



UbiComp 2001 Informal Companion Proceedings

**Editors: Gregory D. Abowd
& Anind K. Dey**

**September 30 – October 2, 2001
Sheraton Colony Square Hotel
Atlanta, GA, USA**

Technical Report: GIT-GVU-TR-01-17

Table of Contents

Blow off your window screens: Towards ubiquitous inhaling and exhaling interaction	1
Soichiro Iga	
Pushpin computing	3
Josh Lifton	
The Gambit	5
Kati Rubinyi, Ewan Branda & Zhenya Gershman	
The Meeting Pot: Coffee aroma transmitter	7
Itiro Siio & Noyuri Mima	
On-demand, in-place help for mixed reality environments	9
Desney Tan, Ivan Poupyrev, Mark Billinghurst, Hirokazu Kato, Holger Regenbrecht & Nobuji Tetsutani	
Ubi-Finger: Gesture input device for mobile use	11
Koji Tsukada & Michiaki Yasumura	
Lessons learned in creating real world interfaces	13
David C. Wrighton, Dillon T. Bussert & D. Scott McCrickard	
Bringing application functionality to small devices: W@Pnotes	15
Andreas Zeidler	

Blow Off Your Window Screens: Toward Ubiquitous Inhaling and Exhaling Interaction

Soichiro Iga

Keio University, Keio Research Institute at SFC
5322 Endo, Fujisawa, Kanagawa, 252-8520 Japan
igaiga@sfc.keio.ac.jp
<http://www.chi.mag.keio.ac.jp/>

1 Kirifuki System

We developed a novel system called “*Kirifuki*” which has an ability of operating computer system by breathing in and blowing out. We also propose a method to manipulate visual objects such as windows, icons, menus which is suitable for inhalation/exhalation interaction style.

Kirifuki system consists of a workstation (SGI Indigo2, MIPS R4400, 192MB memory), a liquid crystal projector (Epson ELP-5300), a magnetic gyroscope (Polhemus sensor), and a breath microphone switch. Software part is implemented in C and Iris GL.

The computer screen is projected onto the physical desk by the liquid crystal projector. The user wears a head set device with a small breath microphone switch and a polhemus sensor (Fig.1). The polhemus sensor determines where the user is looking at. A small microphone detects the sound of the breathing and discriminates between inhaling and exhaling by means of the audio signals of the sound.

The user's head position is mapped to the manipulation of the pointing device, and the inhalation/exhalation will be the trigger to invoke the action of the graphical objects.

2 Application - Kirifuki Window

Kirifuki system can be applied to the manipulation of graphical user interface (GUI) objects. The example here demonstrates the operation of GUI objects such as moving and resizing window objects and cutting and pasting icon objects.

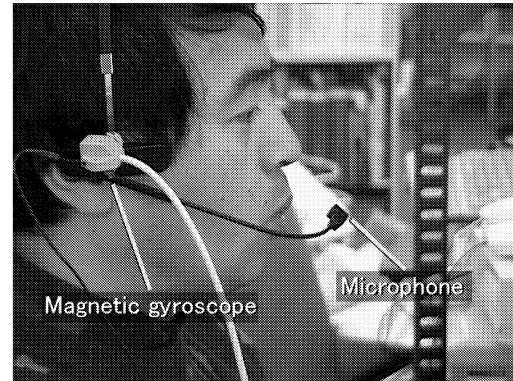
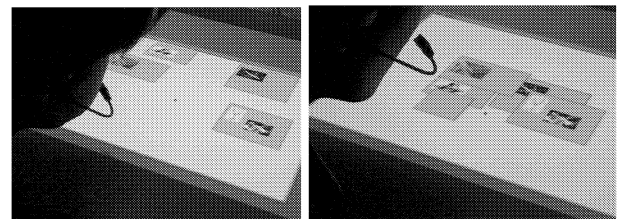


Fig. 1. Detecting inhalation and exhalation by breathe microphone switch and head movement by magnetic gyroscope: The breath microphone switch is configured to point the user's lips. The magnetic gyroscope is located in the side of the user's head.

2.1 Geometric Operation of Windows

When the user shift a mouse pointer in the root window, he or she can gather and distribute window objects just by inhale and exhale actions (Fig.2). When the user exhales to the root window, the window objects near the mouse pointer would be diffused. On the contrary, the windows would be assembled when the user inhales.



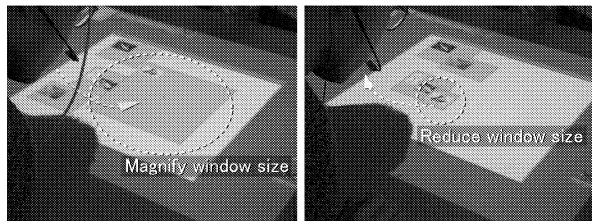
(a) Diffusing

(b) Assembling

Fig. 2. Manipulation of the movement of the window object: Blowing windows away by exhalation and gathering windows by inhalation.

2.2 Resizing Operation of Windows

When the user moves a mouse pointer into a particular window, he or she can zoom in or out its size (Fig.3). When the user exhales to a particular window, the window would be magnified. Conversely, inhalation would reduce the window size.



(a) Zooming-in

(b) Zooming-out

Fig. 3. Manipulation of the size of the window objects: Zooming in the selected window size by exhalation and zooming out it by inhalation.

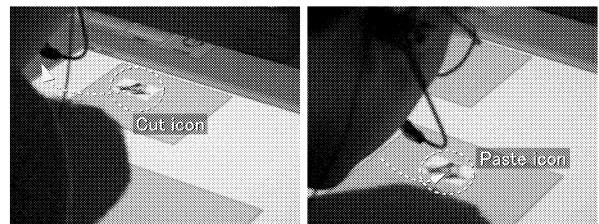
2.3 Cut and Paste Operation of Icons

The user can cut and paste icon objects by inhalation and exhalation (Fig.4). When the user points a particular icon object by a mouse pointer and inhales to a breath microphone switch, the icon object would be stored in the computer memory which can be compared to “cut” action. When the user exhales to a particular window, the inhaled icon would be pasted in the window.

The user also can activate a computer application which is linked to a particular icon by exhaling to the icon.

3 Related Work and Discussion

Pick and Drop interaction facilitates the information sharing among the multiple computing devices[1]. Our system expands the notion of cut and paste by the physical actions of the human behavior.



(a) Cutting Icon

(b) Pasting Icon

Fig. 4. Cut and paste operation for the icon objects: Cutting selected icon object by inhalation and pasting it by exhalation.

In our current prototype system, the user has to wear magnetic gyroscope on his or her head. However, the user who has some handicap in upper half of the body would have some difficulty in operation pointing device. In the future system enhancement, techniques like gaze input[2] can extend the limits of our current pointing device manipulation.

4 Summary

We proposed the novel system called “*Kirifuki*” which has an ability of operating computer system by breathing in and blowing out. We applied this technique for manipulating graphical user interface such as a window screen, menus, icons.

