Talk to your Displays – Position Paper for the Workshop "Ubiquitous Display Environments"

Elmar Braun, Erwin Aitenbichler, and Max Mühlhäuser

Telecooperation Group
Department of Computer Science
Darmstadt University of Technology
Darmstadt, Germany
{elmar,erwin,max}@tk.informatik.tu-darmstadt.de

Abstract. Ubiquitous computing spaces, which have displays generously embedded into the environment, allow interaction with graphical user interfaces in a much more casual manner than desktop computers, which tie the user a particular desk. But simply putting desktop applications on a ubiquitous display will not make their use casual. We propose applications that can roam among displays as well as to personal devices (PDAs, etc.). Applications may also use a combination of public and personal devices simultaneously for interaction. We especially focus on associating displays with speech-based personal devices. Such combinations can be used to present interaction in an effortless, casual, and multimodal manner.

1 Introduction

Ubiquitous computing aims to ease the use of computers by replacing the current desktop computer paradigm, which puts the personal computer at the center of attention, with a human-centered paradigm. This can be paraphrased in two main points. Firstly, increasing power and decreasing cost and size of computers will make it possible to embed them generously in working and living spaces. Secondly, this embedding should be performed in a manner such that use of these embedded devices is effortless and casual.

Both points are valid not only for the processing part of computers, but also for input and output devices. As displays become cheaper, thinner, and larger, they can be embedded everywhere in the walls of working environments. Should the trend towards cheaper displays continue, the same may also happen for living environments. This poses the question how such embedded displays can be used in a manner that is indeed effortless and casual, rather than just putting a desktop interface on a wall. Our approach to this challenge has four steps.

The first step is supporting roaming with location awareness. A positioning system tracks the location of a user. This allows inferring which of the surrounding input and output devices is within usable range of the user. As a user moves about a space augmented with a positioning system and embedded displays, her

session can be redirected to whatever screen is most convenient for her. This is also referred to as *teleporting* [1].

Even if embedded displays become commonplace, personal devices are unlikely to disappear. This has two reasons. Firstly, large embedded displays cannot be used to display private information unless no other person is in the room. Secondly, personal devices are necessary to support users who are away from any augmented space. Our second step is to make use of these personal devices for multi-device interfaces in environments equipped with public displays. Public displays may have no convenient means of input. In such situations personal devices can be used as additional input devices. They can also be used to display and input private information, while the not privacy sensitive part of an application remains on a public display.

Most current mobile computing devices have a screen and are based on a graphical user interface. Speech interaction is an alternative, but often slower and more cumbersome, and also not as popular. However, a personal speech-based device, when used in conjunction with a public display, can present a user interface in a convenient multimodal fashion. Therefore our third step is to combine voice-based interaction with public displays for multimodal interaction. The user can walk around without obstructions, yet make input simply by speaking, and get information from a public display with so much as a glance.

This concept can also lead to smaller personal devices. The size of current mobile devices cannot be decreased much further because of the need to provide sufficient screen space. In a ubiquitous display environment, a screen can usually be associated from the environment. Therefore the personal device can be shrunk to a much smaller voice-only device, which associates screens from the environment for interaction, and offers some limited voice-only interaction as the "last resort" when no display is available. We have developed the *Talking Assistant* as a prototype of such a voice-only personal device for ubiquitous computing [2].

If users should be able to use any of a number of personal devices, ranging from voice-based to screen-based devices of different sizes (e.g. PDA or tablet PC), in conjunction with public displays, the user interface needs to adapt to whatever set of devices the user currently uses. As the number of conceivable combinations of devices is rather large, manually authoring a special interface adapted to each feasible combination would be too much effort. Therefore, in our fourth step, we use *single authoring* techniques to author user interfaces for such settings. Single authoring means that different user interfaces for different device types are generated automatically from one device independent source, so that porting an application to a different device type does require a complete rewrite of the user interface.

Applied to embedded displays that are used in conjunction with various personal devices, this concept has two differences compared to traditional single authoring. Firstly, the target device of the transcoding from device independent to concrete user interface targets not one device at a time, but a set of devices (a ubiquitous display and any available personal device) as a "virtual target

device". Secondly, this virtual target device can change at runtime when the user picks up or drops a personal device, or walks from one public display to another one with different properties, while an application is running.

2 Architecture and Implementation

We have built the runtime infrastructure necessary to render applications distributed across the devices of a ubiquitous computing environment. Such applications consist of an XML user interface description and a Java class, and run on a server called *dialog manager*.

