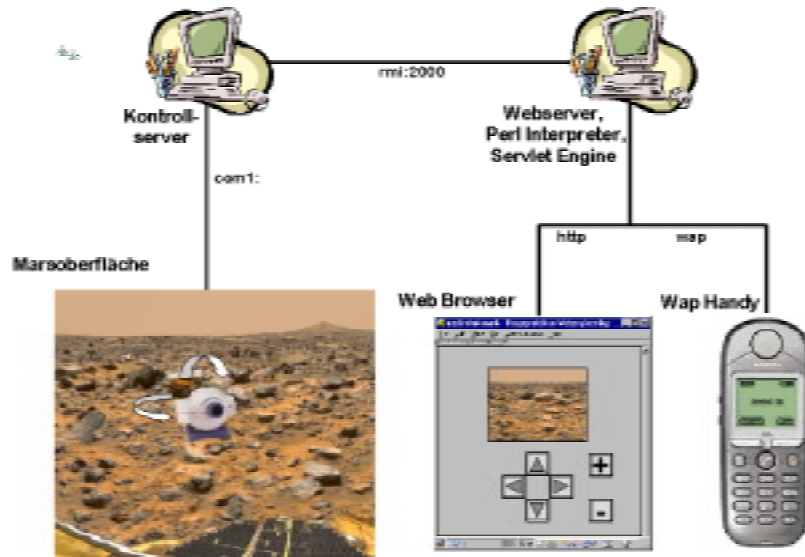


Abbildung 1: Aufbau der WAPcam Steuerung

Um verwirrendes Verhalten der Kamerasteuerung bei konkurrierendem Zugriff aus mehreren Anwendungen zu vermeiden, ist eine explizite An- und Abmeldung notwendig. Bei der Anmeldung erhält die Anwendung ein Token, welches nach der Abmeldung oder nach einem Timeout seine Gültigkeit verliert. Auf einem weiteren Rechner ist ein Webserver installiert, der den Zugriff via HTTP ermöglicht. Bei WAP ist es wegen der eingeschränkten Endgeräte und den beschränkten Bandbreite sinnvoll, soviel Verarbeitungsschritte wie möglich lokal im Server durchzuführen und das Endgerät nur zum Anzeigen der Daten und für einfachste Operationen in Anspruch zu nehmen. Die Anwendungslogik ist daher weitestgehend im Webserver in Form eines CGI-Skripts [6] oder Servlets [7] realisiert. Folgende Funktionen sind umzusetzen:

- Bewegungssteuerung der Kamera gemäß den Steuerbefehlen des Endgerätes und
- Abrufen der Bildinformation und (*sinnvolle*) Umwandlung in ein WBMP Bild [8].

Insbesondere die Bildwandlung verdient eine besondere Berücksichtigung. Das von der Kamera (über den RMI Dienst) gelieferte Ausgangsbild hat eine Abmessung von 640x480 Bildpunkten und ist JPEG kodiert. Beim Empfänger

kann lediglich ein schwarz-weiss Bild mit etwa 100x70 Bildpunkten dargestellt werden, wobei diese Größe noch starken Schwankungen unterliegt. Im Vorfeld äußerten mehrere Teilnehmer ernsthafte Bedenken, ob unter solchen Voraussetzungen überhaupt ein sinnvoller Einsatz dieser Technologie möglich ist. Dazu werden wir im letzten Kapitel Überlegungen anstellen. Es ist jedoch offensichtlich, dass bei der Konvertierung die Bildinformation aufbereitet werden muss. In unserer aktuellen Implementierung verwenden wir folgende Bildbearbeitungsfunktionen, um eine sinnvolle Umwandlung in ein WBMP Bildformat herzustellen::

1. Umwandlung in ein Graustufenbild,
2. Kanten- und Kontrastverstärkung,
3. Ausschnitt (dies realisiert die gewünschte Zoom-Funktion),
4. Skalierung auf das gewünschte Zielformat und
5. Umwandlung in Schwarz/Weiss mit Floyd-Steinberg Dithering (mit einstellbarem Threshold: dies realisiert die gewünschte Helligkeitseinstellung).

Dabei ist es nicht ganz einfach, das Zielformat zu ermitteln. Manche WAP Browser übertragen zwar die maximale Displaygröße oder den Typ des Mobiltelefons, leider ist dieser Mechanismus nicht standardisiert, so dass man oft auf den kleinsten gemeinsamen Nenner zurückgreifen muss und daher die Möglichkeiten des Endgeräts unter Umständen nur ungenügend ausnutzt. Hier sollten im Standard entsprechende Erweiterungen vorgenommen werden.

Die Funktion des WAP Gateways haben wir nicht selbst realisiert. Wir verwenden hier vielmehr die WAP Gateways der jeweiligen Netzbetreiber. Mit den uns zur Verfügung stehenden Karten war der Zugriff auf die Gateways von D1 und D2 möglich. Hiermit traten während der Tests keine Probleme auf, von einer häufigen Nicht-Erreichbarkeit einmal abgesehen.

4 Prototypische Implementierung

Im Vorfeld des Praktikums wurde das System prototypisch implementiert, um die Machbarkeit zu demonstrieren. Das Bild wird in eine statische WML Seite eingebettet und von einem Perl CGI-Skript dargestellt. Steuerungskommandos, Zoom-Faktor und Helligkeitsswerte erhält das CGI-Skript über Parameter mitgeteilt. Da wir zur Bedienung des Systems eine (relativ) grosse Anzahl von Tasten benötigen und uns daher die wenigen Softkeys nicht ausreichen, haben wir uns entschlossen, die Zifferntasten des Mobiltelefons zu belegen. Dabei entsprechen die Tasten 2, 4, 6 und 8 den Richtungen hoch, links, rechts und runter. Über die Tasten 1 und 3 ist die Zoom-Funktion zu steuern und 7 und 9 regeln die Helligkeit. Die Taste 5 setzt die Steuerung auf Ausgangsstellung zurück. Diese Lösung ist in der Praxis sehr gut bedienbar, hat aber leider zwei Nachteile:

- Die Zifferntasten lassen sich nur mit einer proprietären Erweiterung des von den Siemens-Mobiltelefonen verwendeten Browsers von Phone.com [1] belegen (ACCESSKEY Attribut von <ANCHOR>). Damit sind die Seiten nicht mehr allgemein übertragbar.
- Die ANCHOR Tags erscheinen als Textzeilen unter dem Bild. Nach Auslösen einer Funktion muss man u.U. manuell nach oben scrollen, um das Bild vollständig zu sehen.

Unserer Meinung nach bietet der heute implementierte WML/WMLS Standard noch nicht genügend Möglichkeiten für umfangreiche interaktive Applikationen. Daher ist zu befürchten, dass Entwickler verstärkt auf proprietäre Erweiterungen der Hersteller ausweichen werden. Hier sollte der Standard schnell so erweitert werden, dass zumindest alle sinnvollen Ein- und Ausgabemechanismen eines Mobiltelefons (und dazu gehören sicher die Zifferntasten) erfasst werden können.

Die Implementierung war relativ problemlos und lief im Simulator von Phone.com bereits recht früh. Allerdings zeigte sich bei den Mobiltelefonen ein unangenehmer Fehler. WML Variablen (die über <setvar> gesetzt bzw. in WMLS modifiziert wurden), werden bei den uns zur Verfügung stehenden Mobiltelefonen innerhalb von Attributwerten immer in einen leeren String evaluiert. Da somit keine dynamische Parameterübergabe an das Bildskript möglich war (kodiert in der URL des aufgerufenen CGI-Skriptes), mussten wir hier auf eine Ersatzlösung ausweichen. Dabei wird die WML Seite, die das Bild einbindet, dynamisch über ein weiteres Perl CGI-Skript erzeugt und die entsprechenden Parameter hart in die URLs kodiert.

Die Sourcen unserer Implementierung werden nach Ablauf des Praktikums Mitte Februar auf unserem Webserver (<http://webeng.informatik.uni-ulm.de>) abrufbar sein.

5 Erfahrungen und Anwendungsgebiete

Zunächst einige technische Aspekte. Obwohl die Idee der sinnvollen Realisierung einer interaktiven Kamera mit WAP bei vielen Leuten (auch Mobilfunkentwicklern!) auf grosse Skepsis stiess und bezweifelt wurde, liessen wir uns nicht von dem Versuch abbringen und wollten im schlimmsten Fall zumindest diese Skepsis mit Tests bestätigen. Die Ergebnisse des fertigen Prototyps stossen hingegen in der Regel auf eine positive Resonanz. Alles steht und fällt mit der Bildqualität. Mit den heute vorhandenen Endgeräten lassen sich je nach Motiv immerhin befriedigende bis gute Ergebnisse erzielen. Die zukünftige Entwicklung hin zu Geräten mit höherer Auflösung und Farbfähigkeit sollte die heutigen Qualitätsprobleme kurz- bis mittelfristig beseitigen.

Doch auch heute schon sind sinnvolle Anwendungen möglich. So liesse sich mit Hilfe von WAP ein privates Überwachungssystem realisieren, mit dem Hausbesitzer erkennen können, ob eingebrochen wurde oder Pflanzen gegossen werden müssen. Zusammen mit weiteren Steuerungsfunktionen (z.B. Jalousinen,

automatische Bewässerungsanlage usw.) könnte ein wirklicher Mehrwert geschaffen werden. Neben den eingeschränkten Endgeräten sind natürlich auch die geringe Übertragungsbandbreite und vor allem das starre und teure Preissystem der Mobilfunkunternehmen ein Hindernis bei der Durchsetzung solcher Anwendungen.

Unsere Studenten, die momentan die gestellte Aufgabe bearbeiten, nahmen die Aufgabenstellung jedenfalls sehr interessiert an. Die zahlreichen Kinderkrankheiten und Inkompatibilitäten, mit denen WAP heute noch zu kämpfen hat, sind sicher gerade in der Ausbildung problematisch, stören jedoch bei entsprechender Motivation der Teilnehmer nur geringfügig.