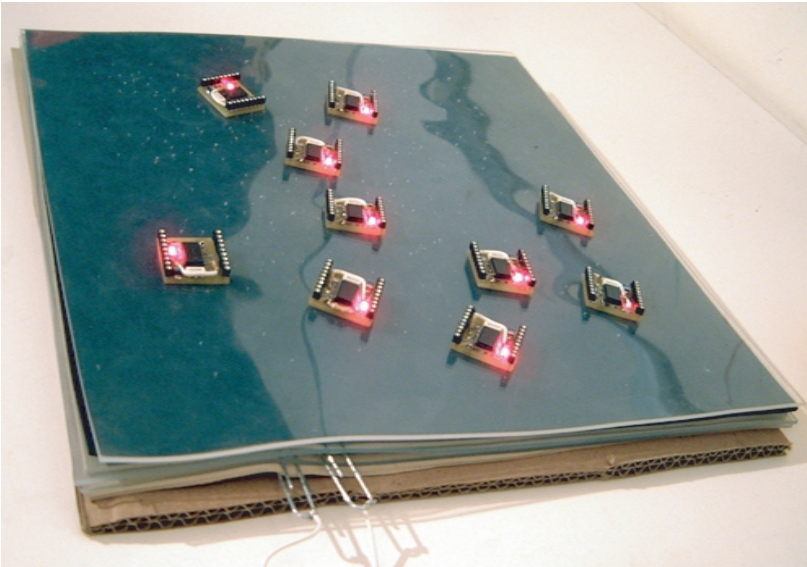
Our future research includes applying this technique in a wireless and ubiquitous environment.

References

1. J. Rekimoto, A Multiple Device Approach for Supporting Whiteboard-based Interactions, In *Proc. of CHI'98*, pp.344-351, 1998.
2. S. Zhai, C. Morimoto, S. Ihde, Manual And Gaze Input Cascaded(MAGIC) Pointing, In *Proc. of CHI'99*, pp.246-253, 1999.

UBICOMP2001 Pushpin Computing

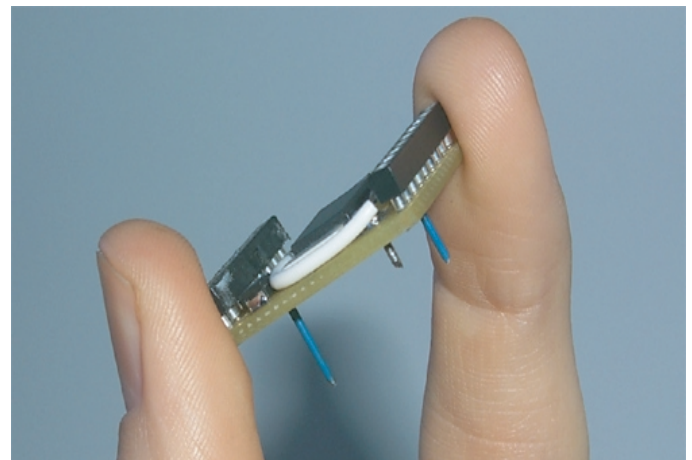
sensate surfaces - paintable computing - emergent systems



Josh Lifton
lifton@media.mit.edu
Responsive Environments Group
MIT Media Lab

PURPOSE

Pushpin Computing is the first attempt at a computing architecture closely following the ideas set forth in the Paintable Computing project. Namely, it is an architecture consisting of many independent, locally communicating computing nodes physically distributed at scales considered dense in comparison with normal human scales. Key to this model is the idea that simple local interaction among computing nodes can result in complex algorithmic structures at the level of the system as a whole. Strong analogies exist between Pushpin Computing and biological systems such as ant colonies, thermodynamic systems such as ideal gases, and abstract systems such as the game Go.



M
I
T

M
e
d
i
a

L
a
b


c

2
0
0
1


HARDWARE OVERVIEW

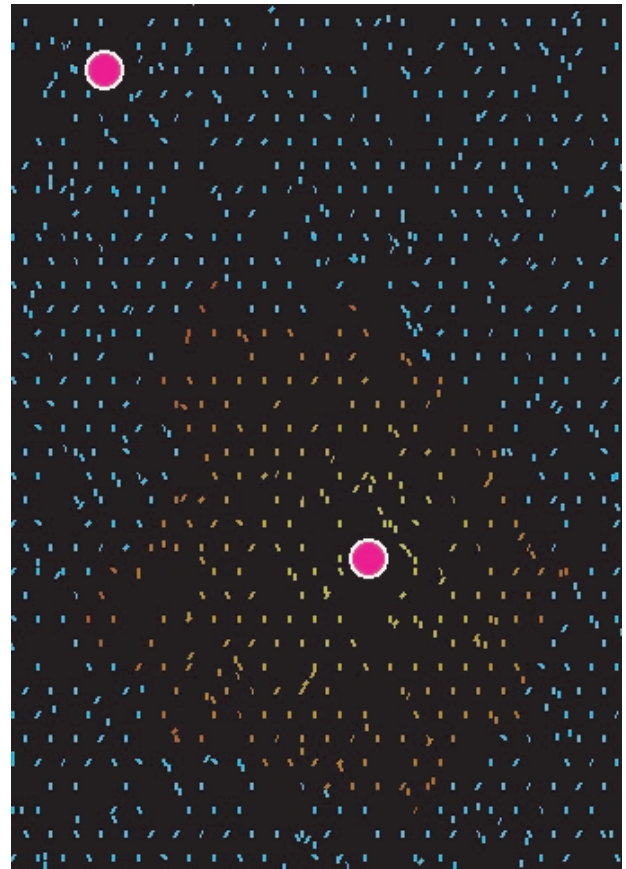
The Pushpin Computing hardware platform consists of computational elements (pushpins) that can be freely arranged and embedded in a layered silicon substrate. Aside from serving as a physical medium for the pushpins, the substrate provides power as well. Once embedded in the substrate, each pushpin is able to communicate with other pushpins within a fixed radius. Pushpins are optionally equipped with various sensing modules, from microphones to stress gauges.

SOFTWARE OVERVIEW

At the software level, Pushpin Computing is governed by a custom operating system designed to seamlessly support mobile code fragments. These code fragments are the atomic algorithmic elements that form the basis of the Paintable Computing model of computation. For example, the gradient code fragment simply copies and transports itself from pushpin to pushpin, all the while keeping track of how many jumps it has made, thus building up a distance gradient that can be retraced to the pushpin of origin. 

PROTOTYPE SPECIFICATION

- Processor:
 - Cygnal C8051F016 8051 core
 - (2304 bytes RAM, 32Kbytes ISP flash,
 - ~22MIPS @ 22MHz, JTAG interface)
 - Communication:
 - capacitive coupling
 - (~50-100kbps over ~10cm)
 - Sensing:
 - configurable expansion port
 - (8 ADC lines, 9 port I/O lines, JTAG interface)
 - Power:
 - 3.3 VDC, ~15mA/pushpin supplied through substrate
- 



REFERENCES

Pushpin Computing: <http://www.media.mit.edu/resenv/PushPin/>
Paintable Computing: <http://www.media.mit.edu/~bill/PaintableComputing/>

The Gambit

Kati Rubinyi
Art Center College of Design
Graduate Fine Arts Department
Pasadena, California, USA
rubinyi@earthlink.net
<http://datsun.net/kati/gambit/index.html>

Project Partners:

Ewan Branda (Programmer)
Zhenya Gershman (Actor)

The artwork The Gambit is an interactive animation in the form of a site-specific performance/installation to take place in the lobby of the Westin-Bonaventure Hotel, Los Angeles (John Portman, 1977).



The medium of the project is a digital compass attached to a PDA with a 2.25" by 3.33" color screen and headphones. The apparatus lets you "see through walls" to spaces outside the hotel that are around you but not directly visible to you. This "seeing through the walls" has a shape, and tells a story. The story is told through an animation made up of a pattern of still photos, words and sounds. The pattern is ordered by a spiral inscribed on the map of the city. The story is of a hotel flower arranger and a world in which flower arrangements have archival and narrative abilities.



