Using Mobile Devices as Universal Personal Controllers

Carnegie Mellon University

Proposal to the Pittsburgh Digital Greenhouse, Inc.

http://www.digitalgreenhouse.com/

Project Proposal Solicitation 00-2

Technical Point of Contact: Brad A. Myers
Human Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-5150
bam@cs.cmu.edu
http://www.cs.cmu.edu/~bam

Administrative Point of Contact: Robert Kearns
Office of Sponsored Research
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
tel: 412-268-5837
fax: 412-268-5841
email: rk2a@andrew.cmu.edu

Overall program cost: $ 279,295

Overall Program Duration: 24 months

Executive Summary

Personal digital assistants (PDAs) like the Palm Pilot are becoming increasingly ubiquitous, and with wireless technologies such as BlueTooth and IEEE 802.11, they will be in close interactive communication with other devices. Furthermore, cell-phones and pagers, which primarily used for communication, are increasingly becoming programmable. We propose to investigate how these kinds of hand-held devices can be used to control all kinds of home, office and factory equipment. The concept is that when users point their own hand-held at a light switch, at a photocopier in an office, at a machine tool in a factory, at a VCR at home, at a piece of test equipment in the field, or at almost any other kind of device, the device will send to the hand-held a description of its control parameters. The hand-held uses this information to create an appropriate control panel, taking into account the properties of the controls that are needed, the properties of the hand-held (the display type and input techniques available), and the properties of the user (what language is preferred, whether left or right handed, how big the buttons should be based on whether the user prefers using a finger or a stylus). The user can then control the device using the hand-held. The device will not need to dedicate much processing power, hardware, or cost to the user interface, since it will only need to contain a description of its capabilities and storage for the current settings, along with hardware for wireless communication. The hand-held programs will use intelligent "model-based" techniques to create useful and appropriate interfaces that are customized for each user.

Technical Background

The proposed research is part of the larger Pebbles research project (http://www.pebbles.hcii.cmu.edu) [10], which is investigating how hand-held devices will be used *at the same time* as other computerized devices. As part of the Pebbles project, we have looked at multiple PDAs connected to a PC to support meetings [12]. For example, in design reviews, brainstorming sessions, and organizational meetings, a PC is often used to display slides or a current plan, and the people in attendance provide input. Our applications allow each person to use their PDA to control the PC's cursor and keyboard input from their seat. In other research, we are investigating how a PDA can be equally useful for a single person as an extension to the Windows user interface for desktop applications [11]. We performed a study that showed, for example, that the PDA could be used very effectively as a scrolling device for desktop applications [9].

Specifically related to the control of devices, a 3Com Corporation project has created a Palm remote controller for an Internet phone [2]. There are many commercial remote controls that come pre-programmed for a variety of devices, or which allow the user to program each function. For example, Harman-Kardon and Microsoft collaborated on a PDA-like remote controller [7] that can be programmed by the user to control up to 15 devices. Similarly, the Omni-Remote program [15] allows a Palm Pilot to be programmed to control many devices. Our approach is different than all of these because we are *not* pre-programming the devices or requiring the user to program the control panels. Instead, the system will automatically create control panels based on the properties of the device.

"Model-based" techniques [18] have been investigated for the automatic design of user interfaces. Here, the programmer provides a specification ("model") of the properties of the application, along with specifications of the user and the display. This approach was moderately successful at creating dialog boxes [8] [19] and creating complete interfaces in a limited range [14] [5] [18]. Other systems focused on the initial creation assuming a user would edit the resulting user interface [4] [16]. We plan to extend these results to create panels of controls on hand-helds of significantly different properties.

Proposed Program of Research

Introduction

The user interface for next generation network systems that will interface with digital TV, home appliances, personal communication devices and other electronics must be simple enough for the general consumer to use, and yet be sufficiently sophisticated to provide access to the enhanced capabilities provided. One component of such a user interface will be advanced "recognition-based" user interfaces such as speech recognition and systems that use cameras to see the user. The integration of these multi-modal interfaces into the future "smart environments" is being explored by many groups, including CMU's Aura project (http://www.cs.cmu.edu/~aura/). One component that has been missed by these projects is that people will be carrying new wireless mobile data communication devices into the environments. The Pebbles project encompasses a broad research program into how the user interface and functions can be spread across the mobile, fixed and embedded devices in an environment. For example, sometimes it may be appropriate to speak a command (e.g., saying "lights on"), but other times, it may make more sense to use a control panel on a hand-held mobile device (e.g., to move a continuous slider to adjust the light brightness, rather than saying "darker, darker, darker, OK" or "lights to 32 percent").