Literatur

- [1] Phone.com: UP SDK Development Kit, Release 4.1, 2000.
URL: <http://developer.phone.com/>
- [2] WAP Forum: Wireless Markup Language Specification, April 30, 1998.
URL: <http://www.wapforum.org/>
- [3] WAP Forum: WMLScript Specification, April 30, 1998.
URL: <http://www.wapforum.org/>
- [4] Directed Perceptions Inc: Pan-Tilt Tracking Mount - C Programmer's Interface, Version 1.0707b, 1995.
- [5] Sun Microsystems Inc: RMI Specification.
URL: <http://java.sun.com/j2se/1.3/docs/guide/rmi/spec/rmiTOC.html>
- [6] E. Wilde: Wilde's WWW. Technical Foundations of the World Wide Web, Springer Verlag, 1999.
- [7] Sun Microsystems Inc: Java Servlets Specification, 2001.
URL: <http://java.sun.com/products/servlet/2.2/>
- [8] WAP Forum: Wireless Application Protocol Wireless Application Environment Specification Version 1.1. URL: <http://www.wapforum.org/>

Navigating Within Tables Used in SAP BW Mobile Reporting (WAP): Results of Usability Tests

Andreas Kramer*, Markus Latzina^o, and Jörg Mann*

* SAP AG, Corporate Research / CEC, Karlsruhe

^o SAP AG, Usability Engineering Center, Walldorf

Abstract

We herein present three designs for navigating within tables used in SAP BW Mobile Reporting (WAP). The designs were tested by 15 ASUG conference participants. The testing situation closely resembled the mobile context of use for which the scenarios were developed. In addition to the scenarios and test results, we also describe the design ideas generated from the problems the participants encountered.

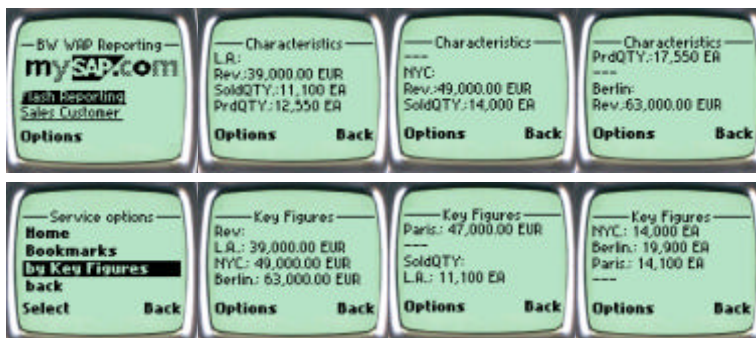
1 Usability Tests: Situation and Setup

Scenarios and participants: The user is a sales executive on a business trip who needs key sales figures to prepare for a meeting. The 15 participants of our usability tests were attendees of an Americas SAP Users' Group (ASUG) conference in October 2000; therefore, for them, the testing situation resembled a mobile context of use.

Testing materials and device: We used reports from SAP Business Information Warehouse (BW) Mobile Reporting. Since no U.S. phones (such as a PCS Sprint Phone) were available, we set up the application as an offline desktop scenario on a Nokia 7110 WAP phone emulator.

2 Test Results, User Problems, and Design Ideas

2.1 Usability Test Results for Scenario 1: Sales Figures Lookup (Table with 2 Columns, 4 Rows)



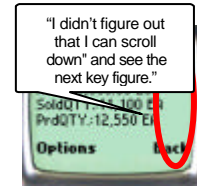
Note: Read screen shots from left to right.

2.1.1 Navigation Was Problematic

Users understood that they could switch between key figures and characteristics. However, there was no indication that you can scroll down in order to see more data/the next key figure. Users argued that this only works with a limited number of locations; a large number of locations would be hard to read.

Design Idea (DI): Provide a vertical scrollbar or an arrow that shows more.

DI: Display a preview of key figure titles at the top; “show the user what to expect,” revenue, then quantity produced, etc.

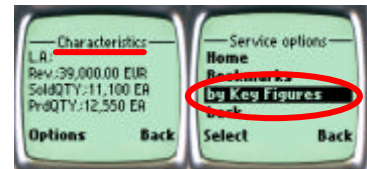


2.1.2 Options Were Unclear

Problem with *Key Figures* and *Characteristics*: Users correctly assumed that the item under *Options* was a display option. However, they didn't understand the option items, and were sometimes unsure of how to switch between them.

DI: Under *Options*, show both items at the same time: *By Key Figures*, *By Characteristics*.

DI: Use more precise terminology.

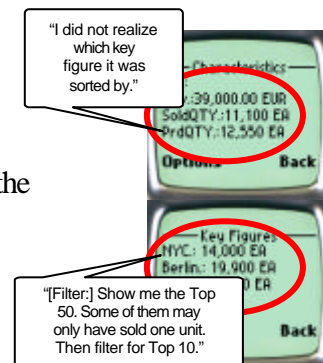


2.1.3 No Sort or Filter Provided

Problems occurred with sorting by key figures vs. characteristics: Users looked for a sort function, and did not grasp which key figure the report was sorted by.

DI: Provide the option of sorting by key figures, ascending/descending.

DI: Offer filtering capabilities (e.g., Top 10, Bottom 5).

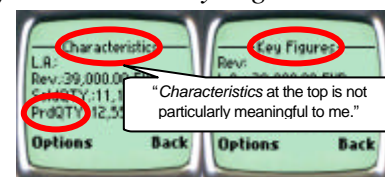


2.1.4 Terminology and Abbreviations Were Unclear

Some terms and abbreviations were not clear: PrdQTY (Produced Quantity) is not intuitive and could be misinterpreted, e.g., as “Predicted Quantity.” The terms *Key Figures* and *Characteristics* at the top were not meaningful.

DI: Use *By Key Figures*, *By Characteristics* at the top.

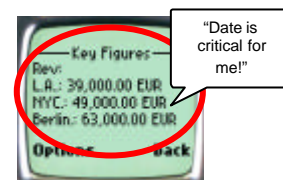
DI: Use the abbreviation *QtyProd* (“Quantity Produced”; “Produced Quantity” is a Germanism) instead of *PrdQTY*.



2.1.5 No Date Information Provided

There was no date information indicating how up to date the data was. However, users regarded the date as critical.

DI: Display date in header line.



2.1.6 No Currency Conversion Available

Displaying all values in a single currency was well received; however, an option for

converting currencies (e.g., EUR⇔USD) should be provided.

DI: Provide option for currency conversion.

2.2 Usability Test Results for Scenario 2: Billed Items Lookup (Table with 5 Columns, > 20 Rows)



Note: Read screen shots from left to right.

2.2.1 Navigation Was Problematic (1)

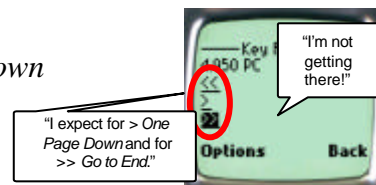
Users encountered *severe* navigation problems with scenario 2. They argued that people would need to “know the landscape.” A graphic showing the table layout would be nice.

Symbols were confusing. “>>” and “<<” imply going to the end/beginning; users would have expected right/left arrows for changing the column.

Rather than “>” and “<,” users would have expected *Arrow Down* and *Arrow Up* for displaying the next couple of data rows.

DI: Display gridlines (key line, values).

DI: Use down/up arrows instead of “>” and “<,” or use text (*pgdown/pgup*) instead.



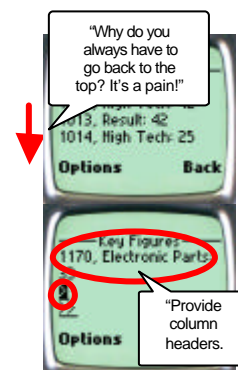
2.2.2 Navigation Was Problematic (2)

Too much scrolling involved: Users heavily criticized that they had to go all the way to the bottom to be able to switch one column to the right. If they wanted to switch two or more columns, this procedure had to be repeated zigzag-style and was therefore particularly cumbersome. Also, column headers were no longer displayed after the user had scrolled down.

DI: Have “>>” at the top and bottom, and/or under *Options*.

DI: To avoid zigzag navigation, enable consecutive scrolling to the right, e.g., by placing “>>” under *Options*.

DI: Provide column headers at all times.



2.2.3 Options Were Unclear

Users tried to search for options. However, no options were available.

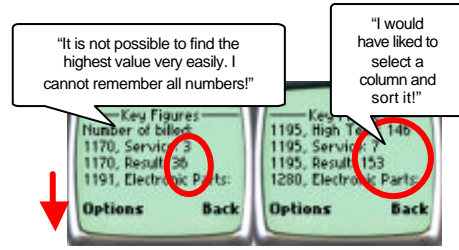
DI: Place functions/links on the screen under *Options* as well.

DI: Provide the option of setting a default column.

2.2.4 No Sorting Provided

Testers assumed they would be able to sort by key figures, as people usually look for highest and lowest values.

DI: Provide a sort function for key figures (ascending or descending).

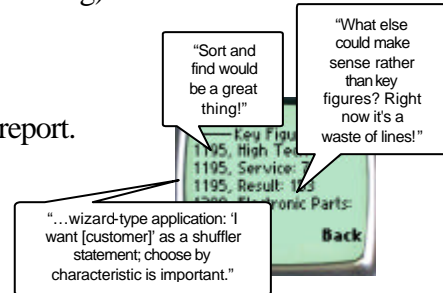


2.2.5 No Search or Filter Available

Users complained that they could not search or filter in this report.

DI: Provide a filter - e.g., Top 10, Bottom 5, by region, or by product line - and use localization information.

DI: Offer shuffler search (find XYZ) as for scenario 3.

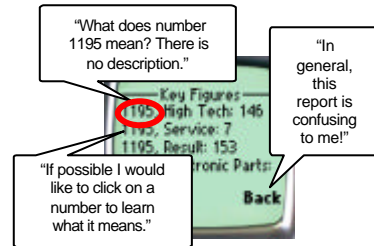


2.2.6 Terminology and Abbreviations Were Unclear

Because there were no column headers, users did not understand what the columns meant.

Division was self-explanatory. *Payer*, which was displayed as a number, was very hard to understand.

DI: Offer an option of viewing numbers or description. If this is not technically feasible, display column descriptions/table layout in a *Help* function under *Options*.

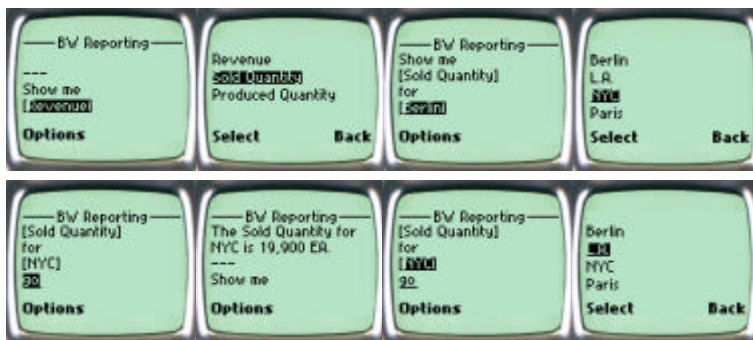


2.2.7 No Date Information Provided

There is no date information indicating how up to date the data is. However, users regarded the date as critical.