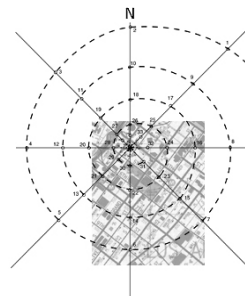
Casio E-105, headphones and digital compass.

Project Description

Introduction

The viewer of the piece stands in the lobby of the hotel with the PDA in her hand, the attached compass strapped to her waist, and headphones on her ears. As she begins to walk around (with no instructions of where in the lobby to go) she sees changing still-images on the screen and hears spoken words and sound through the headphones. Through the computer (the PDA), the changing compass coordinates induce changes in the sound and images.

What the viewer doesn't know is that there is a spiral pattern inscribed in the memory of the computer that mirrors a spiral pattern inscribed on a map of the area surrounding the hotel.



Spiral pattern overlaid on map of city, with hotel lobby as center.

As the viewer moves around the lobby, and turns on her own axis, the computer (via the compass) makes a correspondence between her motions and the spiral on the map. Images and text are being accessed in accordance with her movements inside the lobby. Through her exploration of how the device responds depending on her speed and direction of travel, a narrative emerges.

Nodes and cinematic framing

When the device is first switched on, the screen shows the view from the circular, glazed revolving restaurant 35 stories above the windowless lobby, as if the viewer had a periscope to see through the floors of the hotel and out over the city. In the narrative this constitutes an establishing shot. As the viewer walks around, the panorama of the city changes, according to which way she is facing. The "real" view evolves into fiction as the viewer completes one rotation of the spiral: In other words, when the compass registers a compass coordinate approached from the same direction for the second time. After the first rotation, the images are no longer from above looking over the city. They are at street level, and introduce a character.

The photos of the female character are taken at *nodal points* along a spiral path wrapping around the hotel. The spiral, arbitrarily drawn on a map with the hotel as its center, begins about 4 kilometres away from the hotel and winds its way through the city looping closer and closer to the hotel, and ultimately, inside the lobby. There are approximately 70 *nodes* on the spiral corresponding to the intersection of the spiral with the eight compass axes of N, NW, W, SW, etc. Each node becomes an arbitrarily selected location for a photo. The photos are taken along one of the eight axes in the direction looking *away* from the hotel. The nodes randomly occur on residential streets, commercial streets, in low-income areas adjacent to downtown and in the business district. They cut a looping section through the downtown area, delineating the

topography, literally in terms of hills and valleys, and urbanistically in terms of function and economics.

Getting back to the character positioned in each photo: With every node along the spiral path, she moves one step closer, rotates a quarter turn, and occupies a different quadrant of the frame. In other words, if the viewer were to spin around and trigger a quick succession of changes in compass locations, she would see an animation of the character rotating and weaving back and forth across the frame, and coming closer, occluding more and more of her context. Consequently, the farthest point on the spiral is a long shot and the closest point on the spiral is an extreme close-up. Occasionally the character bends and crouches, looks at the ground. Her movements are somewhat peculiar and robotic. (The actor's instructions were to look for something).



Photos from nodes 51-54.

There is a spoken word fragment tied to each node/photo. The spoken word fragment that takes the longest amount of time to hear, of a length of one or two sentences, occurs at the first node on the spiral. The briefest spoken word segments - a word and even less, corresponds to the last nodes of the spiral, inside the hotel.

Story

The story is in the form a memoir. It is about the protagonist down on her luck getting a job in the human resources department of the hotel. Her task is to look for candidates to fill the role of hotel flower arranger. But the flower arrangements in question have unusual powers of recording and transmitting stories. The stories they tell are of the relationships between the hotel guests. There is some description of the flower arrangements and how they function as "archives" and transmitters of fiction. In the end, the protagonist ends up taking on the job of flower arranger herself.

Technical notes

The PDA is a Casio Cassiopeia E-105 Windows CE with 32 MB of RAM and a 16-bit color screen. The compass is a C100 digital compass sensor from KVH Industries. The control application is written in Java and deployed on the device using the PersonalJava platform from Sun Microsystems. At runtime the control application polls the compass for updated bearing information 10 times per second. If the compass bearing has changed then the image and audio playback states are updated. Communication with the compass is done using the SerialIO Java serial port class library. A native WinCE/MIPS class controls audio playback because PersonalJava does not support high-resolution audio. Both PersonalJava and Waba were evaluated and PersonalJava was selected because of its support for multithreading, required by the MVC patterns employed in the application, its support for 16-bit images, and

the fact that the high memory capacity of the Windows CE device made the virtual machine footprint less of an issue.

User evaluation, comments and future work

The Gambit will be advertised to the art community and available to be seen by appointment. It is the author's intention to position The Gambit in an art context.

In experiencing the piece, the story becomes quite fragmented and would likely only be appreciated by an audience with a high tolerance and familiarity with fragmented narrative. Viewers who have seen it so far have appreciated the story and photos and have determined that the spiral ordering system is, appropriately, not recuperated, but rather served as a device for the composition of the photographs and audio elements.

A second chapter, a continuation of the story, is planned for 2002. This project will use the same or similar apparatus, (but preferably more in number), and will take place on a site in New York. The intention is that new chapters on other sites, in other cities, can be added in the future, the apparatus modified as new technology becomes available.

References

Film: Chris Marker, *La Jetté* (1963).

Video: Gary Hill, *Around and About* (1980).

Dance: *Chair/Pillow* Yvonne Rainer (1970); *Trio A Pressured #3* (1966) Yvonne Rainer *Satsfyn Lover* Steve Paxton (1967)

As performed by Rainer, White Oak Dance Project and members of community Thursday June 7, 2001 at BAM, Brooklyn, NY

Books: Jacob, Mary Jane *Gordon Matta-Clark*, Museum of Contemporary Art, Chicago, May 8-August 18, Chicago, 1985.

Sandford, M. ed. *Happenings and Other Acts* Routledge, London, 1985.

Elliot, James *The Perfect Thought Works by James Lee Byars* University art Museum, Berkeley, 1990

The Meeting Pot: Coffee Aroma Transmitter

Itiro Siio¹ and Noyuri Mima²

¹ GVV Center, Georgia Institute of Technology,
801 Atlantic Drive, Atlanta GA 30332-0280, USA siio@acm.org

² Department of Media Architecture, Future University-Hakodate,
116-2, Nakanochō, Kameda, Hakodate, Hokkaido, 041-8655, Japan noyuri@fun.ac.jp

1 Meeting Pot

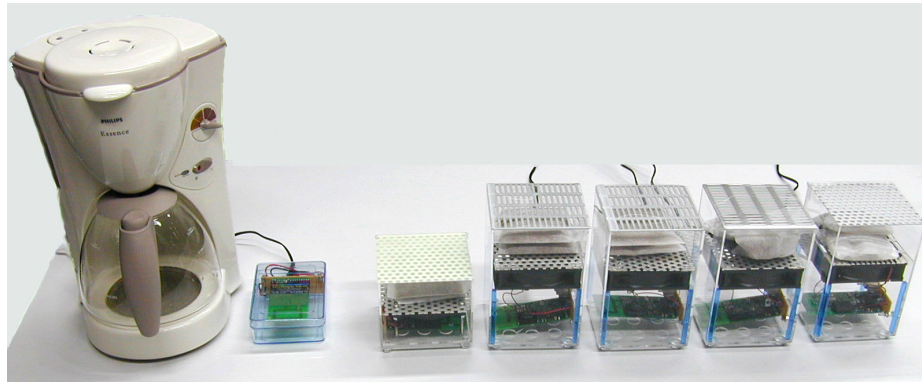


Fig. 1. The Meeting Pot system, with a coffee maker and 5 coffee aroma generators.