This server first collects the information about the devices available in the environment of the user. Then it decides where which parts of the user interface are to be displayed. Thereupon it connects to a client on each involved device, which displays its respective part of the interface. Every interaction is sent back to the dialog manager, where it is processed by the application and relayed to all other clients.

At the conceptual level, this is a model-view-controller (MVC) pattern with multiple views and controllers that run concurrently distributed on various devices in the environment near the user.

The following sections explain our runtime infrastructure in more detail. Much of this is also explained in [3–5].

2.1 Discovery

In order to dynamically adapt to whatever set of devices is currently the most convenient for a user, it first needs to be determined what that set of devices is. We have developed two different methods for detecting user location and association of devices.

The first scheme uses badges akin to the Active Badge [6] location system: badges worn by each user repeatedly broadcast their identifier by infrared. Receivers in each room receive these broadcasts, and thereby determine in which room a user is. This information is then relayed to the network. We have extended this concept with tags, which use the same hardware as badges. Tags also broadcast an identifier, but use a different, less robust encoding, which can only be received within roughly one meter direct line of sight. Badges receive these broadcasts, and relay them through the room receiver to the network. Such events determine not only the room-scale location of a user, but also that she is within one meter direct line of sight of a screen equipped with a tag, and that she is facing that screen.

The second scheme is called IRIS [7]. IRIS also uses infrared broadcasts coming from a transmitter on the Talking Assistant headset. Each room has two infrared cameras mounted in parallel to provide a stereo picture. The three-dimensional position of the transmitter can be calculated from the two pictures. In addition, a compass integrated in the headset determines the bearing of the head. Combined with a world model (a database of the location of all devices

of interest in a room), head location and direction is used to determine which device a user is currently looking at.

2.2 Authoring and Transcoding

Our single authoring language is loosely based on XForms [8] and XHTML [9]. Our additions are mostly hints, which instruct the transcoder how elements of the user interface are to be handled if more than one device is available to render them.

As soon as the set of devices available to the user has been determined, the dialog manager starts to "split" the user interface in pieces for each of the involved devices. This splitting neither means that each part of the user interface is rendered exactly once, nor does multi-device rendering mean that each widget is replicated on each device. The splitting is done according to a number of patterns. The "dominating device", which is the large main screen in a setting with a ubiquitous display, will render most of the widgets, except for privacy sensitive widgets of course. For the speech device, we have a "multimodality pattern", which replicates widgets even if already rendered on another device. This redundancy is beneficial because of the added modality. Small handheld devices, which have the benefit of being right in the user's hand and therefore quick to access, render only the most important widgets according to a "remote control pattern". They also render widgets marked as private, if no other device can do this without compromising privacy. We are currently investigating which additional patterns could be useful in such a multi-device setting.

After the interface has been split in fragments for all the involved devices, these fragments are transcoded to device specific representations fitting the involved devices, as detailed in the next section. Whenever the user roams to a different set of devices, the splitting process is initiated again.

2.3 Clients

Currently we have four different clients for different device types. The first client to be developed was our normal GUI client. It is based on the Internet Explorer. When the dialog manager decides to display something on a display near the user, it transcodes the user interface to an appropriate HTML form, and instructs the GUI client to display this page. Whenever the user makes an input (clicks a button, presses a key to change the value of a text field, etc.) the client sends this immediately back to the dialog manager, which notifies the application and instructs all other clients to update their view. In other words, although we use a HTML form for the graphical display, we do not send user input to the server only when the user presses the submit button, but notify the application of each event in the user interface instead, much like event-based widget toolkits do within one computer. We plan to replace this client with a similar client based on the Mozilla web browser so that we are not forced to use Windows on all displays in our experimental ubiquitous computing room.

Our second client is a speech client. It is based on the Java speech API (JSAPI). Using a bridge from the JSAPI to the Microsoft Speech API (SAPI), we can use any speech recognizer that implements either the JSAPI or the SAPI. The transcoder transcodes the interface to a Java Speech Grammar Format (JSGF) grammar, which is then processed by the underlying speech recognizer. Similar to the GUI client, the speech client sends each input made by speech immediately to the dialog manager.

We have developed two additional graphical clients for different handheld devices. As a PDA client we have developed a Java program using Thinlet[1], a GUI library that uses XML for user interface descriptions and runs on a Java virtual machine for Microsoft Pocket PC PDAs. For cell phones we have developed a client that runs on the Java Mobile Information Device Profile (MIDP) 2.0 which is supported by newer cell phones, and connects to the dialog manager via Bluetooth.

