Tangible Windows

Jonathan Diehl
Media Computing Group
RWTH Aachen University
diehl(at)cs.rwth-aachen.de

Jan Borchers
Media Computing Group
RWTH Aachen University
borchers(at)cs.rwth-aachen.de

ABSTRACT
We are stuck in a world where our digital life is confined into virtual windows that clutter computer screens everywhere. Tangible Windows allow you to break out of this prison by pushing windows into the real world. Instead of interacting with your applications through virtual rectangles, you can hold them in your hands, embed them in your environment, and take them with you. Tangible Windows employ a new set of operations that allow applications to move freely between them. We have implemented a working prototype of a Tangible Windows Systems and evaluated it by comparing it with a common desktop computer in a user study. All participants commended the system and prefer it over the desktop computer to some extend.

Author Keywords
Ubiquitous Computing, Smart Environments, Tangible UIs, Windows, Handheld Devices, Mobile Computing, Interaction Design, User Studies

INTRODUCTION
Mark Weiser has predicted that "ubiquitous computing will gradually emerge as the dominant mode of computer access over the next 20 years" [24]. After almost 20 years, all devices that make up his vision have become widely available: high-resolution projection screens and large LCD panels are the boards; touch-sensitive portable computers, such as the Apple iPad, are the pads; mobile phones are the tabs. Yet, we are nowhere close to fulfilling Weiser's vision of an interconnected network of different devices that seamlessly work together to assist the users in their activities.

The main problem lies in the computing infrastructure limiting interactive system innovation through constrained possibilities, interjected abstractions, and unmediated interaction [7]. We believe that this is especially true for ubiquitous computing, because modern operating systems are based on assumptions that no longer hold and which are built into the functionality of the systems [15]. In a ubiquitous environment, there is no longer a single user working on a single computer with a defined set of input and output devices. Building ubiquitous applications on top of an operating system that is based on these assumptions, is extremely demanding.

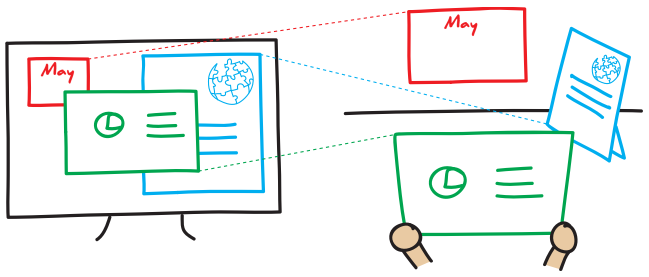

Figure 1. Tangible Windows are the physical equivalent of virtual windows on the desktop computer. They act as physical windows to the digital world, where applications can be moved freely among them.

In this paper, we introduce Tangible Windows, an alternative concept to today's monolithic computer systems. Tangible Windows are windows, taken from the desktop computer into the physical world as illustrated in figure 1. Tabs, pads, and boards, just like more recent technologies, such as tabletop or paper displays, represent different kinds of Tangible Windows. Tangible Windows always show only one application. Applications, however, are not bound to a single window. Instead, they can freely move between windows or be active on multiple windows at the same time. The user commands not only the application but also where on what window it should be displayed.

We demonstrate the feasibility of the Tangible Windows system by describing a prototype that we have implemented based entirely on today's technology. For this prototype, we describe a set of window operations that is suitable to control Tangible Windows and confirm in a survey that the set is useful and complete. Finally, we show in a user study that users can make effective use of Tangible Windows and prefer them to some extend over virtual windows.

RELATED WORK

The Vision of Ubiquitous Computing
In his seminal article [24], Mark Weiser juxtaposes computer technology with writing technology: while writing has disappeared in the human world, allowing us to use it invisibly to reach goals beyond producing text, computers have remained to be only windows to the virtual world, where we

have to focus on the interaction instead of our goals.

Ubiquitous computing aims at "drawing computers out of their electronic shells" [24]—both the data and the way to process it—to the physical world. To achieve this, the computer has to (1) be aware of where it is and behave accordingly, and (2) must be available in many sizes and numbers.

Weiser identified three fundamental sizes for ubiquitous computers: tabs, pads, and boards. Tabs are small, location-aware devices that can be attached to people or things. They act as identifiers, extensions of systems, or as personal information devices. There can be hundreds of tabs in a room. Pads, dubbed as "an antidote to windows" [24], are scrapbooks, which anyone can grab and use anywhere. In a typical work environment, ten or twenty pads can be spread around. Boards provide a space for sharing information. Interacting with boards ranges from close engagement, like writing, to just merely glancing at them.