In a modern networked office, we have an issue as to how to create a community. To encourage informal communication at the coffee break space in an office, we have implemented a Meeting Pot system. This system informs colleagues in their office that people are meeting in the open space for a coffee break, by transmitting coffee aroma to remote locations.

Figure 1 shows the Meeting Pot system, which consists of a coffee maker and 5 coffee aroma generators. The coffee maker has been remodeled to transmit radio frequency (RF) signals when the heater is turned on to prepare coffee. As the system detects human activities indirectly through activities of the coffee maker, we can alleviate privacy concerns.

Though the information that people are meeting for a coffee break is important to activate the informal communication, it is of a comparatively low priority in daily office work. We have decided to display the information of the coffee maker in an ambient manner, that is, by the smell of coffee. The coffee

aroma generator contains a fan, an RF receiver and instant coffee powder. The fan is activated, when an RF signal from the coffee maker is received, and it emits the smell of the coffee from freeze-dried instant coffee powder installed above.

2 Feasibility Test

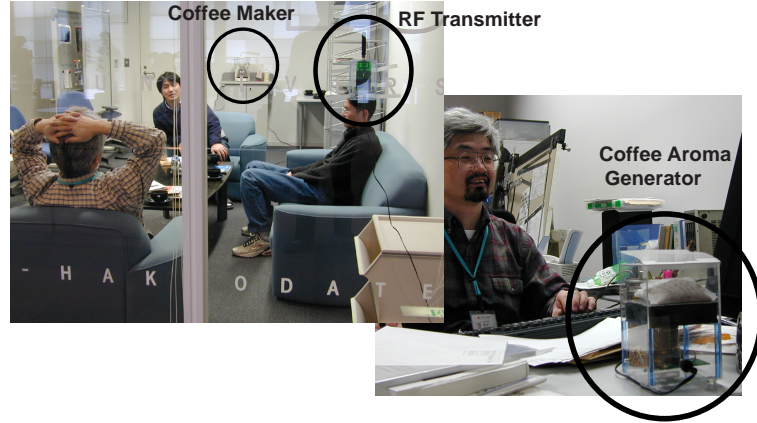


Fig. 2. The common room where the coffee maker is installed (left), and the aroma generator installed in an office (right).

To test the feasibility of the Meeting Pot system, we have installed the coffee maker in the common room, and 5 coffee aroma generators in the faculty offices of the Future University-Hakodate, as shown in Figure 2. The signal from the coffee maker is also received by a server computer, which sends electronic mail.

We have selected 10 subjects from faculty staff, and they are divided into two groups of 5. One of the groups has the aroma generator installed in their office, and the rest of them receive electronic mail that say coffee will be ready in a few minutes. We have swapped their notification method at the middle of the feasibility test period of 16 weeks.

We interviewed them and many of the subjects answered that the system encourages their informal communication very naturally. Most of them preferred coffee aroma generators, however, some of them like e-mail too. Although a few subjects want to know who is in the room, they do not want their behavior to be detected in the room.

Sometimes, they could not go out of the office to join colleagues because they were busy with their job. Even at that time, many of them relaxed by feeling an awareness of colleagues and sometimes they took a short break by themselves. We can apply the system to provide comfortable awareness of office colleagues or family members in the same or remote locations.

On-demand, In-place Help for Mixed Reality Environments

Desney Tan^{1,6}, Ivan Poupyrev², Mark Billinghurst³, Hirokazu Kato⁴, Holger Regenbrecht⁵, Nobuji Tetsutani⁶

¹ School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
desney@cs.cmu.edu

² Interaction Lab, Sony CSL
³ HITLab, University of Washington
⁴ Hiroshima City University

⁵ DaimlerChrysler AG, Research and Technology
⁶ MIC Labs, ATR

ABSTRACT

In many help systems, users are either distracted with a constant barrage of help or have to stop working on the task at hand and explicitly search for help. In this paper, we propose intuitive methods to present on-demand, in-place help in mixed reality interfaces. In these interfaces, users interact with virtual objects that are superimposed on the real world by manipulating physical cards. We describe *Tiles*, a prototype application for designing aircraft instrument panels, from which our work on help systems grew. In *Tiles*, *Tangible Bubble Help* is used, in which users manipulate special ‘help’ cards in combination with data cards to invoke detailed help. *Tangible Bubble Help* may be multi-modal, taking the form of text, audio, graphics, and animations. We also present *Tangible Tooltips*, a lightweight technique in which users control the amount of textual help by tilting data cards to different degrees. In both cases, users can seamlessly transition between performing the main task and acquiring help.

INTRODUCTION

Augmented or Mixed Reality (AR or MR) integrates computer-generated virtual information into the real physical environment [1]. Although several compelling MR systems have been demonstrated, many serve merely as information browsers, allowing users to see or hear virtual data embedded in the physical world. There has been little work done on designing effective interaction techniques, so systems generally provide few tools for the user to request or interact with the information [6].

Help systems have been one of the corner stones in desktop graphical user interfaces (GUIs) and are heavily relied upon by users. Granda et al. found that on-line help was the second most frequent source of guidance for users (asking other users was the first) [3]. Designing on-line help facilities is however, not a straightforward task. It has been shown that poorly designed help systems can actually decrease learnability and usability for novice users [4].

In designing our help system, we consider three main issues: when to provide help, as well as where and how. We challenge the prevailing assumption that users in MR environments want all virtual information visible all the time. In fact, our early experiments with MR interfaces [2] have convinced us that virtual data that is always visible can be distracting. In addition, help should be context sensitive, providing several levels of detail to lead the user through difficult tasks. Therefore in our MR work, we require that the help provided should be *on-demand* and *in-place*, i.e. help should be pro-

vided only by user request and without requiring the user to shift focus from the main task.

This paper is the first step toward designing effective interaction techniques for help in MR environments. We present two techniques – *Tangible Bubble Help* for detailed multimedia assistance and *Tangible Tooltips* for short textual reminders.

APPLICATION PLATFORM: TILES

Although our help techniques are broadly applicable, we implement them within our *Tiles* system, built for rapid prototyping and evaluation of aircraft instrument panels [5]. *Tiles* is a joint research initiative carried out with support from DaimlerChrysler AG and DASA/EADS Airbus.

In our interface, we allow users to quickly layout and rearrange a set of virtual aircraft instruments on a board simulating an airplane cockpit. Users each wear a lightweight HMD with an attached camera. They interact with virtual objects (aircraft instruments) by manipulating physical data cards marked with square tracking patterns (Figure 1a). Our computer vision system identifies these patterns in the video stream and determines their 3D positions and orientations relative to the head-mounted camera. Virtual objects are then rendered on top of the physical cards. Using various interaction techniques, users are able to easily and quickly evaluate the layout, rearrange instruments as necessary, add new instruments or remove those that are not needed.

TANGIBLE BUBBLE HELP

In tangible bubble-help, users are provided with dedicated ‘help’ cards designated with both distinct physical 2D tracking patterns and 3D virtual help widgets. To receive help on any other virtual object, users simply place the help card next to the data card on which they require help. In simplest case, this triggers explanatory text that appears within a bubble next to the icon (Figure 1b).