One part of this work is funded as part of the "Command Post of the Future Project" (http://www.cs.cmu.edu/~cpof/), which aims to make commanders more effective. The goal of the Pebbles part is to investigate how *information* can be easily and fluidly moved between a large group display, and individual hand-held devices. For example, staff people can walk around the command post and use their hand-helds to "drill-down" to get more details about information shown on the public display. By using their hand-helds, they avoid disrupting the commander's and the group's work on the main screen.

We hope that the Pittsburgh Digital Greenhouse will fund a new direction for Pebbles, where we investigate how to use mobile hand-helds to *control* other devices. We feel that this will be an important capability for hand-helds that has yet to be investigated. Our research will create new technology to enable this to happen and perform the appropriate human-computer interaction research to refine and validate the designs.

Complexity of Devices

An important motivation for the proposed research is the increasing complexity of consumer and business devices. Most desk phones, cell phones, clock radios, VCRs, stereos, washing machines, microwave ovens, thermostats, photocopiers, fax machines, etc. have many unused features. Most consumers find it difficult to master the basic functions of some of these devices, never mind the sophisticated features. Even simple devices are not immune to this problem; when traveling, I am frequently stumped by the user interface for setting the alarm on the clocks in hotel rooms.

One reason that devices are so difficult to use is that the manufacturers cannot afford to spend sufficient resources on the user interface. In order to cut costs, the manufacturer often economizes on buttons and displays by mapping multiple functions to the same buttons with terse labels and simple indicators. Most devices probably do not get the extensive usability testing [13] that might help identify usability problems early in the design phase. And there is little standardization of the labeling, placement or behavior of the controls on consumer devices. The increasing computerization of equipment can only make this problem worse, as it becomes cheaper to embed sophisticated computing in even the smallest devices. Today, enormous processing power can be very cheaply embedded in devices as simple as a light-switch, but providing a good user

interface for that processor is still a significant expense. Many devices today come with custom remote controls, which themselves may be poorly designed. Furthermore, the consumer is faced with the familiar problem of having dozens of incompatible remote controls to deal with.

The approach of the proposed Universal Personal Controller aims to alleviate these problems. Wouldn't it be nice to be able to point your own PDA at the hotel clock and use a familiar and well-designed user interface on the PDA to set the alarm? We envision devices of the future that will be able to describe at a high-level their control parameters so that a mobile hand-held will be able to construct or find an appropriate control panel. By off-loading the user interface onto a user's hand-held, more money and effort can be spent on the hand-held controller than would be practical for the devices being controlled. Furthermore, as described below, we expect that the user interface on the Universal Personal Controller will be better than the user interface of today's remote controls.

Advantages for Automatic Design of Control Panels

The most interesting and challenging part of the proposed research is to automatically create panels of controls for devices. Most remote controls today are pre-programmed in the factory for a single device, and sometimes for a few others. For example, most VCR remotes can also control TVs. This approach is clearly not scaleable to many devices, and the remotes often only cover the basic functions and omit useful features the consumer might want (e.g., mine does not have a "mute" button). The other approach used today requires the consumer to laboriously hand-program each function of the remote. For example, with the Harman-Kardon "Take Control System Controller" and Pacific Neo-Tek's "Omni-Remote," the user must have the old remote for the device to be controlled. The old remote is aimed at the controller, and, for each operation to be provided on the controller, the user must specify a button on the controller and then push the corresponding button on the old remote. In contrast, the proposed Universal Personal Controller will be able to create a control panel for new devices without requiring any programming by the user.

One approach might be to have the device send to the Universal Personal Controller a fully-designed control panel. This is similar to the philosophy of Jini [17], which sends a complete implementation of the user interface for the requested service. For the device control envisioned here, sending a pre-defined panel will not work because of the wide range of hand-held mobile devices, with vastly different input and output capabilities. We want to support cell-phones with 8-line displays and 14 buttons, monochrome and color Palm OS devices with 160x160 pixels and a touch screen, color Pocket PC devices with 240 x 320 pixels, etc. In the Jini model, the device would need to provide many different control panels each tuned for a different kind of hand-held. Furthermore, the devices we want to control are expected to last for many years whereas the features of hand-helds change every year. For example, we would want the same light switch or dishwasher that can be controlled by a PDA to work with a future controller built into a watch, which might have a new kind of round display and only four buttons.