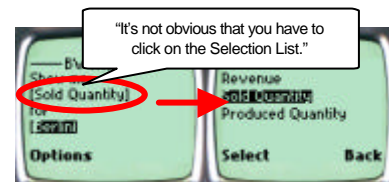
DI: Display the date in the header line.

2.3 Usability Test Results for Scenario 3: Sales Figures Lookup ("Shuffler"-Style Search)



Note: Read screen shots from left to right.

2.3.1 Selection List Was Unwieldy (1)



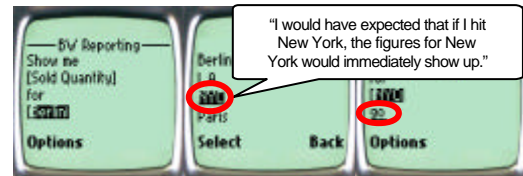
Users were confused about how to use the *selection list*. They did not understand that the brackets indicate that the entry can be changed.

In general, the users liked the selection list. However, they argued that it would not work if there were too many attributes to select from; for this reason, they suggested keeping the number of codes to a minimum.

DI: Use an asterisk (*); users know that they will get a dropdown list box if they hit “*”.

2.3.2 Selection List Was Unwieldy (2)

One user was also assuming that the search would automatically be started after he had selected an entry from the last selection list.

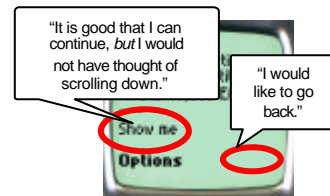


DI: Place Go under Options as well, so there is no need to scroll down.

2.3.3 Navigation Was Problematic

The shuffler search was well received. There were only a few small navigation problems: Users did not expect to be able to scroll down in order to start the next search. Rather, after executing the search, they expected a *Back* function.

DI: In addition to the repeated search option, provide a *Back* function after the user has executed a search.



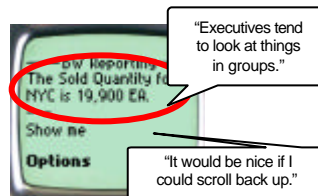
2.3.4 No Comparison, Sorting, or Filtering Provided

Users wanted to be able to compare figures, and sort and filter them.

DI: Use a *multiselection list* to allow the user to select one or more *Key Figures* and one or more *Locations* for comparison (result: e.g., *Sold Quantity* figures for both Berlin and New York on a single screen).

DI (paper prototype sketched by user):

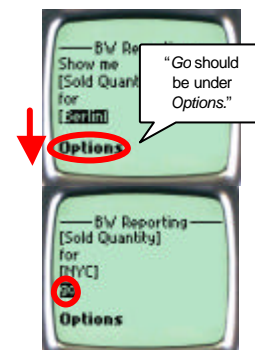
- [Selection list for char.: - Customer Channel
- Product Group
- Material Type
- Top 10]



2.3.5 Options Were Unclear

Users expected to have *Go* as an entry under *Options*. However, they had to scroll down first to start the search.

DI: Place functions/links on the screen under *Options* as well.



3. Summary

Scenario 1: Users liked the navigation provided by the options *By Characteristics* or *By Key Figures* (report with 2 columns, 4 lines).

Scenario 2: Severe problems occurred when users tried to navigate in a large table. As a result, they requested filtering (e.g., Top 10, Bottom 5) and sorting capabilities (e.g.,

descending / ascending by key figure).

Scenario 3: The shuffler search was well received, which resolves the navigation problem.

Towards a unifying Approach to Mobile Computing

Carsten Magerkurth and Thorsten Prante
GMD – German National Research Center for Information Technology
IPSI – Integrated Publication and Information Systems Institute
Dolivostr. 15, D-64293 Darmstadt
{magerkur, prante}@darmstadt.gmd.de

Points of Departure

Today, the domain of *mobile computing* is as diverging as the domain of desktop computing was some twenty years ago. Just like in the early days of *home computing*, when computer manufacturers were enforcing incompatibility even within their own range of devices¹, there are numerous excluding alternatives emerging for the mobile user.

There is no widely accepted standard of what mobile computing should be like. Different *device classes* (such as palm-sized PC's, pocket-sized PC's, or WAP enabled mobile phones) with different *operating systems* (such as PalmOS, EPOC, or Windows CE) support *different paradigms* regarding the purpose and applicability of mobile computing (Greenberg et al., 1999, Myers et al., 1998).

From a developer's point of view, it is crucial, whether a mobile device is to be used as a mere data viewer or a remote control (as e.g. Rekimoto's "painter's palette" (1998)) or, on the other hand, as an autonomous system that can operate independently to some degree (Magerkurth & Prante, 2001). The former approach moves most of the application's functionality away from the mobile device requiring a tight connection to "fixed" computer systems, as it is the case with the mobile phone's WAP browser to the corresponding ISP. The latter approach enables more independent work, but puts higher demands on processing power and memory resources. Therefore, truly autonomous work may not be possible with all kinds of mobile devices due to resource constraints encountered, e.g., on mobile phones.

While this issue cannot be resolved analogous to the one of whether to use thin clients in client-server networks, there is a more immediate limitation to the role of mobile computing: Currently, there are numerous mobile technologies on the rise and most of them come along with very proprietary API's and little facilities to integrate with other technologies. This has led to the now highly cluttered market, where devices of different types are hardly able to communicate with each other.

Most of the proprietary platforms do not make up a market share worth investing to a duly degree. Even though, there are sufficient similarities amongst most kinds of mobile devices, applications are mostly available just for a single platform. In addition, the possibilities of cross-platform data exchange are still moderate, even if similar communications hardware and protocols (e.g. IrDA) are used. Even the largely device independent WAP technology has not yet gained enough attention to raise a sufficient software infrastructure. Many WAP internet offers are buggy or lack features (Bager & Bleich, 2000).

¹ The shift from Palm III to Palm V introduced essentially the same unintelligible serial port changes as from CBM64 to CBM+4 in the early eighties.

In our opinion, device independent application technologies like Java and to some extent WAP are not the only answer to these well known problems of the mobile domain. Relying on common features and standards inevitably both implies losing the flexibility to cater to specific features of the various devices as well as introducing a performance overhead (provided that the underlying hardware is indeed heterogeneous and the software performance is a crucial criterion).

Approach

Our approach is to make use of the similarities in mobile systems' user interfaces while refraining from any unworthy bytecode overhead that is especially fatal in the context of mobile devices with their limited resources. Regarding user interaction, most mobile devices have a lot in common: Text entry/ recognition mechanisms are too awkward to be fun, displays are too small to hold much information, and CPU's are too slow to allow for strategies to overcome the display shortcomings by, e.g., fast and accurate zooming or panning.

We opt for supporting various devices by introducing scaled implementations of user interface components that are applicable to more or less all mobile systems. For instance, a text button on a mobile phone might be made up of a simple sequence of characters surrounded by a black frame. To press it, users have to make use of the hardware buttons on the device, due to the lack of a touch-sensitive display. On an EPOC handheld PC, however, the button can be a more sophisticated control which can, e.g., easily be dragged about the display with a stylus. Nevertheless, taking the application's back-end view, a button is always just a button, i.e., a medium for communicating the user's request to have a corresponding action performed. Likewise, the application itself does not need to bother, if the user is picking his theatre tickets from a plain combobox or a highly polished, rotating, zoomable list-control.

In our opinion, it suffices to abstractly define only a very limited set of user interface controls applicable for mobile devices and implement them differently with regard to the specifications of the respective mobile platform. We found that we actually never needed more than a button, a list/ menu (of alternative choices), a text/ bitmap display, and an edit field. None of these controls are fed with exact positions or sizes from the corresponding application, since we are not dealing with user interfaces based on the windows metaphor, but mostly with modal interaction objects that have the exclusive attention of the user.

Experiences

We are developing a C++ application framework (*independent application framework, iAF*) that adheres to the principles described above. We use iAF not only for the development of our creativity support tool PalmBeach (Magerkurth & Prante, 2001), but also in the development of a number of commercial PDA entertainment software titles.

Taken that computer games may not be representative of all mobile applications, it can be shown that there is a surprising amount of similarities in user interfaces even between full blown desktop applications and mobile software (Magerkurth, 2001). By keeping a high level of abstraction regarding the user interaction code, the development cycle of a software title can be shortened significantly, especially when targeting multiple mobile systems.

Provided that the user interfaces are implemented using scaled, system-specific functionality it becomes trivial to serve different platforms. Applications based on iAF, e.g., are 100% source code compatible amongst all supported platforms. This is not only for the benefit of

unifying the mobile software infrastructure, but it addresses a further issue in software development in the context of mobile computing: Mobile devices hardly ever host their own development tools due to limited resources. Thus, cross-platform compilers running on well equipped desktop machines have to be used, instead. This implies that the development process is subject to a permanent overhead, for testing can only occur after an upload on the mobile device or inside an emulator. Therefore, a very positive side-effect of the use of our framework is a speed-up of the development. This is due to building desktop executables most of the time which permits test-drives of native applications instead of crawling through emulated code.

Conclusion

What we have discussed, is primarily geared towards user interaction on mobile devices and its implications on software development. We have proposed scalable interaction objects. Regarding the constraints of mobile computing, the next logical step would be augmenting scalable user interface implementations to scalable *content*, which also closely relates to WML and Web-Clipping in comparison to HTML.

References

- Bager, J., & Bleich, H. (2000): WAP-Galerie. In: c't Magazin 9/ 2000, S. 200 – 207.
- Greenberg, S., Boyle, M. & Laberge, J. (1999): PDAs and Shared Public Displays: Making Personal Information Public, and Public Information Personal, im Druck
- Magerkurth, C. (2001): Programming Windows To Create Palm Games. In: Proceedings of the Game Developers Conference 2001. San Francisco: CMP Media, 2001.
- Magerkurth, C. & Prante, T. (2001): Metaplan für die Westentasche - Mobile Computerunterstützung für Kreativitätssitzungen. In: Tagungsband der GI-Fachtagung Mensch & Computer 2001 (MC'01). Bad Honnef: Teubner Verlag, 2001.
- Myers, B. A., Stiehl, H. & Gargiulo R. (1998): Collaboration Using Multiple PDAs Connected to a PC. In: Proc. of the ACM Conference on Computer Supported Cooperative Work (CSCW'98), 1998, 285-294.
- Rekimoto, J. (1998): A Multiple Device Approach for Supporting Whiteboard-based Interactions. In: Proceedings, CHI'98: Human Factors in Computing Systems, 1998, 344-351.