To realize the vision, three major technologies are needed: (1) cheap and low-power devices, (2) wide-spread, high-bandwidth network coverage, and (3) supporting software. The first two are increasingly available today, but the supporting software that allows interaction among spread-out ubiquitous computers and tolerates continuous and substantial changes in the hardware configuration at run-time is still missing. This shortcoming has prevented ubiquitous computers from emerging on a large scale and, thus, made the computer remain all too visible.

Conversely, Reeves and Nass argue in their book that humans treat computers as social agents [17]. Consequently, humans are not only aware of computers as a tool but also, unconsciously, treat computers with politeness and give them personality.

Our work cannot bridge the gap of missing software support, but it provides a new approach to interacting with computers that allows the computer to be naturally embedded into the environment by the user.

Ubiquitous Computing Today

Infrastructure

Infrastructures provide support at the lowest level. They supply the fundamental functionality that eases development of ubiquitous applications.

Obje [8] allows the development of systems that can communicate with each other, even if they have only limited prior knowledge of each other. This infrastructure allows systems to gain compatibility to other systems at runtime, which allows computers to work together independent of their original intent.

[14] propose to interpret the pixel data of an application's user interface and map arbitrary input and output devices in the environment to the user interface. This way, existing applications can be controlled through controllers spread out in a ubiquitous environment.

Middleware

Middleware provide a comprehensive framework for developing ubiquitous applications. The middleware supply much of the special functionality required for ubiquitous systems but usually require the applications to be written in accordance with their rules and specifications.

Some examples of ubiquitous middleware enable applications to use multiple devices simultaneously and exploit adaptive resource management [18], employ distributed, migratable, and plastic user interfaces [2], compose services across multiple devices at runtime [19], or connect arbitrary resource providers with end-users through meta-UIs [23].

Developer Support

Developer support aims at assisting developers in creating or migrating their applications to ubiquitous environments by providing tools or processes tailored to address the typical problems of ubiquitous application development.

Malai [4] is a post-WIMP development environment that allows the developer to specify the user interface on an abstract level, such that it is modality independent. From this specification, an appropriate user interface can be generated for any supported platform. [20] introduce a refactoring process that allows existing user interfaces to be transformed such that they become adaptive to the system's modalities.

User Interface Migration and Adaptation

In model-based approaches, the functionality of an application is specified in a model, which allows large parts of its source code to be automatically generated. These approaches can ease the development of ubiquitous applications: The runtime behavior and user interface can be adapted from the model to fit changing hardware configuration.

[13] present a tool for creating model-based nomadic applications that can transition between different devices. [5] propose the use of executable models to allow user interface adaptation at runtime. Other examples of work towards automatic generation of user interface from model-based application are [16] and [22].

The user interfaces of web-applications are inherently model-driven and, thus, a prime candidate for user interface migration and adaptation. [3] provide a service to migrate web applications at runtime from one device to another. [10] allow users to select parts of a web application that they want to migrate to different devices.

The SUPPLE system [9], unlike previously mentioned approaches, considers user interface generation as an optimization problem, which does not impose any requirements on the application architecture. It renders the user interface by considering the device capabilities and a typical user trace, allowing it to focus on important functions and arrange the user interface accordingly.

Interaction Techniques and Systems

In [21], the authors present a comprehensive taxonomy of multi-person-display ecosystems. The dimensions of their taxonomy are the size of the system and the engagement of its users. They also present and discuss interaction techniques for binding multiple displays. The focus of our work is different than that of the projects analyzed in the taxonomy: we aim at creating a new and generic concept for interacting with multi-display multi-user systems, while these systems suggest concrete interaction techniques. Thus, Tangible Windows would not fit into the taxonomy, because they are neither limited in size nor in user engagement.

Interacting with Ubiquitous Computing Systems

[1] investigated how large display environments can be used for the task of sense-making. They show that users perform activities on the virtual screen that were previously done with physical artifacts. The large virtual space is used in two ways: for accessing external memory and for encoding special meaning into the spatial arrangement. They conclude that large arrays of high-resolution screens will change the way users work and think. This conclusion also applies for Tangible Windows, because their "screen space" is only limited by the size of the environment they are in.