Therefore, we propose to have the device send to the controller a high-level specification of the parameters and controls for the device, and have the controller automatically *create* a user interface based on this specification. This will enable the same specification to be used by controllers with significantly different characteristics. The automatic generation of the user interface will take into account the input and output capabilities of the controller along with user preferences and other information. Users might specify, for example, that they want bigger buttons and bigger labels because they want to press them with a finger, or alternatively that they prefer smaller buttons so more will fit on the screen at the same time. Since the

display on the hand-held is likely to be bigger than on the device, longer names and even help text can appear on the controller. If automatic machine language translation is available, the user might be able to request a French or Japanese interface, and have the textual labels translated by the controller.

Another advantage of automatic generation is that the controller can impose consistency across all the devices that are controlled. For example, both a phone and a clock radio have a volume control, so panels for both should use the same widget for volume and put it in the same place. Similarly, when the controller is pointed at a clock from any manufacturer, the same control panel should appear on the controller. The panels should also be consistent with other applications running on the same hardware. For example, a controller on a Pocket PC will use the standard widget for setting the volume (a slider next to a speaker icon), but on a Palm, the volume usually is represented as a pop-up with a few choices.

Dynamically created user interfaces to networked devices, such as telephones, enable new kinds of services to be automatically incorporated into the user interface even after the devices have been deployed. For example, if the phone system starts offering "call-back-last-number," this might automatically appear as a new labeled button on the controller, rather than requiring the user to remember to press *69.

Another possible advantage is that the controller can serve as an "authentication token." Some devices, such as the 3Com Internet phones, require the user to login. Pointing a PDA at the phone is much easier than keying in a user name and password onto the phone keypad. If stronger authentication is needed, technologies like Secure Computing's SafeWord or SofToken (http://www.securecomputing.com/), which already run on Palm OS and Pocket PC, might be incorporated into the controller protocols. Once the user is authenticated to the device, then new capabilities are enabled. For example, the device might only provide certain control functions to certain classes of users. For example, televisions in hotels might only describe the controls to add new channels to authorized service people, whereas anyone's controller would be able to change channels and volume. Thermostats in offices might allow anyone to change the temperature within certain ranges, but only certain people might be authorized to switch from cooling to heating.

Device Discovery and Communication

How will the Universal Personal Controller find out about the devices? There is already a move towards "self-describing devices." For example, 3Com has developed an Internet telephone that can interface to a Palm hand-held [2]. The Palm can control certain features of the telephone using a standard protocol called "PhoneControl" [3] that 3Com has proposed as an IETF standard. PhoneControl provides for 2-way communication and allows the controller to inquire as to the capabilities of the telephone, as well as to initiate and terminate calls.

Many other "service discovery" and "service description" languages are being developed, usually based on the XML format. Currently, most of these are aimed at devices that today connect to computers, but extensions to other kinds of devices are inevitable. For example, Microsoft's "Universal Plug and Play" (http://www.upnp.org) includes a way for devices to describe their features, and has been embraced by "more than 120 consumer electronics, computing, home automation, home security, home appliance, computer networking and other leading companies" (quoted from a press release at: http://www.upnp.org/forum/default.htm). Other efforts include the Jini protocol [17], Salutation (http://www.salutation.org/), and parts of the WML standard for cell-phones [20]. It is interesting to note that the same ability to query about devices'

properties and to control devices remotely will also be necessary to enable other kinds of external control, including centralized computer control and voice control of devices.

Communicating with devices will also get increasingly easy. Today, many systems already have embedded infrared receivers, used by their remote controls. In the future, devices may support two-way infrared communication to enable the controller to receive information from the device as well. The 3Com Internet Phone works this way [2]. All of today's PDAs and many cell-phones also have two-way infrared capability (even the Furby toy has two-way infrared). The advantage of infrared is that it is already available and cheap. Infrared is usually highly directional and short range, which is both an advantage and a disadvantage. It minimizes security concerns if only controllers near a device can control it (people on the street cannot turn on your TV), and people can identify which device they want to control by aiming the controller at it. Disadvantages include that it is not possible to find out all the devices that are around, and the user may not know where to point the device (to control the lights, do you point the controller at the light itself, or do you need to point to the "regular" wall switch for that light?).