Bewertung von WAP-Anwendungen

Entwicklung von Kriterien und Erfahrung bei der Qualitätsbewertung in der Praxis

Lehrstuhl für Wirtschaftsinformatik III

Holger Nösekabel, Franz Lehner

Universität Regensburg

Universitätsstraße 31

93053 Regensburg

Telefon: (0941) 943 - 3201 Telefax: (0941) 943 - 3211

Email: Holger.Noeskabel@wiwi.uni-regensburg.de, Franz.Lehner@wiwi.uni-regensburg.de

Durch den verstärkten Einsatz von mobilen Endgeräten wie z.B. Mobiltelefonen oder PDAs (teilweise mit Modemanbindung) wächst der Bedarf an Anwendungen, die speziell auf die mobile Nutzung ausgerichtet sind. Viele Firmen bieten deshalb neben einer traditionellen Website auch eine entsprechende Umsetzung für das Wireless Applikation Protocol (WAP) an. Aufgrund der allgemeinen Zunahme von M-Commerce Anwendungen (vgl. Berlecon 2000, Müller-Veerse 2000) stellt sich nun auch in diesem Bereich die Frage nach der Qualität und der Benutzerfreundlichkeit dieser WAP-Angebote. Für das World Wide Web existieren bereits mehr oder weniger umfangreiche Kriterienkataloge, die als Richtlinie für gutes Design verwendet werden können. Zwar gibt es keine einheitliche, allgemein gültige Maßstäbe, aber bei der Erstellung einer neuen Website können bereits wichtige Qualitätsmerkmale berücksichtigt werden.

Für WAP-Seiten gibt es solche Richtlinien erst ansatzweise, wobei die Entwicklung mit dem Erscheinen der ersten Styleguides für graphische Benutzeroberflächen vergleichbar ist. Zwar haben Firmen wie Telstra, Phone.com oder Sprint Style-Guides erstellt, an die sich externe Programmierer bei Auftragsarbeiten nach Möglichkeit halten sollten, diese sind zum Teil allerdings sehr eng auf die Bedürfnisse der einzelnen Firmen zugeschnitten. Andererseits könnte man auch versuchen, die Kriterienkataloge für Webseiten auf ein WAP-Angebot zu übertragen. Hier ist aber die technische Ausgangsbasis so unterschiedlich (Displaygröße, Farbtiefe, Umfang Befehlssatz HTML/WML, Nutzung von Plug-Ins, Übertragungsgeschwindigkeit, Dateneingabe), dass eine unmodifizierte Übernahme der Kriterien zu keinem befriedigenden Ergebnis führen wird.

Sinnvoller ist es, bestehende Kriterienkataloge und Style-Guides zu untersuchen, Gemeinsamkeiten herauszuarbeiten und die gewonnenen Erkenntnisse um Erfahrungswerte zu ergänzen. Durch praktische Anwendung kann der Katalog dann weiter verfeinert oder korrigiert werden. Nach dieser Vorgehensweise wurde ein Kriterienkatalog erstellt, der im Folgenden kurz vorgestellt werden soll. Die im Überblick skizzierten Kriterien wurden bereits zur Bewertung von WAP-Angeboten verwendet. In der Präsentation werden sowohl die Kriterien als auch die Erfahrungen bei ihrer Anwendung zur Bewertung folgender WAP-Angebote näher vorgestellt: Yahoo! (USA und Deutschland), Lycos und Spray.net. Die grobe Gliederung unterteilt sich in fünf Abschnitte: Technik, Layout und Design, angebotene Dienste, Benutzerfreundlichkeit und Informationsaufbereitung. Jeder Abschnitt setzt sich wiederum aus etwa vier Unterpunkten

zusammen, die jeweils ein Teilgebiet näher untersuchen.

Im **Technik-Bereich** wird untersucht, inwieweit der Benutzer das Angebot problemlos nutzen kann, ohne auf technische Probleme zu stoßen. Als Eckdaten dienen hier die *Ladezeiten* für die einzelnen Seiten und deren *Größe*. Daneben wird noch die *Verfügbarkeit des Angebotes* untersucht; neben einer subjektiven Beurteilung kann dieser Punkt auch automatisch durch entsprechende Service-Monitore (z.B. von der Firma The Intranet Service Solution GmbH) bewertet werden. Als letzter Punkt sollten sich die angebotenen WAP-Seiten auf allen *handelsüblichen Browsern darstellen* lassen.

Die Untersuchung und Bewertung des **Layout und Design** gestaltet sich auf Grund der begrenzten Displaygröße bei den verwendeten Endgeräten schwierig. Einerseits können Elemente durch das geringe Platzangebot nicht immer optimal platziert werden, andererseits ist es für den Benutzer wichtig, dass wichtige Funktionen schnell zugänglich sind. Zuerst kann die *Aufteilung der Elemente* (die entweder informationstragend, funktional oder schmückend sind) untersucht werden. Jedes dieser Elemente hat eine spezielle Funktion: Informationselemente sind die Elemente, die dem Benutzer das gewünschte Wissen vermitteln sollen. Sie sollten immer vorhanden sein, damit der Benutzer aus der Seite einen Nutzen ziehen kann. Funktionale Elemente dienen der Steuerung des Angebotes. Neben Buttons zum Vor- und Zurückblättern sind z.B. Eingabemasken und Links typische Vertreter. Auch sie sollten immer vorhanden sein, da der Benutzer ansonsten auf die Navigationshilfen des Browsers zurückgreifen muss – fehlen diese Hilfen, so wird dem Benutzer die Navigation unnötigerweise erschwert. Die dritte Art der Elemente sind Schmuckelemente, die keine weitergehende Funktionalität aufweisen und nur zur Strukturierung und Auflockerung der Inhalte dienen. Sie können ohne weiteres fehlen, allerdings stellen sie nützliche Hilfsmittel dar, mit denen man den Lesefluss des Benutzers begrenzt steuern kann.

Wie im Endeffekt das Design der Seiten auch aussehen mag, so sollte es dennoch auf die *Bedürfnisse der Zielgruppe* zugeschnitten sein und nicht all zu sehr vom eventuell bereits vorhandenen *Firmendesign* abweichen. Beispielsweise ist eine informationsorientierte Zielgruppe (z.B. Manager) darauf aus, schnell an die benötigten Informationen zu kommen. Informationstragende und funktionale Elemente sollten in diesem Fall überwiegen. Eine etwas jugendlichere Zielgruppe würde mehr Wert auf ein attraktiveres Design mit vielen Schmuckelementen legen. Weiterhin sollte das Design auf allen Seiten möglichst identisch sein. Nicht nur wird dadurch der Wiedererkennungswert erhöht, es erleichtert auch die Arbeit mit der WAP-Site wenn die Funktionselemente konstant an der gleichen Position vorhanden sind und nicht den Platz wechseln oder ganz verschwinden.

Bei den **angebotenen Diensten** wird sowohl die *Qualität* als auch die Quantität einer näheren Betrachtung unterzogen. Zuerst wird aufgelistet, welche Bedürfnisse von dem Angebot abgedeckt werden. Bei WAP-Portalen wird in der Regel ein Linkkatalog angeboten, *Zusatzdienste* umfassen die Abfrage und das Verfassen von Emails, einen Terminkalender oder ein Adressbuch. Werden Links oder eigener Content angeboten, so ist auch die Qualität von Belang. Links sollten funktionieren und dem Benutzer die gewünschten Informationen liefern. Letzteres gilt auch für den selbst erstellten Content. Eng damit verknüpft ist die *Verfügbarkeit* von Inhalten und Links. Hier stellt sich das Problem, das Links zu Dritten in der Regel nicht im eigenen Verantwortungsbereich liegen und man hier keine Verbesserung erreichen kann. Als Ausweg bliebe nur, Links die dauerhaft schlecht erreichbar sind nicht anzubieten. Denn obwohl der Anbieter nicht

verantwortlich ist, kann nicht ausgeschlossen werden, dass sein Image beim Benutzer trotzdem leidet. Fehlt zudem eine *Feedbackmöglichkeit*, kann der betroffene Benutzer den Anbieter von diesem Problem (oder auch von anderen Problemen) nicht in Kenntnis setzen.

Ein anderer Servicedienst besteht darin, dem Benutzer eine Funktion an die Hand zu geben, mit der er die WAP-Site an seine speziellen persönlichen Bedürfnisse anpassen kann. Durch die *Personalisierung* wird sichergestellt, dass die Informationen noch leichter und schneller zugänglich gemacht werden. Meistens wird eine Registrierung im Web vorausgesetzt, bei der jeder Benutzer eine Kennung und ein Passwort generiert.

Unter der Überschrift der **Benutzerfreundlichkeit** wird zunächst - eng mit der Personalisierung verknüpft - die *mobile Nutzbarkeit* untersucht. Relevant ist hier, ob ein Benutzer auch ohne einen festen Anschluss Modifikationen vornehmen kann, sei es dass er neue Termine einträgt oder die Einstellungen für seine Mailbox verändert. Ebenso wird untersucht, in welchem Umfang Eingaben getätigt werden müssen und wie aussagekräftig eventuelle Fehlermeldungen sind. Auch *Navigationsmittel*, die von der Site bereitgestellt werden, sind hilfreich, da sie die Navigation erheblich erleichtern. Verlässt sich eine Site auf die Navigationsmittel des Browsers (z.B. Back-Button, Home-Button), kann dies bei nicht konformen Browsern zu Problemen führen, zudem muss der Benutzer zwischen den Buttons der Site und des Browsers wechseln. Bietet die Site zudem noch eine *Standortbestimmung im Angebot* an, wird dem "Lost in Cyberspace"-Effekt vorgebeugt.

Der Abschnitt **Informationsaufbereitung** beschäftigt sich mit den Informationen, die der Benutzer auf Grund seiner Aktivitäten bekommt. Zuerst kann die *Informationsdichte* gemessen werden. Sie wird ermittelt, indem die Anzahl der Zeichen der informationstragenden Elemente in Relation zu der Gesamtzahl der darstellbaren Zeichen auf dem Display gesetzt wird. Eine hohe Informationsdichte stellt zwar mehr für den Benutzer wichtige Daten dar, wirkt aber auch weniger übersichtlich, vor allem bei kleinen Displays. Somit ist auf Menüseiten eine relativ geringe Informationsdichte sinnvoll, auf Seiten mit Content den der Benutzer angefordert hat eine eher hohe Informationsdichte. Daneben ist auch die *Seitenlänge* vor allem bei kleinen Displays von Relevanz. Der Benutzer sollte nur in begrenztem Umfang scrollen müssen, um alle Inhalte zu bekommen. Auch hier sollte wieder zwischen Informationen, die der Benutzer durch seine Eingaben angefordert hat, und nicht angeforderten Informationen unterschieden werden. Um die *Navigationsstruktur* einer WAP-Site darzustellen, werden die Menüs und Untermenüs in einer Baumstruktur oder einer Hypertextflowchart dargestellt. Sind die Menüs relativ einheitlich in der Verzweigungstiefe, erleichtert dies dem Benutzer die Orientierung in der Struktur. Gegebenenfalls sind aber auch Schwerpunkte in bestimmten Bereichen gewollt oder unumgänglich, sollten aber soweit wie möglich vermieden werden. Als letzter Punkt ist auf die *korrekte Sprache* zu achten. Tippfehler können je nach Auftreten durchaus schwerwiegende Konsequenzen nach sich ziehen (z.B. bei Börsenkursen). Sie treten naturgemäß weniger auf, wenn sich der Content auf die Präsentation von Links beschränkt oder wenn Volltexte von professionellen Zulieferern bezogen werden (z.B. Nachrichtenagenturen).