Initially this function was used for the instrument designer to leave short annotations on instruments or to provide help on control cards of the *Tiles* interface, e.g. the trashcan/delete card. However, we found multimedia help, such as audio annotations or 3D animations, to be particularly effective. We designed several help cards that bring up different kinds of help with different levels of detail. While one help card brings up textual annotations, another animates the instruments. A third card provides users with a 3D arrow that allows them to probe at different parts of the virtual or real objects in question, bringing up even more detailed help on



Figure 1: (a) Physical manipulation of help and data cards; (b) User view: Moving help card beside data card to acquire Tangible Bubble Help; (c) User view: Tilting physical card to attain Tangible Tooltips on virtual instrument.

component pieces. Although we have mainly used the bubble help techniques for providing help on virtual objects, it can just as easily be used for physical objects. For example, a real aircraft part that is marked with tracking patterns can be augmented with different levels of user help using the tangible bubble help technique.

TANGIBLE TOOLTIPS

Users do not always need extensive help. In some cases, all they need are short reminders, or tips, on the functionality of particular interface controllers. Help systems for this purpose have been implemented in other interfaces. In GUIs, for instance, briefly placing the cursor over a region may bring up a Tooltip – a concise description of the interface control. Such help techniques must blend seamlessly into the working process without requiring the use of special purpose tools.

Using the Tangible Tooltips technique, the user triggers the display of short descriptive phrases associated with each virtual object by bringing corresponding data cards into their working space and tilting them more than 30 degrees away from the body (Figure 1c). Other researchers have shown that tilting may be used as an interface parameter [7]. In the Tiles application this was typically used to display the name of the instrument, perhaps with the date of design. The working space is defined as the area less than an arms-length away from the user's body, so as to eliminate unsolicited help on cards with which the user is not interacting. We are experimenting with displaying different amounts of help depending on the degree of tilt.

Early observations showed that Tangible Tooltips were surprisingly intuitive because users tend to hold cards perpendicular to the camera when interacting with them. Tilting the data card also borrows from our interaction with everyday physical objects, e.g. to find out information about the contents of a package, we turn it around to look at the label on the back. Initial testing, however, suggested that users occasionally tilt the cards for short periods of time while they perform other tasks, such as evaluating their layout from different angles, or looking down to pick them up. Because tooltips rising out of the card at this point are distracting, we implement a small delay (~1 second) before the help message is rendered. This is not unlike GUI Tooltips systems in which the cursor must linger over a region for some time before help is shown.

DISCUSSION AND FUTURE WORK

The Tiles system and help techniques were demonstrated at IEEE/ACM ISAR2000. About seventy users, including many AR researchers, tried the system. With simple instructions, users were able to simulate the design process, laying out and rearranging the instruments on the board and acquiring help when necessary. DaimlerChrysler AG engineers are now evaluating the Tiles interface concept for feasibility in industrial applications. We are currently looking at conducting formal user studies to evaluate the MR help systems and improve the design of interaction techniques.

CONCLUSION

We have described interaction techniques for providing MR users with on-demand, in-place multi-modal help with multiple levels of detail. Although design of the help systems has been an important area of research in traditional user interfaces we are not aware of prior attempts to design help interfaces for AR and MR environments. We presented two help techniques, *Tangible Bubble Help* and *Tangible Tooltips*, that we implemented within the *Tiles* MR environment. These techniques provide users with simple tools to acquire help without shifting focus away from the main task.

REFERENCES

1. Azuma, R. (1997). A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6, August 1997, pp. 355-385.
2. Billighurst, M., Poupyrev, I., Kato, H., May, R. (2000). Mixing Realities in Shared Space: An Augmented Reality Interface for Collaborative Computing. *ICME 2000*, pp. 1641-1644.
3. Granda, R., Halstead-Nussloch, T., Winters, J. (1990). The perceived usefulness of computer information sources: A field study, *SIGCHI Bulletin*, 21(4), pp. 35-43.
4. Neerinx, M., DeGreef, P. (1993). How to aid non-experts. *InterCHI '93*, pp. 165-170.
5. Poupyrev, I., Tan, D.S., Billighurst, M., Kato, H., Regenbrecht, H., Tetsutani, N. (2001). Tiles: A Mixed Reality Authoring Interface. *INTERACT 2001*.
6. Rekimoto, J., Ayatsuka, Y., Hayashi, K. (1998). Augmentable Reality: Situated Communication through Digital and Physical Spaces, *IEEE ISWC '98*, pp. 68-75.
7. Rekimoto, J. (1996). Tilting Operations for Small Screen Interfaces, *UIST '96*, pp. 167-168.

Ubi-Finger: Gesture Input Device for Mobile Use

Koji Tsukada, Michiaki Yasumura

Graduate School of Media and Governance, Keio University
5322 Endo Fujisawa, Kanagawa 252-8520, Japan

Abstract. This paper proposes a new interface in mobile environment called "Ubi-Finger" that realizes sensuous operations for PDA and information appliances by human gesture. Since gesture-input interfaces enables sensuous operations for users, there have been many researches about them especially for Virtual Reality. But almost those existing systems are very expensive and large, and not considered to be used in mobile environment. Ubi-Finger is a gesture-input device, which is simple, compact, and optimized for mobile use. We developed a prototype that enables to control real-world devices with natural gestures.

1. Introduction

As the popularity of personal digital assistants (PDAs) and cellular phones has increased rapidly in recent years, more and more computers are used in mobile environment. In such situations, the user interface is required to be compact and easy to use, compared to conventional computers in office or school environments. Many researchers have developed input devices in such mobile-computing environment [1][2][3]. But many of them focus on more efficient methods of text inputs, and have not enough considered more natural methods with another modality.

Moreover, people expect more and more computers and networks to spread into households in the near future, and various kinds of home electronics become information appliances connected to the network. In such an environment, controlling appliances may become more complicated, increase the difficulty for users to master different operations for each appliance. Therefore, the interface is expected such that it can control various information appliances with simple and friendly operation.

This paper focuses on the human gesture with fingers to realize a new interface, which is intelligible for anyone and can be used in various situations. We propose a wearable gesture input device called "Ubi-Finger" which realizes easy operations for PDA and information appliances and developed a prototype system.

2. Ubi-Finger

2.1 Concepts

We purpose a wearable device, which can be worn anytime and be used flexibly on various situations. The main concepts of Ubi-Finger are: (1) friendly operations with natural gestures, (2) optimized for mobile use, and (3) multiple use with the common interface.

As is widely alleged, nonverbal means of communication is more important than verbal means. The human gesture is a typical example of nonverbal communication. It is not only performed unconsciously on usual conversation but is main means of communication between foreigners or language-disabled. Thus the human gesture is one of the most useful means of communication and naturally performed by anyone. For this reason we focus on the human gesture. Although there have been many researches about gesture input interfaces especially for Virtual Reality, many of those existing systems are very expensive and large, and not considered to be used in mobile environment. Our approach is unique to utilize the gesture input method for controlling PDA and information appliances in mobile environment.

2.2 Device Configurations

Ubi-Finger device consists of three sensors, which are a bending sensor, an acceleration sensor, and a touch sensor, to recognize finger-gestures, an infrared

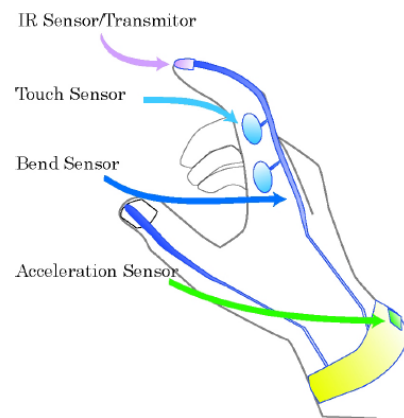


Fig.1 Device Configuration

LED to detect a device in the real world and a microcomputer to send input data from sensors to a host computer. Each sensor generates the information of finger motions, such as (1) a bending degree of an index finger, (2) rotation angles of a hand, (3) push operation of buttons by a thumb. And the system recognizes user's gesture at a host computer and runs an appropriate operation (Fig.1).