Other wireless technologies, such as IEEE 802.11, digital cell-phones and two-way pagers may also be useful. The new BlueTooth technology [6] is advertised to provide short-range (in a room) low-power omni-directional radio communication. Projections are that the chips to support BlueTooth will eventually match the cost and power usage of today's infrared chips. Many vendors, from cell-phones to PDAs, are expected to incorporate BlueTooth communication. BlueTooth promises to significantly expand the range of devices that can be remotely controlled. It is inherently a two-way communication mechanism, and it is not directional. Security is built into the BlueTooth specification, since it is designed for use with consumer telephones and other personal devices. We are hopeful that BlueTooth will be useful for finding out what devices are around and for communicating with devices without requiring the user to know where to point the controller.

Currently, we envision using a combination of BlueTooth and infrared technologies in our Universal Personal Controller research. We hope to get early BlueTooth devices when they become available. Initially, we expect to use plug-in modules such as the Widcomm BlueTooth card for the HandSpring Visor (http://www.widcomm.com/products/blueConnect.htm). For infrared communication, we can use technology such as Omni-Remote [15] to control the built-in infrared port on a Palm PDA.

For the Universal Personal Controller to be a reality, there would have to standardization on a communication protocol, and on the specification language for the devices. Our research will investigate the format and content of such a specification and demonstrate its usefulness. In the short term, there will clearly not be devices that can communicate with our Universal Personal Controller. In order to perform the proposed research, we will first acquire whatever devices we can find that support two-way inquiry and control, such as the 3Com Internet phone. We will adapt their service description protocols to our needs (see below). We will also select a few devices that already accept infrared *one-way* remote control, and create specifications for what they would report if they supported two-way communication. These specifications will then be used to generate user interfaces on the controller as if they had been supplied directly by the devices. We will work with our sponsors to decide what devices should be investigated and to try to get more devices that will report the necessary service description information.

Automatic Design of Control Panels

Automatically creating high quality control panels from a specification is a difficult research problem. Some previous approaches to automatic user interface generation only produced low-quality, rough outlines of the user interface, and required much work from the user to improve the interface [4] [16]. Our task is made more difficult by the desire to support hand-helds such as cell-phones with impoverished input and output capabilities.

In our previous work on automatic creation of user interfaces [19], we showed that we could generate reasonable quality user interfaces that varied based on the Look-and-Feel of the underlying platform. This work will form part of the basis of the proposed work.

As an initial step, we hand-designed control panels for a Palm Pilot for a complicated office telephone (the AT&T 1825) and for a clock-radio (the Bose Wave Radio) – see Figure 1. Initial human-factors studies suggested that our on-screen control panels were much easier to use than the original physical buttons on the devices, especially for complex tasks. For the proposed project, we plan to start by creating by hand some high-quality control panels for other devices. The panels will be designed for a variety of controller platforms, including cell-phones, Palm OS devices, and Pocket PC devices.

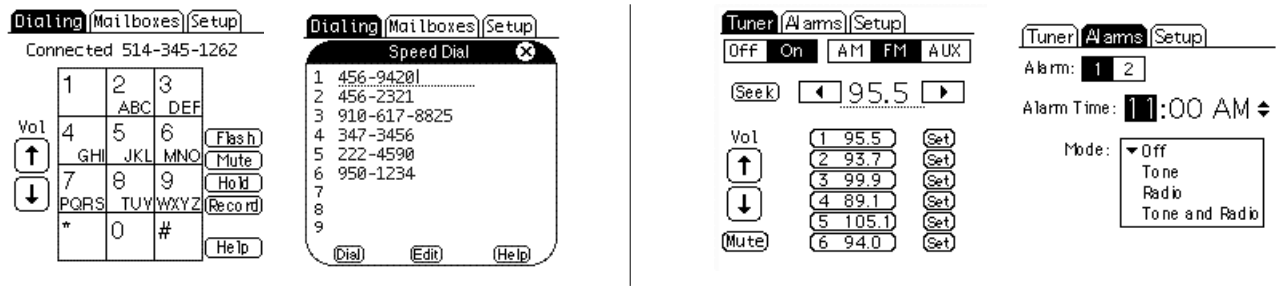

Figure 1. Hand-drawn panels such as might be automatically created for a phone (left) and a clock-radio (right).