Bei der Untersuchung der verschiedenen Angebote blieb der mobile Aspekt dabei nicht nur auf WAP beschränkt, sondern es wurden auch die im WWW vorhandenen mobilen Leistungen betrachtet. Neben Klingeltönen und Logos für Mobiltelefone gab es unter anderem proprietäre Payment-Methoden, Email per Voice Messaging oder die Möglichkeit, an Auktionen teilzunehmen. Mit diesen Methoden versuchen die Anbieter die (momentanen und zukünftigen)

Bedürfnisse der Benutzer soweit abzudecken, dass einmal gewonnene User nicht zu Konkurrenzangeboten wechseln. Gerade in der jetzigen Anfangsphase ist es wichtig, sich einen Kundenstamm mit einem qualitativ hochwertigem Angebot aufzubauen.

Literaturverzeichnis:

Ballard, B., Miller, B.: *HDML Style Guide for the Sprint PCS Wireless Web Browser*.

<http://www.usableproducts.com/wireless/SprintPCSHDMLStyleGuide3.pdf>, Stand: 30.10.2000, Gelesen:
16.01.2001.

Berlecon (Hrsg.): <http://www.berlecon.de/studien/mobile/index.html>; Studie "Internet Mobil? Eine Bestandsaufnahme Mobiler Datendienste in Deutschland", 13.12.2000

Van der Heijden, M., Taylor, M.: *Understanding WAP*. Artech House, Boston/London 2000

Muller-Veerse, N. J.: *IP Convergence: The Next Revolution in Telecommunications*. Artech House, Boston/London 2000

Phone.com: *WML Application Style Guide*. http://www.phone.com/pub/wml_style.pdf, Stand: 10.08.2000, Gelesen:
16.01.2001.

Schmitzer, B., Butterwegge, G.: M-Commerce. In: *Wirtschaftsinformatik*, Bd. 42, 4/2000, 355-358

Telstra: http://www.telstra.com.au/mobilenet/phones/wap2/docs/WAP_Design_Guide.doc. Stand: 10.08.2000, Gelesen:
16.01.2001.

Tilman, J.: *Web-Ergonomie. Theorie und Evaluation am Beispiel der HypoVereinsbank*. Regensburg 1999.

Context-Aware Telephony over WAP

ALBRECHT SCHMIDT^{*}, ANTTI TAKALUOMA[#] AND JANI MÄNTYJÄRVI[#]

^{*}*TecO, University of Karlsruhe, Vincenz-Prießnitz-Str.1, 76131 Karlsruhe, Germany (✉)*

[#]*Nokia Mobile Phones, Oulu, Finland*

albrecht@teco.edu, antti.takaluoma@nokia.com, jani.mantjarvi@nokia.com

Telephone: +49 721 / 6902-29

Fax: +49 721 / 966 3418

Abstract. In this paper we introduce a novel approach to share context to enhance the social quality of remote mobile communication. We provide an analysis of how people start a conversation in situations where they meet physically, especially looking on the influence of the situation. Than this is compared to the way remote communication is initiated using mobile phones. The lack of knowledge about the situation on the other end leads to initiation of calls that are not appropriate in the situation. The solution we propose is to exchange context information before initiating the call. We implemented this concept using the Wireless Application Protocol (WAP). The WML-based Application Context-Call offers a phone interface that provides information about the receiver when setting up a call. The caller can than decide based on that information to place the call, to leave a message or to cancel the call. Privacy issues that arise from this technology are discussed, too.

Keywords: Context awareness, communication setup, mobile telephony, social communication, wireless application protocol (WAP)

1. Introduction

Mobile phones became a ubiquitous companion to many people over recent years. The way people use phones, especially what way of usage is regarded socially acceptable, evolves over time and differs according to the group of users, types of people, and also to the country. One common understanding that is shared over most of the users is that the one who uses the mobile phone is responsible to set the phone in a mode that is appropriate for the situation he or she is in. In most cases this is a binary decision – switch the phone off or leave it on! This puts the trade-off:

- *When switching off the phone:*
the user may miss an very important call, but he will certainly not be disturbed
- *When leaving on the phone:*
the user will certainly not miss a call, but there may be calls that are not appreciated in the situation

When switching off the phone there is also the option to use the voice message system; but this changes the communication paradigm from

synchronous communication to asynchronous communication. It also put the burden to the user to check regularly for messages.

In this paper we investigated how to enrich the set-up of phone communication by sharing context information. Our motivation is based on observations of how people start a face-to-face conversation, and what rules they follow according to their situation. We than compare this to the way remote communication is set up with current phone technology. Furthermore we present an application context-call implemented using WAP that facilitates a richer and context-aware way for setting up remote mobile communication.

2. Communication in Context

Starting a conversation when meeting people face-to-face is completely different from setting up a remote communication with current technology. To motivate the context-call application that narrows the gap we first provide a brief comparison between communication in a

social environment and technically enabled remote communication.

2.1. Communication in a social environment

When observing how communication is established in a social setting we can see the important fact: **situation matters!** In general we see that the context and situation of both possible communication partners are taken into account when the decision is made to start a conversation or not [1]. These rules are a part of our social knowledge and children learn them very early; so the process of deciding to communicate in a certain situation to someone else is based on an implicit analysis of the mutual situations.

The person who is making the decision to approach a possible communication partner or not takes the following facts into account (this will most likely happen implicitly):

- How important is it for me to communicate now?
- How convenient seems it for the communication partner to be interrupted?
- What is the relation between the communication partners?
- What type of conversation will it be (important to whom, how long will it take, etc.)?
- Is it socially acceptable to start a conversation on a certain topic in this situation?

2.2. Technical Remote Communication

The initiation of a remote communication differs significantly from face-to-face communication; in remote communication the caller faces usually a situation where he or she has to make an assumption about the context of the person he or she is going to call (he is at work, she is probably back from the meeting, etc.). Then when the call is made, often at the beginning of the conversation, a question about the situation is posed, e.g.: *Where are you? What are you doing?* or *Are you able to speak?* In this way the knowledge of the situation is shared and this influences the following communication. Exchanging context information at this point is

often to late because the receiver is already interrupted. Examples of typical replies are *It is not really important, I call you later* or *If I knew that you are in a meeting I would not have called.* Considering modern phones (mobile phones, ISDN, etc.) the called party gets at least the information who is calling and can then decide whether to answer the call or not. When the caller is known by the called party, he can control whether to answer or not according his own context but without knowing how important the caller's message is.

Some mobile phones offer caller groups, enabling to comprise certain names and numbers of the phone book as groups e.g. all, family, VIP, friends, colleagues, others, see [2] for an example. By setting specific caller groups in certain profile the called party can prioritize (allow or prohibit) incoming calls according to the predefined contexts. However, the problem is still the same: "no knowledge of the situation of the called party, and no knowledge of the importance of the message of the caller". This concept also carries a major problem – e.g. a call from a person in the family made from unknown number (e.g. in a hospital) would be blocked because the phone is set to a mode where only calls from family members are allowed.

Considering line-telephony the caller knows at least in which location the called party is (e.g. home number, work number) and can then make an assumption of the situation. In mobile telephony the caller faces a more different task – the called party could be anywhere and this makes it very difficult to guess the context of the person he or she is going to call.

There are examples where remote communication using computers offers information about the possible communication partner. One widely used system is ICQ [3]. ICQ is an Internet tool that informs users who is on-line at the same time and allows them to setup a communication channel. A further example on how to increase social awareness on the WWW is described in [4]. The project GarblePhone researches ways that allow sharing moment-to-moment level of activity between people using an audio communication system [5].

3. Context-Call - a WAP-based prototype

Sharing the current context of a mobile phone user appears as a possible way to enable remote mobile communication that is closer to face-to-face communication. In the prototype described in this section we provide means for a mobile phone user to give selectively information about the current context to possible communication partners. Based on this information the caller can decide - now knowing his own context implicitly and also having information about the other end - whether it is a good opportunity to place a call or not.

3.1 Motivation - a Scenario

To motivate the idea of context-call we present the following scenario, composed of three situations where someone wants to call someone else.

A) Mike, a colleague of Mary, is at a customer site. He is calling Mary who is at work. He wants to ask her if she is joining him for a drink later. He calls her mobile phone number using *context-call*, now he gets back the message “I am in a meeting, please do not disturb!” together with the choice to place the call anyway, to leave a message, or to cancel the call. He decides that the question is not that important and he leaves a message.

B) Harry, another colleague of Mary, recognizes that she took the slides with the wrong figures to the meeting. He immediately calls her using *context-call*, he also gets the message “I am in a meeting, please do not disturb!” and the choices. He selects “place the call anyway” and gets her on the phone.

C) Sometime later a customer care person from her bank calls Mary using *context-call*, she also gets the message “I am in a meeting, please do not disturb!” and the choices. She realizes that this probably not a good time to talk to Mary about the new investment fond, so she cancels the call.

Using a standard phone Mary would have had the choice to switch the phone on or off. In any case she would either have missed a very important call or she would have been annoyed by two calls that were inappropriate in her current situation.

It can be seen from the scenario that the information about the context is helpful for both



Figure 1: Select a Context.

sides. Sharing contextual information gives people the chance to apply the rules of face-to-face communication also when calling over a phone. By providing the context to the caller there is an implicit negotiation, where the caller balances his own need to establish the communication against how receptive the other person is.

3.2. Architecture and Implementation

The system consists of three key components: the context-call application, context-selector application, and a common database.

Each of the applications is realized using WAP and consists of front-end part based on WML and WMLScript and a backend part. The backend part is built as a CGI-Script on Apache Web-Server using the scripting language PERL. Both applications share over the backend the same database where the context information is stored.

