The UbicompBrowser

Michael Beigl, Albrecht Schmidt, Markus Lauff, Hans-Werner Gellersen
Telecooperation Office (TecO)
University of Karlsruhe
Vincenz-Priessnitz-Str. 1
76131 Karlsruhe (Germany)
(+49)-721/6902-59
{michael, albrecht, markus, hwg}@teco.edu

Abstract. In this paper we introduce the UbicompBrowser, a system that applies ubiquitous computing to the World-Wide Web. The UbicompBrowser extends the World-Wide Web in two ways into our everyday environments. First, it extends the browser concept by replacing the standard web user interface with a handheld access and control device and surrounding output devices. This ubiquitous user interface is determined dynamically based on the location of the handheld control. Secondly, the UbicompBrowser system extends the web concept of uniform access to resources by applying the same access method, uniform resource identifiers, to environment-specific resources, for example for access to the TV channels available on a TV set, and for access to light switches. In summary, the UbicompBrowser improves web accessibility by realizing a ubiquitous environment-based user interface, and by extending accessibility to environment-specific resources. The approach constitutes a user interface for all flexibly adapted to their environment, interfacing both global resources in the web and local resources in a given user environment.

1. INTRODUCTION

In his famous article *"The Computer for the 21st Century"* [11], Mark Weiser introduced his vision of ubiquitous computing. The cornerstones of this vision are that computers as we know them will be replaced by a multitude of networked computing devices embedded in our environments, and that these devices will be *invisible* in the sense of not being perceived as computers. A simple interpretation of this vision used by Abowd and less concerned with invisibility is, that ubiquitous computing pushes the user interface away from the desktop and into our everyday environments [1]. Exactly this is what the UbicompBrowser, which is introduced in this article, does with respect to the World-Wide Web: the UbicompBrowser pushes the web user interface into the environment, providing access to web resources through an interface dynamically composed of devices available in a given environment. But it is not only with respect to the user interface that the UbicompBrowser constitutes a move toward ubiquitous computing. In addition, the UbicompBrowser integrates environment resources such as TV channels and home controls, making these available in the same way as web resources. This integration blurs the boundary between virtual world with computer-based resources and real-world with resources that are not computer-based or at least not perceived as computer-based. The two fundamental contributions of the UbicompBrowser, the ubiquitous web user interface, and the uniform access to web resources and environment resources, will be further discussed in section 2 below.

Figure 1 illustrates the general concept of the UbicompBrowser. Central to the browser is the idea to use a mobile handheld device as control/access device, to enable access to resources and control of the environment in principle anywhere and at any time. Through the control/access device, different kinds of web and environment resources can be accessed, as

symbolized by the different media depicted in figure 1. Resources can be web documents, media streams related to receiver devices in the environment (for example TV, radio and telephone), and control sequences related to control devices in the environment (for example door buzzer, light and heating controls). Thus, the handheld control/access device seamlessly integrates the of handheld web browsers and all-purpose remote controls. In contrast to standard browsers, accessed resources are not delivered to the access device though, but dispatched to suitable output devices available in the immediate environment. The dispatching of resources is based on detection and characterization of devices in the usage environment, and rule-based matching of resource media characteristics with device capabilities. In this way, the UbicompBrowser *mediates* between the virtual world of media (documents, streams, applications) and the physical world of output devices (generic devices such as palmtops, computers, TV sets and also non-generic devices such as light and other home controls).