[6] studied how people use an increasing number of personal devices together. They report that activities often span multiple devices, the role of each device is not fixed, users want to separate some activities from others, and techniques for accessing information vary across devices. The biggest challenge for users, they found, is to manage the information across different devices. To improve the matter, they suggest that system developers should focus on the user, not the application; make devices role-aware; provide simple information transfer mechanisms; and include trustworthy synchronization services. The design of Tangible Windows supports these recommendations, among others, through its simple information exchange by copying applications between windows.

In PaperWindows, [12] simulated the interaction with digital paper displays by projecting virtual windows onto sheets of paper. They introduce a set of operations for interacting with paper displays that are based on metaphors from physical paper. These operations allow the user to transition the content of the paper between different sheets of paper and manipulate it. Tangible Windows explore a different set of operations that are suited for controlling application rather than content.

TANGIBLE WINDOWS

Virtual windows on the desktop computer allow the user to interact with applications. Every application is contained in one or multiple windows that display the interface of the application. Typical virtual windows can be focused, repositioned, resized, opened, closed, and hidden. Through these operations, windows allow their users to arrange applications, switch between them, and work on multiple applications in parallel.

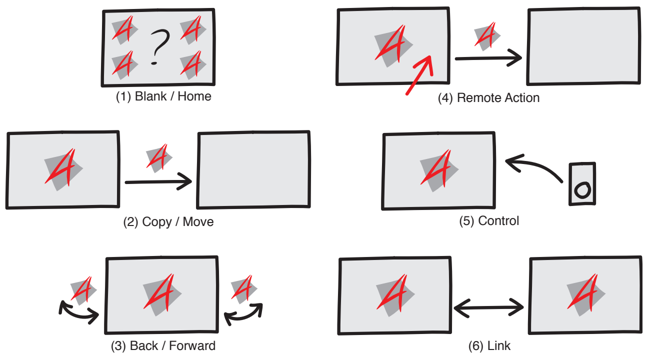

Figure 2. Set of standard operations for Tangible Windows: (1) close the active application or show the entry point to other applications; (2) copy or move an application from one Tangible Window to another; (3) navigate the application history of a Tangible Window; (4) perform an action on another Tangible Window; (5) create a specialized control interface on another Tangible Window that controls the application on the source window; (6) copy an application to another Tangible Window and link both, such that all user actions on either window are also done on the other

Tangible Windows, like virtual windows, allow the user to interact through them with applications. Every Tangible Window, like a virtual window, shows only one application and allows the user to interact with it through its own input channels on the application's interface. However, Tangible Windows are physical devices that users can pick up, carry around with them, and embed in their environments.

Because of their physicality, Tangible Windows require a different set of operations to be used effectively. Since every Tangible Window comes with its own input channels, user input can always be mapped to the appropriate application. Thus, a focus operation is not required. Repositioning and hiding are supported naturally, by repositioning or hiding the Tangible Window in the physical world. Resizing a Tangible Window is not feasible with today's technology. However, moving application between Tangible Windows with different form factors (e.g., tabs, pads, boards) presents an alternative approach to resizing. Opening and closing Tangible Windows is not possible, because physical devices cannot appear out of nowhere. Instead, existing Tangible Windows have to be reused to realize these operations. To change the active application on a Tangible Window, the previous application (e.g., a home application listing all installed applications) can open a new application, or the user can transfer an application from another Tangible Window to this one.

Operations

To be able to effectively use Tangible Windows, a new set of window operations must be designed. Below, we propose a set of operations for Tangible Windows, we have found most useful. The first two operations (Copy / Move and Blank / Home) are essential for every Tangible Windows system and must always be provided. The others form useful extensions to the basic set of operations. Figure 2 summarizes the operations.

Blank / Home
Blanking a Tangible Window removes the active application from the Tangible Window. This could be used to clear the screen and turn a device idle to preserve power.

The home operation, on the other hand, provides an entry point to the system. It activates a special application that acts as a gateway to other applications, e.g., by displaying a list of available applications, from which the user can choose the desired application. Upon user confirmation, the selected application is opened and becomes active on the screen. The home operation can be considered an alternative to the blank operation.

All operating systems that are based on applications must provide a means for the user to launch these applications. The home screen is a well-established solution for this purpose and suites the Tangible Windows very well.

Copy / Move
Copying an application from a one Tangible Window to another opens the application on the target Tangible Window and restores the application state from the source to the target. After the operation, both Tangible Windows display the same interface and data, but they act independent of one another, i.e., subsequent user input on either Tangible Window has no effect on the other.