3.2.1. Context-Selector Application

The context selector application is a WAP-application where the user of a mobile phone can set the information that is given to a caller. The user has the choice of a number of pre-defined text strings, such as *Free*, *Meeting*, *Working*, *at Home*, or *BUSY*, see figure 1. There is also an option to enter free text, if the user prefers to give more specific information. The concept is similar to the profile settings used in most NOKIA phone; for a wider use this function could be integrated in the profile selection process and would than not need any additional attention from the user, another option to automate context recognition and context selection is suggested in [6].

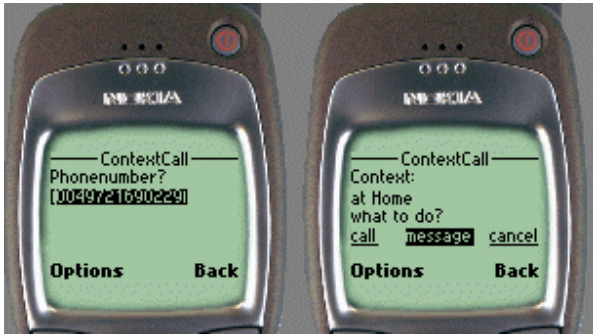


Figure 2: Using Context-Call.

When the user submits his or her context this is sent using a WAP-request to the WAP server and then handed on to the application on the server. Here a record of the phone number and the current context is put into the database. If an old record for this phone number exists this is deleted.

3.2.2. Context-Call Application

The context-call application replaces the normal interface for making phone calls; it is a phone interface based on WAP. The user opens the application and he or she is asked to provide the phone number or name of the person he or she wants to call, see figure 2. When the call identifier is submitted to the server a script on the server checks for the record containing the identifier (either directly with the phone number or using a dictionary to look up the phone number for a given name). If a context is recorded for the identifier that the user wants to call he or she gets the information with the current context (e.g. figure 2, *at Home*). If no context is available he gets the information that there is currently no context available. Additionally a menu with the following options: *to proceed the call*, *to leave a message* or *to cancel the call* is sent to the user.

The user has now the choice - knowing the context of the communication partner - to establish a connection, to leave a message, or to cancel the call. If the user selects message or cancel no phone call is established and the person to be called is not bothered at all.

3.3. On Privacy

Privacy seems to many people a concern when talking of context. Some observations at the beginning:

1. Users often share their context when they are called, e.g. "sorry I am in a meeting, could you call me later?"

2. People want to be in control of what is visible for others about them.
3. People want to know what others know about them
4. People like to share information selectively

In our first prototypical implementation we provide full control for the user about what will be displayed to other people. The user himself/herself can choose what text will appear on the callers display, and so the user is able to provide enough information for the caller to know about the context without giving away important information.

If the caller decides, "cancel" after seeing the context this is not visible to the potential receiver of the call. You could now imagine that someone checks every few minutes (or even writes a program that does this) your context and uses this information to monitor your behavior. In principle it is impossible to prevent this, because this information is needed for the caller who uses the context-call as anticipated. In our system we offer users to have a look at the statistics of their context-requests; if someone is suspicious the user is free to deny context information to this caller.

Point 4 is not yet implemented, but the system is prepared to handle this case in the following way: the user of the context-selector application can select a context description that is specific and only given to people in a buddy-list, for all other callers a generalized description is provided. E.g. the user selects the context description "in a meeting"; this description is given to people on his/her buddy list, whereas to an unknown caller the message "please don't disturb" is shown.

4. Conclusion

Context and the situation people are in play an important role when establishing communication. Comparing remote mobile communication and natural communication in a social environment we can observe that the social communication is much richer – and one important reason for this is the knowledge of the mutual context.

In this paper we describe the system *context-call*; it enables users of WAP-enabled mobile phones to share context and enables thereof a more selective way of setting up a remote communication. The user – who is potentially a receiver of a call, can provide information about his or her context to people who will call them. The caller can then

decide on knowledge of both sides if it is a good time to call now or not. Using this application can make the mobile phones less disturbing in many everyday situations with the benefit of being available any time.

In our future work we see space for enhancement in the user interface, especially the integration of context-selection part into the profile setting mechanism that is already build-in the phones seems important. Also an integration of the call application with the address book of the phone is important.

5. References

1. Chin, J., Tatchell, G., The Telephony Customer Interface: Five Perspectives on Problems and Solutions. Proceedings of CHI 96, 1996.
2. Nokia Mobile Phones. The Nokia 7110 at a Glance. 2000. <http://www.nokia.com/phones/7110/index.html>
3. ICQ Home Page. <http://www.icq.com/>. July 2000.
4. Liechti O, Siefer N and Ichikawa T. A Non-obtrusive User Interface for Increasing Social Awareness on the World Wide Web. *Personale Technologies* 3 (1&2), 1999: 22-32.
5. Jacknis, M., Sawhney, N., Schmandt, C. GarblePhone: auditory lurking. <http://www.media.mit.edu/~nitin/projects/GarblePhone/>. July 2000.
6. Schmidt, A., Aidoo, K.A., Takaluoma, A., Tuomela, U., Van Laerhoven, K., Van de Velde, W. Advanced Interaction in Context. 1st International Symposium on Handheld and Ubiquitous Computing (HUC99), Karlsruhe, Germany, 1999 & Lecture notes in computer science; Vol 1707, ISBN 3-540-66550-1; Springer, 1999. pp 89-101.

m-Commerce für Finanzdienstleister - eine Chance mit vielen Hindernissen

Michael Semrau, Achim Kraiß

Dresdner Bank AG

Konzernstab Informationstechnologie

IT Research

Jürgen-Ponto-Platz 1

D-60301 Frankfurt am Main

(michael.semrau, achim.kraiss@dresdner-bank.com)

Immer wieder ist in der Presse zu lesen: Finanzdienstleister gehören zu den großen Gewinnern des m-Commerce. Schaut man sich dagegen die real existierenden m-Commerce Anwendungen an, so findet man nur selten interessante Informationen und eine ganze Reihe von Pilotanwendungen im Finanzbereich, die noch nicht wirklich für alle Kunden verfügbar sind. Am Beispiel einer produktiven Anwendung wird im folgenden gezeigt, welche Probleme auf Unternehmen zukommen, die mobile Technologien wie WAP innovativ anwenden wollen. Dabei werden insbesondere die Felder Anwendungsdesign, Anwendungsentwicklung, Security und Infrastruktur sowie ökonomische Fragestellungen betrachtet.

1. Einführung

Die Zeiten in denen allein das Schlagwort m-Commerce eine Goldgräberstimmung auslöste, sind längst vorbei. Angesichts der realen Probleme haben sich die hohen Erwartungen längst relativiert. Mit Anwendungen für mobile Nutzer können Finanzdienstleister wie die Dresdner Bank aber eine Lücke beim Zugang zu Bankprodukten schließen. So werden neben klassischen Zugangskanälen wie der Filiale, dem Internet oder dem Call Center nun viele mobile Endgeräte als Zugänge zu Produkten und Dienstleistungen erschlossen. Vor der Entwicklung eines Bankprodukts oder einer Dienstleistung für mobile Kunden sind jedoch zunächst die Bedürfnisse mobiler Nutzer zu analysieren.

Aus unserer Erfahrung sind an jede mögliche Anwendung drei Fragen zu stellen, die alle bejaht werden müssen, damit eine Anwendung überhaupt für eine mobile Realisierung in Frage kommt:

- Ist es für den Kunden wichtig, dass er jederzeit und von jedem Ort Zugriff auf die Anwendung hat?
- Lässt sich die Bedienung der Anwendung auf wenige einfache Prozessschritte und eine einfache Benutzerschnittstelle reduzieren?
- Ist eine Personalisierung der für den Kunden notwendigen Informationen so weit möglich, dass der Zweck der Anwendung trotz der Limitierungen der Ausgabegeräte erreicht werden kann?

Aufgrund einer entsprechenden Analyse dieser Fragestellungen wurde im vergangenen Jahr eine Brokerage-Anwendung konzipiert und realisiert, die den Kunden die Durchführung von

Wertpapiertransaktionen und die Zeichnung von Aktienemissionen erlaubt. Darüber hinaus stehen verschiedene Informationsangebote zur Verfügung.

2. m-Commerce stellt neue Anforderung an das Design von Anwendungen

Sicherlich trivial ist die Erkenntnis, dass eine m-Commerce Anwendung sich nicht als einfache Kopie einer entsprechenden e-Commerce Anwendung realisieren lässt. Wesentlich schwieriger ist dagegen schon die Frage zu beantworten, wo denn eigentlich die Unterschiede liegen, die bei der Planung und beim Design einer Anwendung für mobile Nutzer berücksichtigt werden müssen. Unsere Erfahrungen zeigen, dass jeder Bereich der Anwendung genau analysiert werden muss. Im folgenden werden einige Beispiele dargestellt:

- Die Bereitstellung von Informationen für die Zeichnung von Aktien ist an bestimmte rechtliche Voraussetzungen gebunden. Diese werden im Online-Auftritt in einem längeren Disclaimer dargestellt. Eine gleichartige Darstellung auf einem Endgerät mit einem kleinen Display ist nicht praktikabel. Hier musste eine neue Lösung gefunden werden, welche die rechtlichen Anforderungen ebenfalls erfüllt.

Dieses Beispiel zeigt, dass Anwendungen im m-Commerce nicht nur aus IT-Sicht besondere Aufmerksamkeit verlangen, sondern durchaus auch aus fachlicher Sicht neue Anforderungen stellen.

- *Eingaben sind auf das notwendige Mindestmaß zu beschränken.*
Auf Mobiltelefonen sind Eingaben einer der kritischen Faktoren für den Erfolg einer Anwendung. Insbesondere alphanumerische Eingaben sind zu vermeiden. Im Zweifel sollte auf bestimmte Funktionalitäten verzichtet werden, wenn diese nur durch umfangreiche Eingaben und - damit meist einhergehend - nur durch mehrere, eher komplexe Prozessschritte realisiert werden können. Eingaben sollten auch erst dann verlangt werden, wenn diese aus Gründen des Programmablaufs unbedingt benötigt werden.
- *Nur wenige Informationen sind wirklich wertvoll.*
Auch wenn GPRS, der neue Mobilfunkstandard, eine wesentliche Steigerung der Bandbreite verspricht, ist die Menge der für den Nutzer verwertbaren Informationen durch die Bauart der Endgeräte stark eingeschränkt. Aus diesem Grund ist ein strikter Zuschnitt der angezeigten Information auf die Bedürfnisse der Nutzer notwendig. Welche Informationen braucht welcher Nutzer in welchem Prozessschritt der Anwendung?