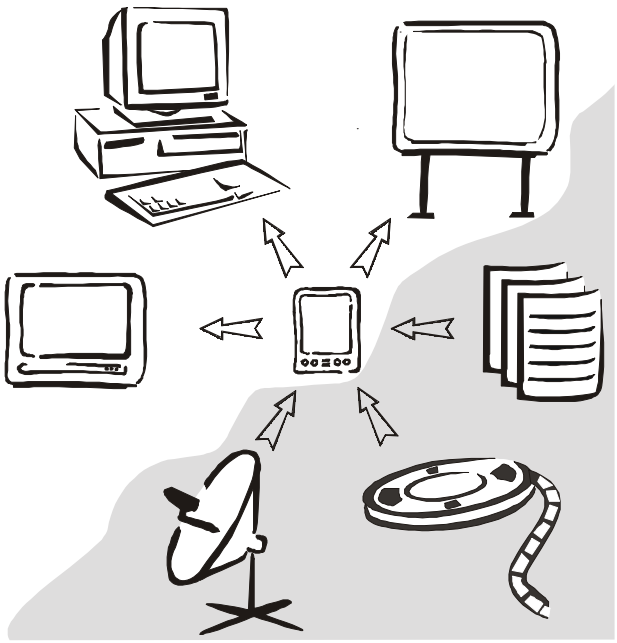

Figure 1: UbicompBrowser

Figure 2 shows a simple usage scenario to further illustrate how the UbicompBrowser works. In this example the user uses his handheld access device to requests a web resource that contains text and a video resource. The reply of the web server contains header information that is analyzed by a dispatcher running on the handheld device. The dispatcher identifies the media types and allocates these to suitable output devices in the user's environment. Which output devices are available is determined dynamically based on the relative location of the user, or more precisely the location of their handheld control/access device. In this example, a TV set capable of displaying the given video format is detected, and so video output is allocated to the TV set while text output is allocated to the handheld device itself. Note, that the dispatcher redirects the actual delivery of resources to the allocated output devices, that is resources are not simply delivered to the access device to be forwarded. The redirect is crucial as otherwise the access device would constitute a bottleneck in the system.

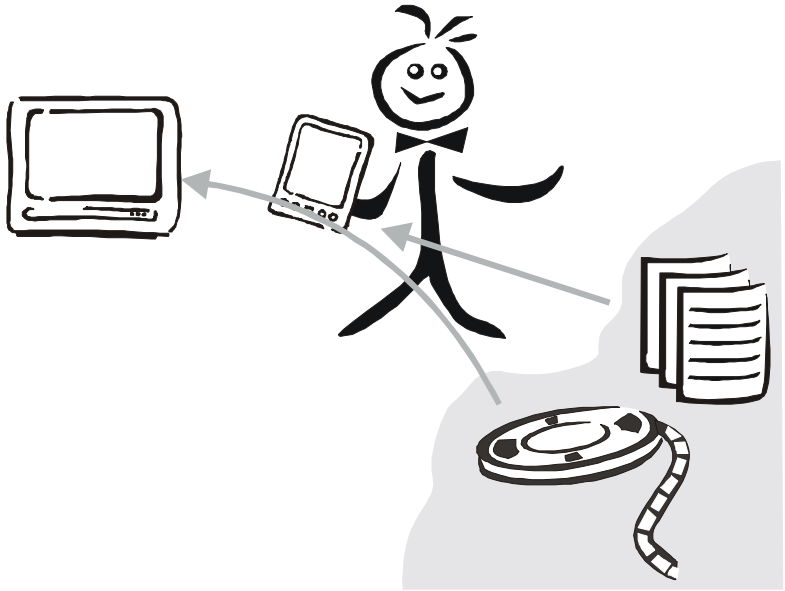

Figure 2: Usage Scenario

The remainder of the paper is organized as follows. Section 2 discusses the fundamental concepts underlying the UbicompBrowser and related work. In section 3, the system architecture is described. Sections 4 and 5 discuss key components of the system, first the detection and management of environment resources and device infrastructure, and then the dispatching of resources to devices. The paper concludes with a summary of our work and findings, and an outlook on experimentation with the system.

2. FUNDAMENTAL CONCEPTS

In this section we will discuss the two fundamental concepts and contributions of the UbicompBrowser in more depth, and with respect to related work: *truly ubiquitous web access, and uniform access to web and environment*.