The panels will be evaluated and refined using heuristic analysis by experienced designers and also by more formal user testing. We expect to be able to conclusively show through user testing that the control panels are easier to use than the original physical devices.

The next step will be to develop a set of controls (also called "widgets") along with descriptions of the types of data that they can handle. Our early work, and those of others, concentrated on primitive types, such as enumerated types (select one or more from a list), strings, commands, and numbers, and then assigned widgets for each (radio buttons or check boxes, text input fields, buttons, and sliders, respectively). In the proposed research, we will investigate the set of primitive widgets appropriate for the devices to be controlled. We will also extend the primitive types with higher-level types, such as "Volume," for which some controllers may have special widgets. If not, these higher-level types will decompose into primitive types (e.g., volume might be a number between 1 and 10). The specification will list the parameters with their higher-level type, and will be able to describe the lower-level component types if necessary.

The proposed system will also be able to add extra information to the types to help the user interface generator. For example, lists will be annotated with the usual number of entries to help choose the appropriate layout. String values will have formatting information to help the controller display them appropriately, and for input fields, to determine if the user has entered a legal value.

As an example, a specification for the parameter "Power" might be:

```
Type: Power        // high-level type for which there may be a special widget
Is-A: Boolean      // primitive type, in case there isn't a widget for power: value can be true or false
Label: Power || {Radio}[Off,On]  // various choices for how this might be labeled
Importance: High   // to tell the layout handler to make this element more prominent
```

The syntax and values of this example are just meant to be illustrative. We plan to use XML as the basis for the specification language, and will investigate whether any of the existing XML-based languages, such as uPnP, WML or PhoneControl can be extended to serve our needs.

The most interesting part of the descriptions will be the specification of the *relationship* of the parameters to each other. In many interfaces, certain controls are only visible in certain modes. For example, on a controller for a phone, the "hang-up" button should only be available when the phone is in use. Depending on the controller, the hang-up button might either be grayed out or else it might be invisible when it is not available. For other controls, the default or current value will be dynamically determined depending on other settings. These dependencies will all be in the specification that the device sends to the controller.

Another kind of relationship that seems to be needed is the grouping of related objects. We investigated grouping in our earlier work [19]. For example, on a phone controller, the flash, mute and hold buttons all should be together. At a higher-level, there will be groupings that might be translated into different panes on small screens, or to pop-up dialog boxes. This relates to the "card" concept in WML for WAP [20]. For example, on the phone, the widgets for dialing might be put on a different pane than the widgets for voice mail messages.

In parallel with developing the specification language, we will also be building software for generating user interfaces for the panels. This will start from the model-based algorithms used for our earlier work and that of others [8], augmented by more rules to insure that the layout conforms to the best design practices. We hope to be able to use a machine-independent language for the implementation, such as Java, so that we do not need to re-implement the software for the various machines.

We hope that the generated user interfaces will be very close to the hand-designed panels we started with. To measure whether they are "close enough," user studies will be performed to compare the hand-designed and automatically generated panels against the original physical devices. The automatically generated interfaces should be able to match the performance of the hand-designed panels.

User Customizations

After the controller has automatically generated a user interface, we propose to allow the user to customize it in various ways. This will use technology like in our ShortCutter application [13], which already provides the ability to hand-create and hand-edit panels of controls. We expect that only advanced users will want to customized their interfaces, but it does provide many advantages to those who do:

- Elimination of buttons for functions that are never used, to simplify the interface. The user might also edit the buttons for common functions to be more prominent.
- Supporting default values for frequently used parameters. For example, a clock controller can always come up ready to set the alarm to 7:00am.

- Constructing control panels that control *multiple* devices, by dragging in controls from different panels. Macros can even be created so that a single button will send multiple commands to different devices. For example, a button might lower the lights while turning on the TV, the VCR and the cable box.

- Sharing customizations. Third parties might even provide a market of specially designed control panels.

These customizations would be saved for the user and used for all devices of the same type. Techniques that combine designer-specified and user-specified constraints [1] might be used to implement customizations.