3. Anwendungsentwicklung per Hand

In einem Forschungsprojekt sind wir der Frage nachgegangen, wie WAP-Anwendungen möglichst effizient entwickelt werden können. Dieses Forschungsvorhaben wurde durch eine Reihe von Problemen angestoßen, vor die wir bei unserer ersten Beschäftigung mit dem Thema WAP gestellt wurden:

- Während es für fast jede Programmiersprache, jede Skriptsprache und natürlich erst recht für HTML viele verschiedene Entwicklungstools gibt, müssen WAP-Anwendungsentwickler mit geringer Unterstützung auskommen. Eine Integration in bestehende Entwicklungssysteme findet nicht oder nur unzureichend statt.
- Bestehende Toolkits und Simulatoren decken nur Grundfunktionen ab und weichen in ihren Darstellungsmöglichkeiten teilweise stark von realen Endgeräten ab.
- Jedes Endgerät unterscheidet sich hinsichtlich der Funktionalität und der Bauart von anderen Geräten, so dass die Darstellung von Inhalten an das jeweilige Gerät angepasst werden muss. Dieser Effekt wird durch die in den Endgeräten verwendeten Browser noch verstärkt. So haben wir allein bei verschiedenen Nokia 7110

Mobiltelefonen im Laufe der Zeit mehrere Browservarianten mit unterschiedlichem Funktionsumfang kennen gelernt.

Aus diesen Erfahrungen schlussfolgerten wir:

- Bestehende Simulatoren sind zur Anwendungsentwicklung nicht geeignet und sollten allenfalls für erste Tests verwendet werden. "Richtige" Tests müssen unbedingt mit den Zielendgeräten durchgeführt werden.
- Es ist einem Entwickler nicht zuzumuten die Anwendung an alle denkbaren Endgeräte anzupassen. Aufgrund der damit verbundenen Komplexität wäre dies letztendlich auch nicht finanzierbar.

Insbesondere die letzte Feststellung motivierte uns zur Entwicklung eines Frameworks zur effizienten Realisierung von WAP-Anwendungen. Dabei wurden neben Best Practice Empfehlungen vor allem zwei Lösungsansätze auf Basis von Java und XML-Technologien entwickelt, welche die Anwendungsentwicklung wesentlich vereinfachen sollen.

Die erste von uns entwickelte Lösung geht davon aus, dass für die Erstellung der Präsentationsschicht einer Anwendung Java-Servlets verwendet werden. Aufgrund dieser Annahme wurden Java-Klassen entwickelt, die alle nötigen Funktionen bereitstellen, um einen WML-Output zu generieren (siehe Abbildung 1). Ausgehend von einer Basisklasse für „Standard-WML“ wurde für jedes Endgerät bzw. für jede Entwicklungsumgebung eine spezialisierte Klasse bereitgestellt, die auf die Spezifika des Browsers eingehen. Eine statische Methode liefert der Anwendung in Abhängigkeit vom Endgerät (erkennbar am User-Agent String) die entsprechende Spezialisierung zurück.

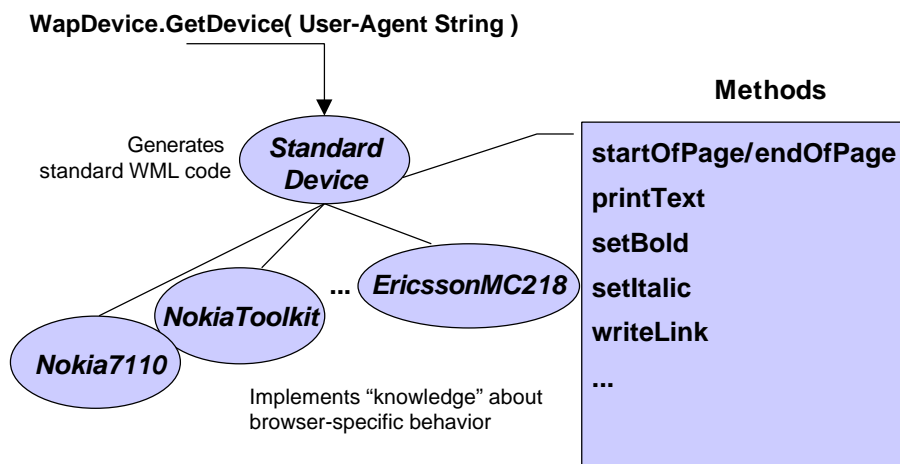


Abbildung 1: Spezialisierte Java-Klassen abstrahieren von den Geräte-Spezifika

Typische Spezifika, die von den realisierten Java-Klassen abgedeckt werden, sind:

- Anpassung der Größe von Grafiken für die verschiedenen Displays
- Anpassung von Grafiken, die von einigen Endgeräten invertiert angezeigt werden
- Ersetzung einzelner Befehle (BOLD, ITALIC, ...) durch Alternativen, falls diese auf dem spezifischen Browser nicht verfügbar sind

- Anpassung des Seitenlayouts, z.B. automatische Generierung von Zeilenumbrüchen auf verschiedenen Endgeräten für ein einheitliches Erscheinungsbild
- Behebung bekannter Bugs, wie etwa die Anpassung der Länge von Eingabefeldern, die bei einem Endgerät gegenüber der vorgesehenen Anzahl um 1 verkürzt wird

Die vorgestellte Lösung war in der Lage, die wichtigsten Probleme bei der Erstellung von WAP-Anwendungen für den Entwickler transparent zu beheben. Die weitere Arbeit mit den Java-Klassen hat gezeigt, dass diese performant arbeiten und für Java-Entwickler ohne großen Schulungsaufwand zu verwenden sind. Der große Nachteil besteht in der nicht vorhandenen Trennung zwischen Content, Logik und Präsentation in der Anwendung, so dass die oft gewünschte Arbeitsteilung zwischen Designern einerseits und Entwicklern andererseits nicht umgesetzt werden kann.

Da aber die meisten Entwicklungsprojekte bereits auf dieser Arbeitsteilung beruhen, entwickelten wir aus diesen Erfahrungen heraus eine zweite Lösung, die es erlaubt, jede beliebige WML-Ausgabe einer Anwendung durch XSLT-Stylesheets in Browser-spezifisches WML zu transformieren.

Zunächst gehen wir davon aus, dass die Anwendung z.B. mittels (statischer) WML-Seiten und (dynamischer) XSP oder JSP eine WML-Ausgabe erzeugt, die am WML-Standard orientiert ist (siehe Abbildung 2). Dieses „Standard-WML“ wird dann mittels XSLT in ein Browser-spezifisches WML transformiert. Dabei haben wir als Umgebung für unsere Versuche sowohl Apache's Cocoon, als auch den WebSphere Transcoding Publisher von IBM verwendet.

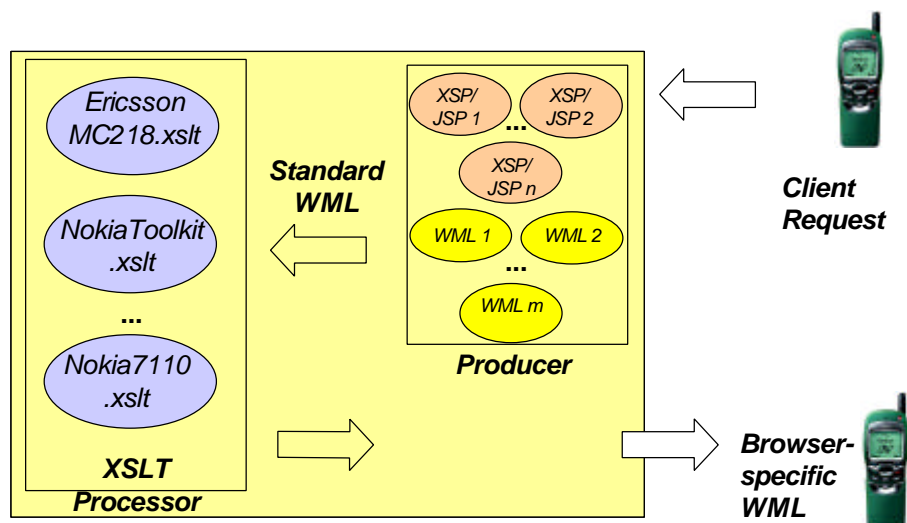


Abbildung 2: Ein XSLT basierter Ansatz gleicht unterschiedliche Endgeräteeigenschaften aus

Diese Lösung ermöglicht die Trennung von Content, Logik und Präsentation in der Anwendung und erlaubt es dem Designer, vollständig von Browser-Spezifika zu abstrahieren. Sie ist der Java-Lösung jedoch unterlegen, wenn es auf Performance ankommt. Erschwerend kommt hinzu, dass zumindest für die Erstellung der Stylesheets tiefes Wissen über XML-Technologien vorhanden sein muss. Die von uns erstellten XSLT-Stylesheets sind teilweise komplexer und aufwendiger zu pflegen als die entsprechenden Java-Klassen. Beispielsweise

ist die Umwandlung von Textelementen in Großbuchstaben mittels JAVA leichter und "geradliniger" zu realisieren als mittels XSLT.

4. Eine eigene Infrastruktur ist notwendig, erfordert aber besonderen Aufwand

Während die meisten Informationsangebote für WAP heute über beliebige WAP-Gateways erreicht werden können, stellen insbesondere die Finanzdienstleister ihre Angebote oft ausschließlich über eigene Gateways bereit. Dies ist insbesondere dann der Fall, wenn Transaktionen ausgeführt und vertrauliche Daten übermittelt werden müssen. Diese in der fehlenden End-to-End-Security begründete Vorgehensweise bedeutet einen enormen Aufwand für die Einführung der noch jungen und bezüglich ihrer Zukunftschancen nur schwer abzuschätzenden WAP-Technologie.

Für die bereits erwähnte Brokerage Anwendung wurde ebenfalls eine eigene Infrastruktur aufgebaut. Nach der Konzeptionsphase war der Aufbau der gesamten Infrastruktur relativ schnell zu realisieren. Die eigentlichen Probleme stecken hier eher im Detail. So zeigten sich Inkompatibilitäten zwischen dem verwendeten Gateway und einigen Endgeräten. Und auch die Umsetzung der Planungen im Bereich Sicherheit wurde durch Probleme mit Zertifikatsformaten erschwert. Insgesamt kann auch festgestellt werden, dass heutige WAP-Gateways meist nur schwer in bestehende Systemmanagementprozesse eingebunden werden können.

5. Fazit

Unsere Erfahrungen mit mobilen Anwendungen und der zugehörigen Infrastruktur belegen, dass m-Commerce Anwendungen, die dem Nutzer einen echten Mehrwert bieten sollen, heute nur mit einem erheblichen Aufwand zu realisieren sind. Dieser Aufwand lohnt sich jedoch, da so performante, auf die mobile Situation zugeschnittene und einfach zu bedienende Anwendungen entstehen können.