Copying can be done in three different ways: the application can be copied directly from the source to the target (copy); the application can be copied directly from the target to the source (retrieve); the application from the source can be copied to a pasteboard and later pasted to on or multiple targets (copy/paste). This third way does not require a target operation.

Moving an application from one Tangible Window to another, can be realized as a copy operation, followed by an operation that blanks the source Tangible Window or removes the application in some other way (e.g., by showing a home screen).

Copying application is a concept that is unknown in the world of virtual windows because it is not needed: virtual windows can be created or dismissed arbitrarily, and there is no obvious advantage of having copies of existing virtual windows available. Tangible Windows, on the other hand, are a scarce resource that must be reused efficiently. Therefore, it is important that the user is in control of what runs where, which includes the ability to transition applications between the Tangible Windows.

Back / Forward
Moving back in the application history, opens the application that was active before the current application. Likewise, moving forward in the history reverts a back step and opens the application that was opened before performing the back step. When no back operation was performed, the forward operation is inoperative.

The Back and Forward operations help the user in two ways: they provide a way to recover from errors when using window operations, and they provide an additional navigation dimension when working with multiple applications on a single Tangible Window.

Remote Action
A remote action is a user-initiated action, where the effect of the action occurs on another Tangible Window. If the action changes the state of an application, the application is first copied to the target Tangible Window, where the action is performed and the effect is shown (e.g., open a web link in a new window). If the action opens a new application, the application is opened on the target Tangible Window. The source Tangible Window is unaffected in all cases.

In virtual windows, opening applications and links in new windows is a core functionality. It allows the user to postpone something until later or to start a new, parallel interaction path. The remote action mirrors this functionality in the world of Tangible Windows.

Control
By controlling an application from another Tangible Window, a specialized interface that can be used to remote control the connected application (e.g., playback functions for a movie player) is created. All actions that are performed on the control interface are sent to the connected application and processed there. The specialized control interface could be designed by the application author or generated through user interface generation tools, such as [9].

Many applications make use of multiple virtual windows, on which they distribute different controls or other aspects of the application. This can greatly benefit the clearness of the user interface. Consequently, the interface of complex applications on Tangible Windows should be equally distributed among different windows. The control operation provides the means to set up the windows for this purpose.

Link
The link operation creates a copy of the application on the target Tangible Window and links both applications, such that all actions performed on either Tangible Window are mirrored on the other. Virtual windows systems with multiple screens provide a similar feature, where two screens can always show exactly the same.

Linking is especially useful for collaborative scenarios, where spectators need to follow a presenter, or multiple users need to control the same application together.

Target
Several of these operations require a target to be executed: Copy, Move, Remote Action, Control, Link. Targeting a Tangible Windows selects the Tangible Window as a target for such an operation.

Targeting can be done before or during the execution of another operation, which allows the operation to be performed

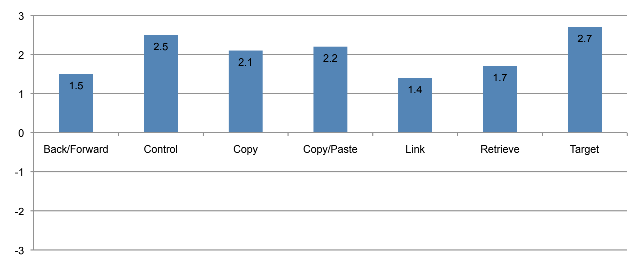

Figure 3. Rating of the operations from is very useful (+3) to not useful at all (-3)

immediately on the target. However, targeting can also be done after the execution of the other operation, which will defer the execution of the operation until the target is selected. Depending on the situation, either way of targeting might be more appropriate, which is why we suggest that both modes should be supported.

Discussion

We have evaluated the proposed operations, with the exception of the Blank/Home operation, in a user study, where we asked ten participants to rate the usefulness of each operation in a general sense. We skipped the evaluation of the Home operation, because it is already an well-established operation in many mobile operating systems. The results of the study are shown in figure 3. On average, all gestures were rated useful (Mean=2.01).

When asked about additional operations that users found useful for Tangible Windows, the following operations were mentioned:

- remove the connection between windows, i.e., linking or control
- copy more than a single application to the pasteboard
- view an overview of the states and connections of all windows

Removing the connection between windows should be realized through the Blank / Home operation. This operation should always restore a safe initial state with no connections in place. Multiple pasteboards software for the classic pasteboard is already available (e.g., for Mac OS X[1]). This concept could be ported to the Tangible Windows pasteboard. An overview of the complete configuration would be beneficial for a complex setup of Tangible Windows and should be considered in the future.