2.1. Truly Ubiquitous Web Access

It is generally claimed that the World-Wide Web provides *ubiquitous* access to information resources and applications delivered over the web, based on the ubiquity of standard web browsers. Indeed web browsers are available on desktops almost anywhere, and public browsers are becoming more and more common, for instance in Internet Cafes and on multimedia kiosks. Still, we would suggest that this kind of ubiquity is rather coarse-grained in its dependence on standard desktop-based browsers. This limitation is seemingly overcome by browser implementations on mobile handheld devices, now commonly available on Personal Digital Assistants (PDA) and handheld PCs. These handheld browsers though realize only largely compromised access to web resources because of constraints regarding media presentation such as limited screen real estate, limited compute power for media streams, and limited bandwidth for network access. To address these limitations, resources are either filtered for adaptation to handheld browser capabilities or explicitly designed for delivery to handheld browsers [5]. Both approaches are not satisfactory. Filtering compromises the intended delivery of content and media and often yields unacceptably poor

quality. Explicit design of resources for handhelds implies additional development effort and compromises the fundamental web concept that content providers should be able to abstract from browser implementation.

The UbicompBrowser aims at truly ubiquitous web access anywhere without compromising content and media delivery. To this end it picks up the concept of handheld browser integrated in personal devices that ideally accompany their users most of the time. Of the two key functions of a browser, access and rendering, only the first one is allocated on the handheld device while the second one is delegated to the set of output devices that is available in the environment of the handheld device. The separation of access and rendering functionality in general increases flexibility with respect to resource access and delivery. In [7] this separation has been discussed in more length, for example describing a service that uses a mobile phone for access to web resources and a fax machine for output. Other work on multiple interacting devices and in particular integration of PDA and TV has for example been presented by Robertson et al [8]. In their PDA-ITV system information can be retrieved and browsed both on a PDA stand-alone or in conjunction with television images. In contrast to the UbicompBrowser their system is an application explicitly designed for a dual device user interface, and not a generic browsing system. Related to both PDA-ITV and UbicompBrowser are also WebTV configurations, with increasingly ”intelligent” handheld remote controls to specify requests and with TV sets capable of rendering web resources. There are substantial differences though. First, in WebTV the access to web resources is actually located in the TV and only remotely controlled, whereas in the UbicompBrowser the access is located in the handheld device; this is important with respect to personalization of access as will be discussed below in the architecture section. Second the allocation of media to devices is obviously static in WebTV where as media dispatching is dynamic in the UbicompBrowser. Third UbicompBrowser as opposed to WebTV is based on a dynamic notion of environment; which output devices are available is determined dynamically.

2.2. Uniform Access to Web and Environment

Uniform access to resources based on uniform resource locators (URLs) is a key concept of the web. A URL specifies the server on which a resource is located, the file reference on that server, and the transfer mechanism. An interesting problem is that resources can only be accessed from specific servers, and that there is no mechanism to abstract from servers. This is a drawback when we consider integration of resources in local environments. For example there is no way to request a resource ”home control” from a locally relevant environment control server unless we explicitly know the name of such a server. And if there is such a server then it is accessible from everywhere in the web which may not make too much sense for information of entirely local relevance. There are many examples of appliances that can be controlled through a standard web interface; all these examples demonstrate web-based remote control but there is no general solution for location-aware web access for local environment control.

The UbicompBrowser addresses this issue by considering the dynamically established environment as a server that holds locally relevant resources. Note, that the environment is only considered as server on a conceptual level; in the implementation the actual resources and services available in a given environment are not served by a web server but accessed directly, facilitated by an environment resource management system. For example, a home environment may be configured with a TV set, an answer machine and light control. All resources accessible from devices in this configuration are made accessible in a generic web

uniform way with URLs of the type <protocol>://local/<resource>. Related work on centralized PDA-based control of appliances has been described in [10], but the seamless integration with the is novel, providing users with access to both local and global resources based on the same protocol. With respect to the above example, the web-conform access method can be applied to the channels available on the specific TV, the recorded messages on the answer machine, and the control sequences for the lighting system, which would all be considered as environment resources. Again, it has to be stressed that in this approach the environment control is not based on a web server but on an abstract notion of local service provision.