2.3 System Configuration

Next, we describe the whole system for operating information appliances in the real world. The system consists of four main parts: (1) Ubi-Finger, (2) information appliances connected to the network with a unique ID, (3) a host computer or PDA to recognize the user's gesture, (4) a server to associate the device's ID numbers with the user's ID or operating commands.

First, a user points at an information appliance with Ubi-Finger, and detects the device by transmitting a signal, which includes his unique ID via infra-rays. When each information appliance receives the signal, it transmits the device's ID and the user's ID to the server, and the server associates each ID by using the device database. Next, the user performs a gesture with his finger to control the information appliance. The gesture is recognized at the host computer and converted to a specific gesture-type, then transmitted to the server with the user's ID. The server selects the target device associated with the user's ID, converts the gesture-type to an appropriate command and controls the device by transmitting it.

In this system, (1) a user can point at a target device with his index finger, (2) then he can control it flexibly by performing natural gestures with his fingers. Since it is not necessary to learn different operation methods for every appliance, users can control devices more easily than the conventional ways. And by using corporeality and existing metaphor to control appliances, it becomes easier to use for anyone. We made a prototype system in order to verify the usefulness to operate devices in the real world with simple gestures.

3. Prototype

We have developed the prototype of Ubi-Finger focused on controlling devices in the real world (Fig.2). The system configurations of the prototype are as follows. The prototype consists of some sensors, infrared and normal LEDs, a microcomputer and a note PC. To detect motions of user's fingers, we apply a bending sensor and two switches in the portion of an index finger, and a 2-axis acceleration sensor on back of the hand. And there are also infrared LED and two LEDs on the portion of an index finger to detect a target device in the real world and visualize the current status of Ubi-Finger. When the switch is pushed by a thumb, the microcomputer will begin to transfer the input data from sensors to note PC via serial

interface. Then the note PC detects the form of fingers in real time and when it recognizes a specified gesture, Ubi-Finger performs as a PC input device or a remote controller for appliances according to the setup of software.

As mentioned, we have developed applications to control some devices in the real world with a simple gesture. For example we realized a light control with "bend down/up" gesture of an index finger, and also volume and channel controls of TV with "rotate right/left" gesture of a wrist. Since suitable information appliances don't exist at hand, the prototype system currently controls appliances directly by infrared signal. We are going to develop the system as soon as possible in which (1) a user points at a target appliance via infrared signal, (2) then he can control it with gestures via network.

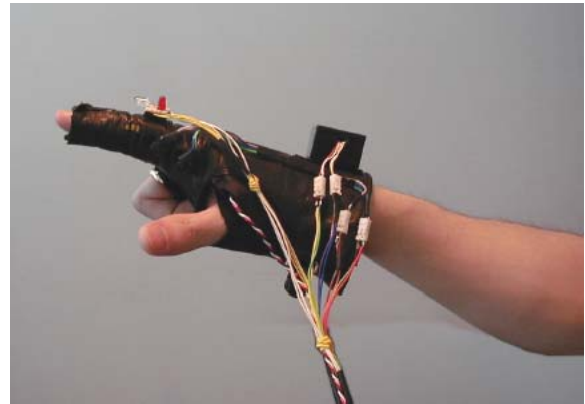


Fig.2 Prototype

4. Conclusion

We have introduced a new interface in mobile environment called "Ubi-Finger", which realizes sensuous operations for PDA and information appliances with human gesture. We have also developed the prototype of Ubi-Finger focused on controlling devices in the real world. With the proposed system users will be freed from complicated operations of various information devices and be able to control them more easily and joyfully.

References

- [1] Fukumoto, M. and Tonomura, Y. Body coupled FingerRing: Wireless wearable keyboard. Proceedings of the 1997 Conference on Human Factors in Computing Systems, CHI. 1997
- [2] Starner, T. et al. Wearable Computing and Augmented Reality. Technical Report355, MIT Media Lab. 1995
- [3] Twiddler2.
<http://www.handykey.com/site/twiddler2.html>

Lessons Learned in Creating Real-World Interfaces

David C. Wrighton, Dillon T. Bussert, D. Scott McCrickard
Department of Computer Science
Virginia Polytechnic Institute and State University
{dwrighto, dbussert, mccricks}@vt.edu

ABSTRACT

Informational displays have been developed that use lighting, air flow, and physical objects external to the computer screen, but typically lacking are simple and straightforward steps for creating these displays. This paper describes our experiences in creating a real-world interface (RWI) using X10 devices and common household appliances, and outlines our framework for creating RWIs quickly, easily, and inexpensively.

1. INTRODUCTION

The goal of this work is to develop a framework and programming interface for creating interfaces using real-world objects. We call the resulting informational displays real-world interfaces, or RWIs. Typical user interfaces use buttons, scrollbars, sliders, and similar on-screen visual displays to convey and interact with information, whereas RWIs will augment or replace these displays with changes in the surrounding environment that will convey information in a less direct but noticeable manner. For example, changes in lighting could correspond to the upcoming forecast for temperature, changes in air flow could reflect fluctuations in the stock market, and changes in ambient music could signal an upcoming meeting. We plan to provide programmers with the ability to use real-world devices in much the same way that they would use a standard user interface toolkit.

As a first step, we have constructed a RWI that displays weather information using changes in lighting and air flow. The RWI was created using X10 devices, which provide the means to send signals from computers to household devices like lamps and fans. As X10 devices are typically not used in this way, we found several issues at the programming interface level that makes it difficult to create robust, reliable RWIs. We describe these issues and outline methods for dealing with them.

This work was inspired by several projects in the area of ubiquitous computing. Mark Weiser from Xerox PARC developed the notion of “calm technology” that seeks to encalm and inform simultaneously [3]. The work is illustrated by artist Natalie Jeremijenko in her “Dangling String”, an attractive wire hanging from the ceiling in a hallway that reacts to signals from an Ethernet connection. A quiet network results in only occasional twitches, while a busy network causes the wire to whirl around noisily. Hiroshi Ishii and his Tangible Media Group at MIT have developed the concept of tangible user interfaces that use physical objects as tangible embodiments of digital information [2]. For example, a light pattern on the ceiling reflects the activity of the lab hamster and traffic noises indicate the level of network traffic. Scott Hudson at CMU designed an information percolator that uses bubbles passing through transparent tubes to display images and patterns in a pleasing yet non-intrusive manner [1].

While these and similar projects provide interesting theories and examples of interfaces outside the desktop, few steps have been taken to enable programmers to build their own devices. While previous work has argued that lighted interfaces in the environment

may be too distracting or that certain elements like air flow may be too difficult to notice or too difficult to control, one can certainly think of scenarios when people *want* an interface to distract them (when the interface is used as an alarm) and *would* notice air flow (when it was placed near a door or narrow hallway). We hope to empower designers with the techniques necessary to create real-world interfaces quickly, easily, and inexpensively.