3 Related Work

Somewhat similar to our Talking Assistant is the Personal Server [10], a completely UI-less personal device that exclusively relies on associated ubiquitous devices for interaction. Our concept is motivated similarly, except that we have added speech interaction, which should not increase size by much.

Applications that follow roaming users across devices have been proposed by Bennet, Richardson and Harter [1]. This work lets an exact graphical copy of her desktop follow the user from desktop computer to desktop computer. We extend this by teleporting an application described in a device independent manner, so that we can teleport between different types of devices and even modalities. There have been attempts at device independent roaming [11], which did not take multi-device scenarios in account.

The notion of using several devices concurrently has been explored before (e.g. [12, 13]), but usually for a single fixed set of devices, with no regard for single authoring.

Connecting several devices together to render a user interface has been attempted before. Some have attempted this using different web browsers on different devices [14], which looks relatively similar to our approach, since we use a web browser as large display client too. However, this approach again does not regard the problem of single authoring. Others have tried a similar approach for the rendering of multimedia, by rendering different channels of a multimedia presentation (e.g. audio, video) on different devices which are assembled from the environment [15].

[1] `http://www.thinlet.com/`

References

1. Bennett, F., Richardson, T., Harter, A.: Teleporting - Making Applications Mobile. In: Proceedings of 1994 Workshop on Mobile Computing Systems and Applications, Santa Cruz, USA (1994)
2. Aitenbichler, E., Mühlhäuser, M.: The Talking Assistant Headset: A Novel Terminal for Ubiquitous Computing. Technical Report TK-02/02, Fachbereich Informatik, TU Darmstadt (2002)
3. Braun, E., Hartl, A., Kangasharju, J., Mühlhäuser, M.: Single Authoring for Multi-Device Interfaces. In: Adjunct Proceedings of the 8th ERCIM Workshop "User Interfaces For All". (2004) `http://www.ui4all.gr/workshop2004/publications/adjunct-proceedings.html`.
4. Braun, E., Mühlhäuser, M.: Extending XML UIDLs for Multi-Device Scenarios. In Luyten, K., Abrams, M., Vanderdonckt, J., Limbourg, Q., eds.: Proceedings of the AVI2004 Workshop "Developing User Interfaces with XML: Advances on User Interface Description Languages". (2004) 143–149
5. Braun, E., Austaller, G., Kangasharju, J., Mühlhäuser, M.: Accessing Web Applications with Multiple Context-Aware Devices. In: Proceedings of the ICWE2004 Workshop "Device Independent Web Engineering (DIWE04)". (2004)
6. Want, R., Hopper, A., Falcão, V., Gibbons, J.: The Active Badge Location System. ACM Transactions on Information Systems (TOIS) **10** (1992) 91–102
7. Aitenbichler, E., Mühlhäuser, M.: An IR Local Positioning System for Smart Items and Devices. In: Proceedings of the 23rd IEEE International Conference on Distributed Computing Systems Workshops, 3rd International Workshop on Smart Appliances and Wearable Computing (IWSAWC03), Providence, USA (2003) 334–339
8. Dubinko, M., Klotzs, L.L., Raman, T.V.: XForms 1.0. `http://www.w3.org/TR/2003/REC-xforms-20031014/` (2003)
9. Axelsson, J., Epperson, B., Ishikawa, M., McCarron, S., Navarro, A., Pemberton, S.: XHTML 2.0. `http://www.w3.org/TR/2003/WD-xhtml2-20030506` (2003)
10. Want, R., Pering, T., Danneels, G., Kumar, M., Sundar, M., Light, J.: The Personal Server: Changing the Way We Think about Ubiquitous Computing. Lecture Notes in Computer Science **2498** (2002) 194–209
11. Chu, H., Song, H., Wong, C., Kurakake, S., Katagiri, M.: Roam, a seamless application framework. Journal of Systems and Software **69** (2004) 209–226
12. Rekimoto, J.: A Multiple Device Approach for Supporting Whiteboard-Based Interactions. In: Proceedings of ACM CHI 98 Conference on Human Factors in Computing Systems. (1998) 344–351
13. Myers, B.A.: Using Handhelds and PCs Together. Commun. ACM **44** (2001) 34–41
14. Kleindienst, J., Seredi, L., Kapanen, P., Bergman, J.: Loosely-coupled approach towards multi-modal browsing. Universal Access in the Information Society **2** (2003) 173–188
15. Pham, T.L., Schneider, G., Goose, S.: A Situated Computing Framework for Mobile and Ubiquitous Multimedia Access Using Small Screen and Composite Devices. In: Proceedings of the 8th ACM international conference on Multimedia, Marina del Rey, USA, ACM Press (2000) 323–331