3. SYSTEM ARCHITECTURE

The system architecture of the UbicompBrowser is described in figure 3. Below we will first describe the system components and then consider the allocation of components to the physical infrastructure underlying the UbicompBrowser.

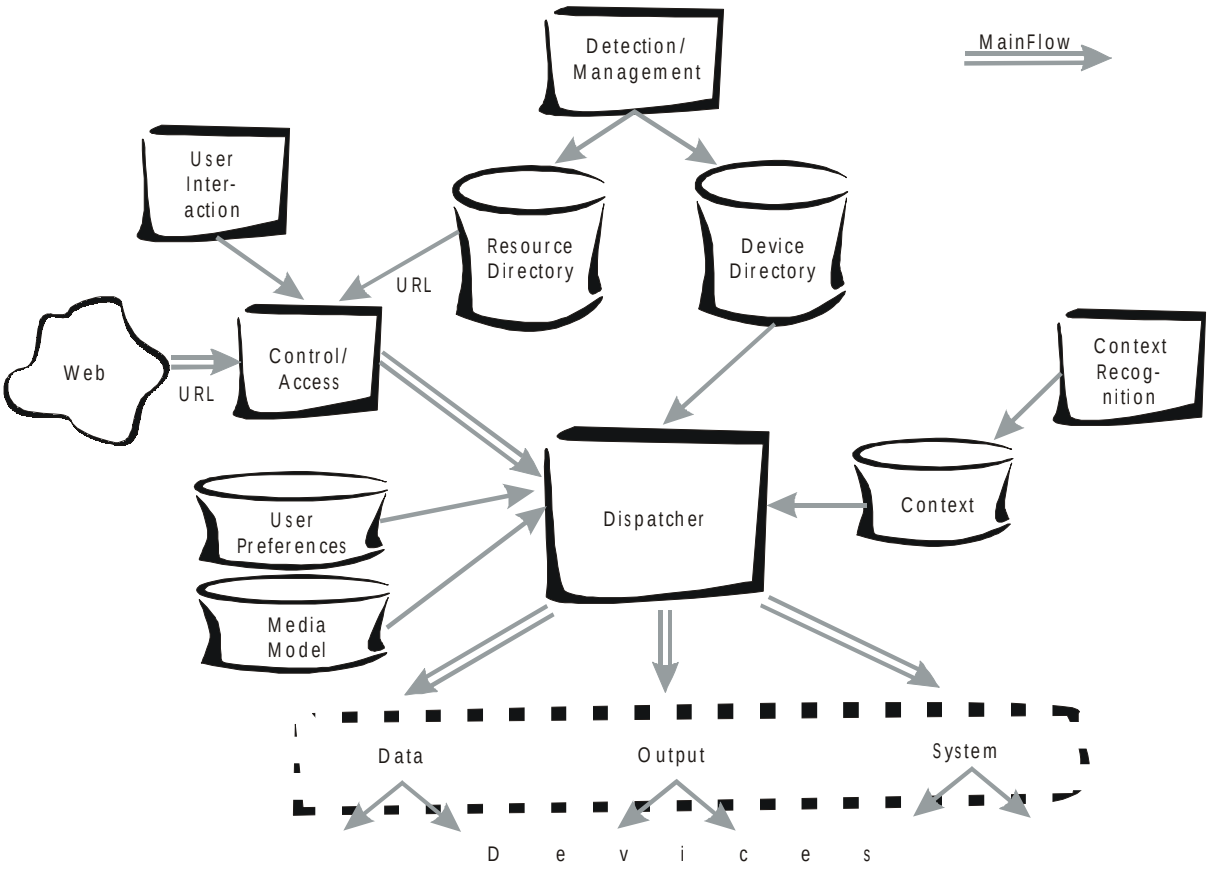

Figure 3: Architecture of the UbicompBrowser

3.1. System Components

Control/Access is the central component of the UbicompBrowser. It is the system component that interacts with the user, that provides access to both web resources and environment resources and that controls other system components, in particular the device configuration

detected in the environment. From a web point of view, this component is a web browser but it has the additional role of keeping the strings together in a user interface system that is based on multiple devices.