Conclusions

The proposed research addresses important design and implementation challenges in the development of "smart products" for home automation and in the use of handheld devices with wireless communication systems. The Universal Personal Controller would make an exciting commercial product, which might be further developed by the PDG firms or marketed through the creation of a new commercial enterprise. However, significant research is necessary before such a controller can be created, especially to investigate how the parameters of the device can be specified, and how a high-quality user interface can be created from that specification. Devices are already increasingly able to communicate and accept external control. If we can show success in this research, then manufacturers may be more motivated to also have the devices export the specifications of their parameters.

Program Plan, SOW, Milestones, and Deliverables

Deliverables will include:

- Hand-designed screens for various devices.

- The general specification language for device parameters, which might serve as the beginnings of a standard.

- Software application for multiple platforms (probably Palm, Pocket PC and WAP phone) that take the specification language and produce high-quality user interfaces.

- Simulated and actual control of devices using the generated control panels.

- Papers and reports describing the research results.

Our research group has a long history of producing systems that are made generally available and are widely used. The Amulet toolkit (http://www.cs.cmu.edu/~amulet) was used by hundreds of research and commercial projects and is still being downloaded about 200 times a month. Our current Pebbles project has produced software that is being downloaded about 200 times a week, and one application has been licensed for commercial sale.

We feel we need at least two years to perform this work, because of the extensive new research that is required.

Quarter	*Work*
Year 1, Quarter 1:	Create by hand the control panels for two more devices. Begin design of specification language.
Quarter 2	User tests of hand-designed control panels.
Quarter 3	Begin implementation of automatic generation on one platform.

Quarter 4	Initial implementation of automatic generation completed for various devices. Begin 2nd platform.
Year 2, Quarter 1:	Initial implementation on second platform completed. Begin third platform.
Quarter 2	User tests of generated user interfaces. Begin end-user customization support.
Quarter 3	End-user customization facilities available. Expand support to further devices.
Quarter 4	Full implementations completed.

Budget Summary

Proposal Summary Budget	**Two Years (11/01/00 to 10/31/02)**
Brad Myers 25%	$ 47,068
Jodi Forlizzi 10%	$ 12,844
One PhD student 100%	$ 88,097
Fringe Benefits	$ 14,676
Operating expenses	$ 14,508
Travel	$ 10,589
Capital Equipment	$ 11,000
Total Direct Costs	**$ 198,782**
Indirect Costs	$ 80,513
TOTAL	**$ 279,295**

Budget Notes: Fringe benefits are charged at a rate of 27.44% and 26.94% during year one and two, respectively. Fringe benefits do not apply to graduate students. Overhead applies to all categories at the 56% rate, except for graduate students who are charged at half the rate or 28% and capital equipment to which overhead does not apply.

Qualifications of the Investigators

Brad A. Myers is a Senior Research Scientist in the Human-Computer Interaction Institute in the School of Computer Science at Carnegie Mellon University, where he is the principal investigator for various research projects including: Silver Multi-Media Authoring, Natural Programming, the Pebbles Hand-Held Computer Project, User Interface Software, and Demonstrational Interfaces. He is the author or editor of over 200 publications, and he is on the editorial board of 5 journals. He has been a consultant on user interface design and implementation to about 40 companies, and regularly teaches courses on user interface software. Myers received a PhD in computer science at the University of Toronto. He received the MS and BSc degrees from MIT during which time he was a research intern at Xerox PARC. The current Pebbles project is investigating how hand-held computers will interoperate with other devices and computers. Funded by DARPA and industry, it has already generated 9 conference and journal publications, while creating about 20 applications, one of which is available commercially. His Amulet project included the Jade system that automatically created user interfaces from specifications.

Jodi Forlizzi is an Assistant Professor in the Human Computer Interaction Institute in the School of Computer Science and the School of Design at Carnegie Mellon University. Her research interests include experience-based design, types of user experience, and user-product interactions. She is interested in how computational devices can bring people information in new contexts. To this end, she is working on industrially funded research to identify interface design guidelines for hand-held and mobile devices. Forlizzi received her BFA in Illustration from the Philadelphia College of Art, and her Master in Design in Interaction Design from CMU. She publishes in the design, HCI and usability communities, and is a member of ACD, AIGA, UPA, and ACM SIG CHI.