2. APPROACH

In our work, we are leveraging the X10 protocol and hardware. The X10 protocol defines an overlaying method for sending signals over power lines between X10 hardware devices. Using this protocol, up to 256 different channels could be used with a single in-house power grid. X10 hardware devices send signals from transmitters to receivers. Originally, transmitters were wall-mounted switches or remote controls, though devices have emerged allowing users to control and check the status of household receivers from a computer via a device that connects to the serial port. The simplest receivers plug into any 110-volt wall outlet, toggling power to appliances plugged into them. More complex receivers include dimming and querying capabilities.

The X10 protocol has been in use for over 20 years, primarily for security systems (programming lights to turn on when away) and convenience (remote control access to lights and other devices, cameras, and motion detectors). However, we see significant possibilities in the areas in the communication and interaction with information. The lighting, sounds, and visual effects generated by typical household appliances have the ability to inform users, and the dimming capabilities with newer X10 devices provide a richer, more subtle set of cues.

As a first step, we wanted to gain experience in building interfaces that use X10 technology. Our goal was to understand the unique difficulties that arise in treating an X10 receiver like a typical widget in a user interface. We acquired an X10 PowerLinc device capable of sending and receiving signals from a computer, a number of X10 two-way modules that can receive signals and respond to state queries, and several household lamps and fans. Figure 1 shows the X10 components we used in constructing our RWI.

We created a RWI that uses changes in lighting and air flow to preview the weather forecast. Weather data are extracted from a popular weather Web site (www.weather.com), parsed to identify the predicted temperature and wind in the upcoming hours relative to the current measured values, and used to control the brightness of a lamp and the degree of air flow from fans all connected to X10 devices. The RWI can continually preview the weather in a set number of hours (say, three hours from now), or it can cycle through the predicted weather for a given number of hours (say, for the next 12 hours). The RWI is located in a lab near a door, with the expectation that as people leave the lab, they will have a sense of the weather outside and will be reminded to bring a coat or hat, if necessary. In addition, those in the lab can look at, listen to, or feel the effects of the RWI to get a sense of the weather outside.



Figure 1: X10 components used in constructing a RWI. The black box is the Powerline interface that allows a computer to send signals via the serial port through the power lines. The white boxes are X10 receivers into which appliances can be plugged. Each receiver should have a different code, with up to 256 different codes available.

We encountered numerous problems in designing our initial RWI, mainly because the existing X10 command set is not designed to be utilized in the same way as a typical user interface toolkit. Information can only be communicated very slowly, at a rate of approximately 4 bytes per second. When information was transmitted more quickly, X10 devices reacted unpredictably or erroneously. It is not surprising that the developers of the X10 protocol did not consider these issues in designing the toolkit as they did not expect people to use the commands in the way that we do. However, to effectively create interfaces using X10 devices, it is necessary to mask these problems. As such, we plan to create a module of functions for Perl that will provide an effective programming layer for using electrical appliances in the development of interfaces.

Below are some key concepts that we are incorporating into our module.

- **Provide a non-blocking programming interface for all commands.** Given the slow transmission time in sending commands, it is unreasonable to require programs to halt while waiting for completion and verification of the commands. As such, systems that interface with our library will not block upon calls to our library. This becomes far more critical as the systems we interface with do things other than just create their output through our library.
- **Provide built-in status/error checking, detection, and recovery.** We plan to check for the presence of a signal on the interface before it begins its transfer. There are several reasons for this, but primarily, it is poorly defined how frequently one should check for the presence of a communication on the line. To avoid failure in trying to send a command too soon after it sent the last one, despite the device reporting that it is ready to send, we plan to maintain state at the controlling computer, then use regular communication between the home X10 device and the remote appliance devices to help ensure consistency.
- **Associate special meanings with remote control access to X10 devices.** X10 users can control electrical appliances in a variety of ways: through the computer, via wall switches or motion sensors, or with a remote control. Our module will allow programmers to associate different behaviors with each control. For example, dimming a RWI using a remote control may mean that the end-user wants less noticeable alerts from the RWI: less intense lighting changes, quieter audio alerts,

etc. Alternatively, it may signify that the end-user is less interested in seeing updates, so the rate at which the RWI is updated may decrease with dimming. Our module will provide the programmer with the power to associate unique meanings to external end-user inputs.

Our next step will be to create a Perl module that provides a programming interface that incorporates the previously mentioned concepts. The commands will be based on the ones used to interface Tk, a popular graphical toolkit. With compact, simple commands using a syntax familiar to any Perl programmer, a programmer can control the power levels to an electrical device or can associate the power levels with changes to variables. The end product will be a Perl module that, with the appropriate hardware, will allow programmers to create applications for real-world interfaces.

As we develop the module, we plan to construct additional RWIs. In so doing, we expect that we will learn more about the needs of programmers in developing RWIs, knowledge that we will use in designing and improving the module. Specific areas in which we plan to develop RWIs include the following:

- **Augmenting handheld computers.** Handheld computers like the Palm have enjoyed a burst of popularity in recent years, but their small screen sizes make their output abilities somewhat limited. By grabbing desktop synchronization data, we will develop RWIs that integrate with the core handheld programs: calendars, address books, to-do lists, and notepads. For example, a RWI could cause a fan to blow progressively harder as a meeting in the calendar approaches, or the brightness of a lamp could represent the percent of completed items in the to-do list.
- **Computer management.** We intend to support awareness of computer system behavior at both the user level and the administrator level. For typical users, RWIs could be used to show the ambient temperature inside case, temperature of speed of computer fan, CPU usage, and battery availability. For system administrators, RWIs could reflect system logins, network usage, or server hits. One potential way to show this information would be to use an aquarium bubbler, where problem levels in the system would be reflected with a loud motor and lots of bubbles, thus providing audio and visual cues of the problem.
- **Tracking Web information.** Information on the Web changes frequently, to the point that it is impossible for a user to keep track of new information. These changes can be monitored with a script and displayed using a RWI. Stock prices, traffic data, weather data, news, sports scores, and library checkouts all exemplify this type of information.

When the development of the RWI module has been completed, we plan to use it in human-computer interaction courses. The students in the classes will use the toolkit to create RWIs similar to ones in the list above or of their own invention. Upon completion of the classes, we expect to have a toolkit and set of sample applications that is robust enough for general use.

3. REFERENCES

- [1] Jeremy M. Heiner, Scott E. Hudson, and Kenichiro Tanaka. The information percolator: Ambient information display in a decorative object. In *Proceedings of UIST*, pages 141–148, November 1999.
- [2] Hiroshi Ishii and Brygg Ulmer. Tangible bits: Towards seamless interfaces between people, bits, and atoms. In *Proceedings of CHI*, pages 234–241, March 1997.
- [3] Mark Weiser and John Seely Brown. Designing calm technology. *PowerGrid Journal*, 1.01, July 1996.

Bringing Application Functionality to Small Devices

W@Pnotes

Andreas Zeidler

Databases and Distributed Systems Research Group
Wilhelminenstr. 7, 64283 Darmstadt, Germany
+49 6151 16 6233
az@ubicomp.de

ABSTRACT

User interface (UI) design for ubiquitous computing is different from UI design for classical desktop-oriented applications. Interaction of a user with the system and applications takes place in a highly distributed fashion. Input- and output functionality is distributed over space, as well as time. Human-computer interaction (HCI) has to take place wherever and whenever needed. Due to the highly dynamic pattern of use, “classical” UI design for a single screen is poorly suited.

We present an architecture for designing applications bringing different in- and output channel to particularly limited devices, like mobile phones, as well as taking advantage from the benefits of a desktop computer. The W@PNotes system serves as a proof of concept for the successful use of this architecture.