The Data Output System is the system component that facilitates output of resources. The system component is based on the set of output devices detected in a given environment at a given time. The device capabilities with respect to rendering or effecting of resources are specified in the Device Directory, and the environment-based resources that devices provide are kept in the Resource Directory.

Registration, detection and management of both device directory and resource directory are allocated in the Detection/Management system. This system goes beyond mere administration of devices and resources in the environment, but is also responsible for facilitating access to environment resources.

Another key component is the Dispatcher that is responsible for allocating resources to output devices. The Dispatcher contains a decision system for resource allocation that uses different models for decision-making: a media model relating resource/media characteristics to device capabilities, a model of user profiles and preferences, and a model of usage context. Beyond decision-making the dispatcher actually facilitates the redirection of resources to the devices they are allocated to.

Context Recognition is a system component for establishing situational context to be used for dispatching of resources. A fundamental context required for the UbicompBrowser is the location of the user, or actually his/her access component, relative to the environment. In our prototype system we have not yet investigated the recognition of further context information. We are interested though in recognition of situational context such as whether a user is by himself or with people he knows well or in awkward situation and so on. We are currently investigating context-awareness with a focus on these complex kinds of context within the ESPRIT IT for Mobility project TEA (Technology for Enabling Awareness, cf. http://www.omega.it/tea/) [9].

3.2. Implementation Architecture

In the implementation architecture, both Access/Control and Dispatcher are allocated to a handheld device, to facilitate ubiquitous access to resources. The UbicompBrowser system does not depend on this device to be personal, but for a number of reasons we prefer to think of it as personal companion rather than as a public remote-control-type of device. Above all, the integration of Access/Control and Dispatching with a personal device enables personalization and addresses concerns of privacy [4]. As an example consider the advantage of carrying personal web bookmarks around in a personal device, as opposed to non-availability of personal bookmarks or the alternative of required authentification.

Detection and management of environment resources are allocated to an operating system for ubiquitous computing environments, which is described in more detail in [6]. This system provides device and protocol abstractions based on which the UbicompBrowser can integrate output devices and environment services and resources. The system is based on an open architecture so that arbitrary devices can be integrated. Most notably, the system can integrate devices that are controlled over different communication media, such as infrared, power circuit, and local area network. The UbicompBrowser and in general applications based on

this system can fully abstract from how devices and resources are actually accessed. Their interface to the system are the device directory and the resource directory.

4. DETECTION AND MANAGEMENT OF ENVIRONMENT DEVICES AND RESOURCES

As described above, the detection and management of environment devices and resources is not so much a component of the UbicompBrowser but considered as operating system functionality. This functionality is key to the UbicompBrowser though and thus will be described in this section.

The Detection/Management module is responsible for monitoring the environment and updating the Environment Resources and Device Directory. If a device has the capability to register itself with the directory this process can be automated. If devices do not have this capability, consider for instance standard TV sets, then they have to be registered manually. Another option though is to tag these kinds of devices with an active badge, an electronic bar code or a similar technology to facilitate their automated detection. This option is particularly interesting for devices that are mobile within the environment, for example overhead displays used in different meeting rooms at different times.

Devices are registered in the directory with information regarding their capabilities and with specific protocol drivers if they use non-standard protocols for communication. This approach ensures openness for integration of devices, and ease of extension by simply adding another communication protocol, for instance based on a different physical technology (e.g. X10 for transmission using power circuits or IrDA using infrared).