References

1. Borning, A., Lin, R., and Marriott, K. "Constraints and the Web," in *Proceedings of the 1997 ACM Multimedia Conference.* 1997. pp. 173-182. http://www.cs.washington.edu/research/constraints/web/mm97.html.

2. Dalgic, I., *et al., True Number Portability and Advanced Call Screening in a SIP-Based IP Telephony System.* 3Com Corporation, http://www.3com.com/technology/tech_net/white_papers/503054.html, March 14, 2000.

3. Dean, R. and Belkind, R., *PHONECTL Protocol; Internet Engineering Task Force (IETF) draft standard.* 3Com, Rolling Meadows, IL, http://www.cs.columbia.edu/%7Ehgs/sip/drafts/draft-dean-phonectl-01.txt, Sept., 2000.

4. Foley, J.D., *et al.* "A Knowledge-Based User Interface Management System," in *Proceedings SIGCHI'88: Human Factors in Computing Systems.* 1988. Washington, D.C.: pp. 67-72.

5. Frank, M.R. and Foley, J.D. "Model-Based User Interface Design by Example and by Interview," in *Proceedings UIST'93: ACM SIGGRAPH Symposium on User Interface Software and Technology.* 1993. Atlanta, GA: pp. 129-137.

6. Haartsen, J., *et al.*, "Bluetooth: Vision, Goals, and Architecture." *ACM Mobile Computing and Communications Review*, 1998. **2**(4): pp. 38-45. Oct. www.bluetooth.com.

7. Harman-Kardon, *Take Control System Controller.* http://www.harmankardon.com/products/productlines/takecontrol.asp, 2000. Washington, D.C.

8. Kim, W.C. and Foley, J.D. "Providing High-level Control and Expert Assistance in the User Interface Presentation Design," in *Proceedings INTERCHI'93: Human Factors in Computing Systems.* 1993. Amsterdam, The Netherlands: pp. 430-437.

9. Myers, B.A., Lie, K.P.L., and Yang, B.-C.J. "Two-Handed Input Using a PDA And a Mouse," in *Proceedings CHI'2000: Human Factors in Computing Systems.* 2000. The Hague, The Netherlands: pp. 41-48.

10. Myers, B.A., *et al.*, "Using Hand-Held Devices and PCs Together." *ACM Communications of the ACM*, 2001.: pp. To appear.

11. Myers, B.A., *et al.* "Extending the Windows Desktop Interface With Connected Handheld Computers," in *4th USENIX Windows Systems Symposium.* 2000. Seattle, WA: pp. 79-88.

12. Myers, B.A., Stiel, H., and Gargiulo, R. "Collaboration Using Multiple PDAs Connected to a PC," in *Proceedings CSCW'98: ACM Conference on Computer-Supported Cooperative Work.* 1998. Seattle, WA: pp. 285-294.

13. Nielsen, J., *Usability Engineering.* 1993, Boston: Academic Press.

14. Olsen Jr., D.R. "A Programming Language Basis for User Interface Management," in *Proceedings SIGCHI'89: Human Factors in Computing Systems.* 1989. Austin, TX: pp. 171-176.

15. Pacific Neo-Tek, *OmniRemote.* http://www.pacificneotek.com, 2000.

16. Singh, G. and Green, M. "A High-Level User Interface Management System," in *Proceedings SIGCHI'89: Human Factors in Computing Systems.* 1989. Austin, TX: pp. 133-138.

17. Sun, *Jini Connection Technology.* Sun Microsystems, http://www.sun.com/jini/, 2000.

18. Szekely, P., Luo, P., and Neches, R. "Beyond Interface Builders: Model-Based Interface Tools," in *Proceedings INTERCHI'93: Human Factors in Computing Systems.* 1993. Amsterdam, The Netherlands: pp. 383-390.

19. Vander Zanden, B. and Myers, B.A. "Automatic, Look-and-Feel Independent Dialog Creation for Graphical User Interfaces," in *Proceedings SIGCHI'90: Human Factors in Computing Systems.* 1990. Seattle, WA: pp. 27-34.

20. WAP Forum, *Wireless Application Protocol Wireless Markup Language Specification: WAP WML, Version 1.2.* Wireless Application Protocol Forum, Ltd., http://www.wapforum.org/, November, 1999.