In der Zukunft werden wir unsere Forschung auf einen allgemeineren Transcoding-Ansatz ausweiten. So soll nicht mehr nur eine Übertragung von Standard-WML in browser-spezifisches WML realisiert werden, sondern es soll die Frage untersucht werden, wie Content in XML allgemein ausgezeichnet und dann möglichst automatisch in unterschiedliche Endformate überführt werden kann. Desweiteren werden wir die Auswirkungen neuer Mobilfunkstandards, wie etwa von GPRS, auf Bankanwendungen untersuchen, da wir hier über die reine Erhöhung der Bandbreite hinaus ein besonderes Potential sehen.

Was unsere Wünsche an die Industrie anbetrifft, so sollte insbesondere die Lösung der folgenden Themen den m-Commerce schneller voranbringen:

- Unterstützung einer mobilen Public-Key-Infrastruktur mit sicherer Ende-zu-Ende-Verbindung, die von den Anwendern akzeptiert wird
- Erweiterung bestehender Entwicklungswerkzeuge um entsprechende WAP-Funktionalitäten
- Verbesserung und Erweiterung von Transcoding Technologien für effizientere und komplexere (z.B. verkettete) Transformationen

Workshop

WAP - Interaktionsdesign und Benutzbarkeit

Workshop auf der Konferenz: Mensch und Computer 2001.

Die Workshop wird im Rahmen der ersten fachübergreifende Konferenz Mensch & Computer 2001 angeboten. Die Konferenz findet vom 5. bis 8. März 2001 in Bad Honnef (Bonn).

Einführung

Das Wireless Application Protocol (WAP) eröffnet neue Möglichkeiten, aber auch neue Herausforderungen für die Gestaltung und Entwicklung von mobilen, nomadischen Informationssystemen. In diesem Workshop sollen die Stärken und Schwächen von WAP in Bezug auf benutzerorientierte Gestaltung und technische Machbarkeit erörtert werden. Dazu sollen 15 Teilnehmer aus Wissenschaft und Praxis zu Vorträgen und zum Erfahrungs- und Ideenaustausch eingeladen werden. Insgesamt soll in diesem Workshop Wissen zu Interaktionsdesign und Benutzbarkeit von WAP gesammelt werden.

Workshop Moderatoren

[Albrecht Schmidt](#), [TecO](#), [Universität Karlsruhe](#)
albrecht@teco.uni-karlsruhe.de

Tom Gross, GMD-FIT, Sankt Augustin
tom.gross@gmd.de

Oliver Frick, SAP AG, CEC Karlsruhe
o.frick@sap.com

Einreichung bis 22.1.01

Umfang: 2-5 Seiten

**Der Workshop findet
am 7. März 2001 von
8:30 bis 11:30 Uhr statt.**

Themen und inhaltliche Ausrichtung

Kleine mobile Computer wie Mobiltelefone, Pager und PDA finden eine immer größere Verbreitung. Diese Geräte bringen die Vision, dass Zugriff auf Information und Kommunikation ubiquitär verfügbar ist, der Realität ein Stück näher. Für den Benutzer wird es somit möglich, immer und nahezu unabhängig von seinem Aufenthaltsort auf Information zuzugreifen und an Kommunikationsprozessen teilzunehmen. Entwickler können mit diesen neuen Technologien einen Zugang zu Anwendungen schaffen, der weit über den traditionellen Schreibtischarbeitsplatz hinaus geht.

Bei der Einführung dieser Technologien entstand eine sehr große Euphorie; unter Slogans wie *“WAP bringt das World Wide Web auf das Handy”* wurden Erwartungen geweckt, die sich nicht mit der Realität deckten. Dennoch geht aus Studien hervor, dass WAP bis Mitte nächsten Jahres in den meisten neu verkauften Mobiltelefonen integriert sein wird und binnen weniger Jahre die Anzahl der WAP-Benutzer die Anzahl der PC-basierten Internetnutzer um das dreifache übersteigen wird. Viele Gruppen, die sich mit der Entwicklung von WAP-basierten Anwendungen beschäftigten, haben realisiert, dass wirklich benutzbare WAP-Anwendungen gänzlich andere Charakteristika haben, als Anwendungen im WWW.

Forschungsaspekte

Bezüglich ihrer Interaktion mit dem Anwender weichen mobile Endgeräte heute konstruktionsbedingt signifikant von üblichen Arbeitsplatzrechnern ab, so dass neue *Methoden zur Analyse und Bewertung der Benutzbarkeit von Anwendungen auf mobilen Endgeräten* erforderlich sind. So erschweren beispielsweise die stark reduzierten Bildschirmgrößen die Übersicht über die Anwendung und

erfordern spezifische *Präsentationskonzepte* sowie an die verfügbare, in der Regel sehr einfach gestaltete Eingabemethaper angepaßte *Navigationskonzepte*.

Bereits heute sind dabei mobile Endgeräte bezüglich ihrer interaktionsrelevanten Ausprägung stark unterschiedlich. Erste Erfahrungen beim Einsatz von Anwendungen auf mobilen Endgeräten haben gezeigt, dass unterschiedliche spezifische Geräteeigenschaften beim Design der Anwendung berücksichtigt werden müssen. Es ist heute jedoch noch offen, wie *Anwendungen effizient für multiple und unterschiedliche mobile Geräte entwickelt werden können*. Existierende Ansätze zur dynamischen Generierung von Inhalten sind vielversprechend, ihre Anwendung für die Klasse der mobilen Endgeräte ist zu untersuchen. Offen ist dabei insbesondere die Berücksichtigung des Gerätekontexts bei der Interaktionsmodellierung.

Die beiden spezifischen Eigenschaften mobiler Computer “beschränkte Interaktion” und “hohe Mobilität” beschneiden zunächst den Einsatz eines mobilen Endgeräts als universelles Arbeitsmittel, eröffnen aber auf der anderen Seite durch die Mobilität ganz neue Situationen und Formen der Computernutzung. Erkenntnisse über die *prinzipiellen Grenzen der Anwendbarkeit mobiler Geräte* sowie über *besonders geeignete Anwendungsfelder* können die Anwendungsentwicklung früh in die richtige Richtung lenken und helfen, grobe Fehler zu vermeiden.

Liste der Themenfelder

Die potentiellen Teilnehmer werden gebeten, Beiträge oder Stellungnahmen, die sich mit einem oder mehreren der folgenden Themen beschäftigen, einzureichen:

- Interaktion mit kleinen Geräten
- WAP – verschiedene Geräte, Browser und Navigationsmuster
- Gestaltung und Layout auf kleine Bildschirmen, Ausgabepartitionierung
- Eingabeprobleme in WAP, z.B. Texteingabe, Scrollen und Auswählen
- Konsistenz in der Benutzerschnittstelle
- Analysen zur Benutzbarkeit
- Implikation der Technologie (z.B. Verzögerung) auf die Umsetzung (z.B. Design)
- Anwendungserfahrung mit WAP, z.B. für Auskunftssysteme, M-Commerce, Virtuelle Gemeinschaften
- Ortsabhängige oder kontextabhängige WAP-Anwendungen
- Personalisierung und Portale, Komplexe Systeme, Integration Web - WAP
- Entwicklungswerkzeuge für WAP-Anwendungen

Zielstellung

Ziel des Workshops ist es ein umfassendes Verständnis für die grundlegenden Fragestellungen, welche sich im Umfeld des ubiquitären Informationszugriffs mit WAP-Endgeräten ergeben, zu erarbeiten. Insbesondere sollen Antworten und Ansätze in den folgenden Teilbereichen ausgearbeitet werden:

- Was sind die Besonderheiten von WAP-Anwendungen, wie werden sie benutzt und was ergibt sich daraus für das Interaktions- und Navigationsdesign? Welche Methoden können verwendet werden, um Interaktions- und Navigationsdesign zu beschreiben und zu entwickeln?
- In welchen Anwendungsbereichen läßt sich WAP positionieren? Wo sind Grenzen durch die Benutzbarkeit gegeben und wo entsteht ein Mehrwert (z.B. M-Commerce, Communities, Auskunftssysteme)? Wie läßt sich eine WAP-Anwendung in ein großes System einfügen?

- Wie kann man die Benutzbarkeit von WAP-Anwendungen beurteilen? Wie können Werkzeuge aussehen, welche die Entwicklung und Optimierung der Interaktion unterstützen?
- Welche Randbedingungen (technisch, politisch und sozial) erschweren es, WAP-Anwendungen zu erstellen, die eine hohe Akzeptanz erfahren? Was sind Wünsche an einen zukünftigen Standard?

Zielgruppe

Der Workshop wird ein Forum bieten, in dem sich Praktiker und Wissenschaftler, die sich mit dem Entwurf, der Entwicklung und dem Test von WAP-Applikationen und den daraus resultierenden Fragestellungen beschäftigen, Erfahrungen austauschen und gemeinsam neue Ansätze entwickeln können. Zur Zielgruppe gehören insbesondere Informatiker, Psychologen, Designer und Soziologen, welche sich mit Möglichkeiten des ubiquitären Informationszugriffs und den sich ergebenden Problemstellungen beschäftigen.

Zeitplan

Einreichung bis 22.01.2001

Workshop 7.3.2001, 8:30-11:30 Uhr

Arbeitsformen und Teilnehmerzahl

Im Workshop wird es eine kleine Zahl an Vorstellungen von Projekten und Forschungsvorhaben von den ausgewählten Teilnehmern geben. Ein Schwerpunkt des Workshops liegt auf gemeinsamer Diskussion, die sowohl im Plenum als auch in Untergruppen durchgeführt wird. Die Teilnehmeranzahl wird auf 15 Personen begrenzt, um ein großes Maß an Interaktion zwischen den Teilnehmern zu gestatten.

Auswahl der Teilnehmer

Aus den Einreichungen werden von einem Programmkomitee die Teilnehmer und Vortragenden für den Workshop ausgewählt.

Details zur Konferenz

Unter dem folgenden URL finden sie weitere Details zur Konferenz Mensch & Computer 2001.

<http://mc2001.informatik.uni-hamburg.de>

Einreichung von Beiträgen

Einreichungsschluß: wird in den nächsten Tagen bekannt gegeben.

Beiträge in einem Umfang von **2-5 Seiten** bitte in **elektronischer Form** (Postscript, PDF, Microsoft Word, Framemaker oder HTML) an albrecht@teco.uni-karlsruhe.de einsenden.

Die für den Workshop angenommenen Beiträge werden auf dieser Webseite und ebenfalls in gedruckter Form veröffentlicht.