The Device Directory stores all information about devices within the environment. A device is defined by the name of the device, a functional classification and the information on how to communicate with this device. In addition to this fundamental information the Device Directory may contain additional information depending on the device, context or implementation of the system.

The Environment Resources Directory contains the description of resources available in the environment. The resources are related to devices in the environment, and detection of resources is consequently based on asserting for each device the set of resources that are exported.

The directories contain information on devices and resources that are available somewhere in the larger environment, for instance a directory might cover a whole building or even a whole company with many facilities. For the functioning of the UbicompBrowser it is fundamental that at any time the subset of devices and resources can be determined that are ”near” the user (i.e. the handheld access/control device). To deal with the concept of proximity we use the notion of rooms in our environment model to be able to assert which devices and resources are available in the room the user is in. This requires of course that the location of the handheld access device can be determined with the required accuracy. In our prototype system, this information on locality is inferred from the infrared communication system used for network integration of handheld devices. Other solutions could be based on simple technologies for smart environments, for instance active badges [3].

5. MATCHING RESOURCES TO DEVICES: THE DISPATCHER MODULE

The Dispatcher matches resource media to devices, based on context information from the application, the environment and the user of the system. Note that only media output is dispatched while interaction is centralized in the handheld access/control device. This is somewhat in contrast to the original ubiquitous computing vision as described in [11] but has important advantages regarding personalization and respect of privacy. Still, for further work it has to be investigated in which ways and to what extent user interaction can be dispatched to devices in a given environment.

In contrast to Arens et al. [2] and other approaches described in the area of intelligent multimedia systems, the mapping of media to devices in the UbicompBrowser is not so much lead by the view ”what is good for the user” but by the more pragmatic and technical view of which output device is capable of displaying a given resource or piece of content. Also, in our current approach media are only dispatched to device that can process them directly; we do not resort to automated media conversion but of course the inclusion of such a functionality constitutes a possibility for further extension of the UbicompBrowser. For the currently prototyped system though, we concentrate on a pragmatic and robust approach to gain usage experience as early as possible.

The dispatcher is based on a rule-based decision-making. The rules are based on the following kinds of information: the environment as described in the Device Directory, the application that may contain constraints or further rules to be interpreted, the media characteristics, and possibly user preferences and further context information.

Media content could be manifold. We distinguish three different types:

- persistent web documents, like text, movies or audio clips
- transient media streams related to receiver devices, like TV or radio programs
- programs, like control sequences for devices, Java applets for handhelds and for set-top boxes

The type of an information can either be explicitly given, or implicitly detected by means of media analysis, which we would consider as typical functionality for a ubiquitous computing operating system. In the absence of such a functionality, we resort to explicitly given media types. These can be specified for example as MIME type, or through the use of identifiers in the URI, such as tv://... for accessing TV channels. If the type is not specified, an option may be to deliver the resource to possible output devices which can accept or reject them depending on whether they are able to interpret and process the content.

The approach actually taken in the UbicompBrowser system is based on the fact, that different devices have different possibilities and characteristics, and that not every output device is capable or useful for any kind of media content. This is somewhat in contrast to the PC paradigm, where one output device is build to meet every output condition. As mentioned above the classification of devices concentrates on the basic characteristics to keep the system small and simple but the UbicompBrowser allows the extension of the device characteristics in the future. Devices are characterized by their name, the device class (TV, Computer, radio, etc.), and their type. Actual types are visual, audio and other physical devices. Visual devices are classified by the size, pixel and framerate available on the device,

audio by the quality parameters spectrum and volume range. The characteristics of physical devices are diverse and depend mostly on the device capability. Because it is difficult to find rules for arbitrary devices, the rules controlling these devices must come from the application; the behavior of the UbicompBrowser is therefore depends on the specific application context.