Keywords

Function shipping, ubiquitous computing, Wireless Application Protocol.

INTRODUCTION

Obviously, in the vision of ubiquitous computing [1] “distribution” of human-computer interaction is crucial. The user and her need to interact with her applications easily have to the main focus of UI design. Given this and the need for ubiquity, the term “distributed systems” must be defined for the distribution of system functionality as well as for the distribution of UI functionality. Apparently, the distribution of UI functionality needs new paradigms of design. UIs have to become “smart” and “intelligent”. From a top-level point of view, two distribution schemes for UI functionality to actual devices seem to be appealing: (1) Use the input- and output devices available “here and now”, including public displays and/or terminals, (2) try to take advantage of devices with some input/output capabilities carried around by the user anyway, like mobile phones or personal digital assistants (PDA).

The W@PNotes-system presented here serves as an example for the second approach, although it incorporates some aspects of the first design dimension. The main goal of the system is (1) the design of smart and distributed user interfaces and (2) design of an architecture, which is able to (a) separate and distribute system’s functionality over several computers by means of service-oriented design using Jini [3] as an underlying distribution infrastructure and (b) bring advanced functionality to a very limited device: A mobile phone with an integrated browser for the Wireless Application Protocol (WAP) [6].

RESEARCH GOALS

The W@PNotes project is an ongoing research project. The research goals of this project can be summarized as follows:

- Design and architecture of APIs for ubiquitous accessible applications.
- Design of UIs for ubiquitous accessible applications.
- Distribution of application functionality among various components and separation of concerns by using service oriented programming.
- “Good” composition of concerns, accordingly.
- Distribution of customized HCI functionality among various UIs.
- Exploration of UI limitations with respect to device capabilities

The main research focus of this project lies on the design patterns for designing distributed applications on top of and as proof-of-concept for an ubiquitous computing infrastructure. This includes the “how” of distribution by means of separation of concerns between different components (services), as well as “composition of concerns”, by means of a supporting framework for combining all those autonomous parts of a system in a reliable way.

The choice of a mobile phone with integrated WAP browser as the central UI/device in the W@PNotes project has three reasons: (1) it is the most ubiquitous device available today, so we can explore the limitations of devices we already have at hands; (2) it has built-in access to the world-wide web via WAP; (3) in many respects it is a very limited device and therefore challenging for an infrastructure.

Throughout the next section we describe the realization of this research goals in the W@PNotes system.

W@PNotes -- Architecture and Design Goals

The functional part of the W@PNotes-system is a distributed and multi-user enabled PostIt-like application, which is neither new, nor exciting. The application itself serves only as a demonstration and proof of concept for the underlying architecture for composing application out of separated services and bringing application functionality over a network to all sorts of devices. The functionality is limited to some of the features which can be expected within a PostIt-like application: Addition and naming of a note, deletion, altering of items in a note. Moreover, some convenience functions were defined.

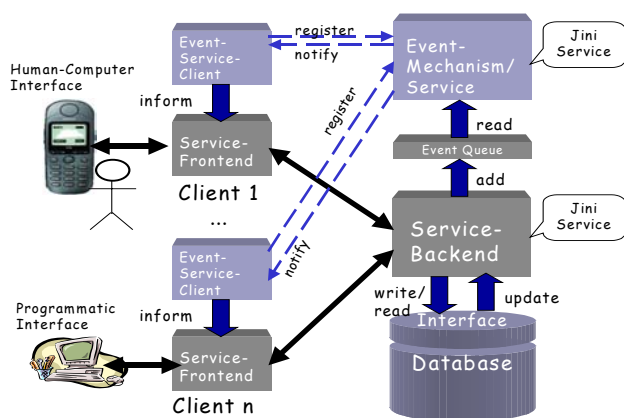


Fig. 1: Top-Level Architecture

The PostIt-like application consists of a single backend database, storing and maintaining the actual notes for each user of the system (see figure 1). Through a well-defined interface other components of the application gain access to the stored messages. Separated, except for a common message-queue, an event mechanism can be coupled to the database. This mechanism, following the observer pattern, can communicate changes in the database to interested listeners via the remote event mechanism defined by Jini, which serves as easy-to-use mechanism for UI synchronization. From the architectural point of view a central goal of this design stage was, to explore to which extend separation of concerns can be achieved by using services as components of a system and therefore, how and whether systems can be composed out of autonomous entities exclusively and what impact such a paradigm has on the design process. We define this as service-oriented design. Both services are registered as fully qualified services in the Jini registry, called Lookup Service (LUS) [5], and can be found by the standard lookup mechanism [4] provided by the Jini infrastructure. The components are autonomous, functionally separated, and can communicate only through well-defined interfaces, defining their functionality on a syntactical and type-safe level. The overall functionality of the application is achieved by dynamically combining appropriate services at runtime, using the infrastructure provided by Jini and the framework defined within the W@PNotes system. We omit the runtime behavior because of lack of space and refer to the poster.

The architecture is easily extensible. As both services, W@PNotes service and event service, are decomposed into their functional part and their UI or client, respectively, new UIs and event clients can be “plugged in” at runtime by re-registering the service with a new set of attributes.

The WapUI programmatic interface is transported and initialized at runtime and has the functionality of a W@PNotes-to-WML (Wireless Markup Language [6]) gateway. Part of every WML-document are the command

options the user has at this stage of application interaction, encoded as links in the document. Parts of the WML documents are input fields for the user (e.g., adding a new item to a note is translated to an input field). The input typed in here is translated by the WapUI to an appropriate method call on the W@PNotes service (i.e., `addItem(<UniqueNameofNote>, <itemtoadd>)`).

The W@PNotes architecture is then responsible for delivering the changes made on the mobile phone to all other user interfaces currently registered with the service. One major drawback of using WAP is visible here: The synchronization mechanism is not able to *push* changes made on other UIs to the mobile phone. The change is visible only if the user is requesting the document on which the change is visible. This drawback is getting importing when thinking of allowing other users of the system to add notes for a user to the system.

Summary and Discussion

Various research goals have been tackled in the architecture presented here. We feel distributed user interfaces and HCI are touching many layers of a distributed system. Especially, separation of concerns, like, separation of function and UI is very important. Otherwise adaptivity and extensibility are hard to guarantee. Also, on lower layers of an infrastructure for ubiquitous computing the choice of services as separated components seems to be a feasible approach. Here, asynchronous delivery and encapsulation of state changes into discrete events seems to be a good approach. But, as mentioned, not every protocol is well suited for delivering changes to a device. WAP as an example is not able to *push* content to a device like a mobile phone. Here, the WapUI is the terminal point of the architecture. Also, the limitations of a mobile device imposed by the small display and the keypad as input device are painful. This limitation might get weaker in the near future with the advent of new protocols and devices for the consumer market. To summarize, we think that the approach presented in this paper, as an example of ongoing research in the field of UI design for ubiquitous computing, is both: Promising and far from being finished.

REFERENCES

1. Mark Weiser. The Computer for the Twenty-First Century. <http://www.ubiq.com/hypertext/weiser/SciAmDraft3.html>
2. ServiceUI Project. <http://www.artima.com/jini/serviceui/>.
3. Sun Microsystems Inc. Jini Architecture Specification– Revision 1.1, 2000.
4. Sun Microsystems Inc. Jini Discovery and Join Specification– Revision 1.1, 2000.
5. Sun Microsystems Inc. Jini Lookup Service Specification– Revision 1.1, 2000.
6. The Wireless Application Protocol Forum. <http://www.wapforum.org>