To make decisions, the Decision Maker stores rules for the classified media content. These rules are expressed as if–then conditions e.g. ”IF text THEN viewed on visual devices” similar to the approach of [2]. In addition, we consider the use of fuzzy logic, because some of the parameters (e.g. display quality on a video display) are rather fuzzy.

Two factors influence the made decision: the context and the user. Context is an important factor when making decision about redirecting media output, for instance it is inappropriate to show private content on big public screens. In the current prototype, we have not yet addressed awareness of such rather complex contexts though. The user can influence the decisions made by the UbicompBrowser in two ways: First he can set preferences like ”IF video THEN view on TV” that are then integrated in the rule base. Secondly, he can influence the system in the ”direct interaction mode”, forcing the system to have media redirection explicitly confirmed by the user.

6. DISPATCHING OF ENVIRONMENT RESOURCES

The concept of MIME-types (RFC 1521) can be used to describe media types. Resources on the web, such as texts, html files, images, or movies are qualified by their MIME-type. This description of the media type, used for any standard web communication, is sent in the header of an HTTP-reply. Web-browsers use this information to decide how to deal with the incoming media stream, the data is either processed within the browser (e.g. HTML or GIF), or handled by additional programs such as Plug-Ins or helper applications. The main implication of this concept is that all data has to be handled by the browser. Considering handheld devices with low bandwidth and little computing power this does not seem appropriate. A further problem when using MIME-types to identify media streams is that legacy media, based on other technologies, such as television or radio, can not be include in the system.

Additional to the rule-based dispatching of arbitrary media stream we suggest a dispatching system that is identifying the media types by their URL. We extended the URL concept for further media types such as TV and radio. The following protocols are specified:

```
tv://<location>/<station>
radio://<location>/<station>
x10://<location>/<device>?<control-sequence>

<location> ::= local
<station> ::= <TV-station-by-name> | <Radio-station-by-name>
<device> ::= light | fan | heating
<control-sequence> ::= on | off | low | high
```

```
<TV-station-by-name>::= ARD | ZDF | RTL | PRO7 | SKY | MOVIE |
CNN | NBC
<Radio-station-by-name>::= SWR1 | SWR2 | SWR3 | BR3 | NEWS
```

The location "local" can be regarded as a variable that is replaced by the dispatcher with the current location of the user, which is regarded as an abstract server. It is easily feasible to include further protocols, by extending the dispatcher. The definition of each protocol is open to include further devices or media-types. In the table below examples for the extended protocol are given.

URL	Protocol/Media-type	Media Content
tv://local/ARD	Television	German TV channel ARD
radio://local/SWR3	Radio	German radio station SWR3
x10://local/light?low	House automation, control sequence	Dim the light low.

The URL provides information about the protocol (e.g. http, ftp, tv, radio) and about the resource depending on the protocol. In the case of the extended URLs the protocol gives the indication of the media type. Considering standard Web-URLs it is in general possible to extract the media type already from the filename. Using this mechanism the system knows[1] before sending a request what media type is expected in the reply. With this knowledge the request for the media stream can be delegated to the appropriated device.

Delegation of the media stream is done rather on conceptual level than on a data stream level. The dispatcher delegates the retrieval of an URL to the appropriate device, therefore the dispatcher needs to know how to communicate with the devices that have certain display and control capabilities. The communication channels are stored in the Device Directory. After matching the media and the output device the appropriate communication channel is queried from the directory. In our implementation the dispatcher is communicating via TCP/IP with all other devices. For those device that can not communicate over TCP/IP (e.g. TV, radio, light) we use a controller in the network (e.g. a house automation controller) as surrogate. For illustration, the following list presents a selection of the used communication channels:

- The dispatcher communicates with a house automation controller that sends IR-command to switch the TV or Radio to the desired station.

- The dispatcher communicates with a house automation controller that sends X10-command (power line modem commands) to dim the light.

[1] In some cases it is not possible to detect the media type for http-resource; in these cases it is possible to use the http-command HEAD to get the information (MIME-qualified) for the resource in question.

- The dispatcher communicates with a modified WWW-browser on desktop computer to remotely control the page that is displayed.

- The dispatcher communicates with a modified WWW-browser on the PDA to remotely control the page that is displayed.

7. SUMMARY AND CONCLUSIONS

The UbicompBrowser introduced in this paper merges ubiquitous computing ideas with concepts of the World-Wide Web. With the help of the UbicompBrowser a user can access global resources in the web and local resources (like TV, radio) through an environment-based user interface composed of a handheld control and dynamically detected output devices. There are two fundamental concepts of the UbicompBrowser: truly ubiquitous web access, and uniform access to web resources and environment resources.

A prototype, depicted in figure 4, was build based on the proposed architecture to demonstrate the UbicompBorowser principles, and to gain first experience with the possibilities of ubiquitous access. To enable this kind of access two methods are used in the UbicompBrowser: rule-based matching of media content to devices and an extension of the URL concept. Using the later concept we enable authors to include links that represent a certain media-content but relay on local resources, such as ”dim the light”.

Figure 4: Prototype

After a decidedly pragmatic approach to implementation of the first prototype, ongoing and further work concentrates on a range of scientific challenges. In particular the focus is currently on more elaborate models for rule-based matching of resource media to output

devices, on management of distributed input devices, and on recognition and exploitation of complex situational context. Also related to the UbicompBrowser project is the further development of an operating system for ubiquitous computing environments as referred to in some parts of this article. Last not least further work will also extend the current prototype for integration of additional devices, to further validate the concepts.

REFERENCES:

[1] Abowd, G. Ubiquitous Computing: Research Themes and Open Issues from an Application Perspective. Technical Report 24-96, Georgia Institute of Technology, Atlanta, GA, USA, 1996.

[2] Arens, Y., Hovy E.H., and Vossers, M. On the Knowledge Underlying Multimedia Presentations, Multimedia Interfaces, Mark Maybury Ed., MIT Press, 1993

[3] Beadle, P., Harper, B., Maguire, G.Q. and Judge, J. *Location Aware Mobile Computing*. Proceedings of IEEE International Conference on Telecommunications, Melbourne, Australia, April 1997.

[4] Beigl, M., Schmidt, A., Gellersen, H.W. and Lauff, M. A Concept to Move Ubiquitous Computing from the Socialistic Ideas into the Real World of Existing Capitalism, Proc. of International Workshop on Networked Appliances'98 (IWNA98), Kyoto, Japan, November 3-6, 1998

[5] Gaedke, M., Beigl, M. and Gellersen, H.W. Web Content Delivery to Heterogeneous Mobile Platforms. Proc. of Workshop on Mobile Data Access, Singapore, November 19-20, 1998

[6] Lauff, M., NA3 – Networked Appliances Automation Architecture. Proc. of International Workshop on Networked Appliances'98 (IWNA98), Kyoto, Japan, November 3-6, 1998

[7] Lauff, M., Schmidt, A. and Gellersen, H.W.. A Universal Messaging Agent Integrating Device Modalities For Individualised Mobile Communication. 13th International Symposium on Computer and Information Science ISCIS'98. Antalya, Turkey, October 26-28, 1998.

[8] Robertson, S., Warton, C., Ashworth, C. and Franzke, M. Dual Device User Interface Design: PDAs and Interactive Television, CHI 1996

[9] Schmidt, A., Beigl, M. and Gellersen, H.W. There is more to Context than Location, In Proc. of Interactive Applications of Mobile Computing (IMC), Rostock, Germany, November 24-25, 1998

[10] Spinellis, D. Palmtop Programmable Appliance Controls. In Personal Technologies, Vol. 2, No. 1, March 1998.

[11] Weiser, M. The Computer for the 21st Century. Scientific American 9/91