PROTOTYPE IMPLEMENTATION

Infrastructure

The system consists of a central server, running on a MacPro desktop computer, and several clients, running on Apple iPads. All clients are connected to the server through a wireless network. The server runs the Tangible Windows Server application (see below) and a web server, where the web appli-

[1]`http://pth.com/products/pthpasteboard`

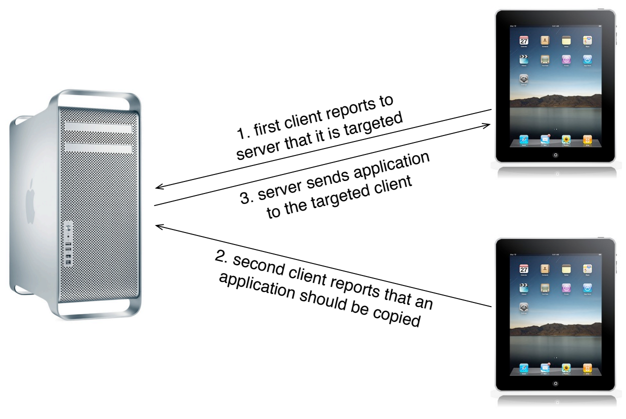
1. first client reports to server that it is targeted
3. server sends application to the targeted client
2. second client reports that an application should be copied

Figure 4. The communication protocol during a copy operation. All clients report their state to the server, which decides where to relay operations.

cations that are used by the clients are hosted. The clients all run the Tangible Window Client application (see below), which embeds multiple web applications. We could not rely on the iPads ability to switch between different applications, because, due to the limitations of the operating system, we would have lost the connection to server during each switch. Figure 4 illustrates the communication protocol for the example of copying an application.

This infrastructure is somewhat limiting, in that it requires a central server and, consequently, is not suitable for ad-hoc situations. We have decided to include a server for two reasons: (1) simple development; (2) good traceability of the activity of all clients. For future versions, we plan to migrate the system to a peer-to-peer system, where no central server is needed any more.

Tangible Window Server

The server is implemented as a Mac OS X application. Its main purpose is to keep track of all connected clients and relay operations between them. For deferred operations, it also provides a pasteboard, where the previous operation is stored.

Tangible Window Client

The client is implemented as an iOS application. After connecting to the server, it displays the home screen (a list with icons of all provided applications), and a bar on the side with buttons for the implemented window operations (copy, retrieve, link, back, forward, home, target). Figure 5 shows a screenshot the client.

By tapping on an icon of the home screen, the respective application is opened and shown instead of the home screen. The operations bar will always remain visible. All applications, with the exception of the Maps application, are realized as web applications that show realistic imitations of real applications designed for the Apple iPad. The Maps application is implemented using the native MapView to allow for better performance. Of the web applications, only the calen-

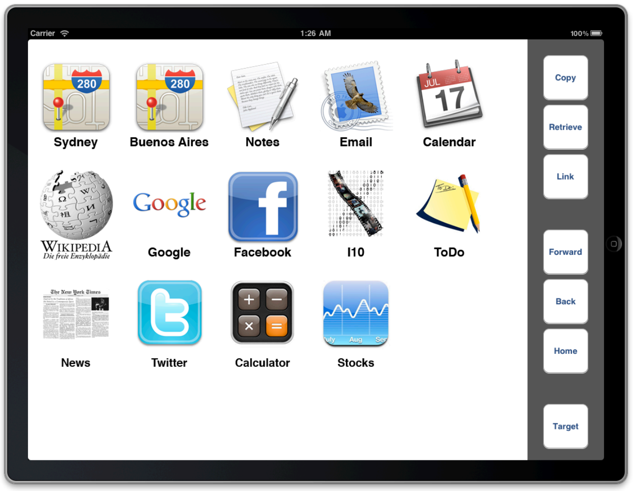

Figure 5. A screenshot of the iPad client's home screen. Operations are executed by tapping the appropriate buttons on the right. The button on the lower right is held to mark the client is the current target.

dar and the email application were interactive with limited functionality.

By tapping on a window operation, the operation and the current state of the application is sent to the server. The server can then decide, which client should respond to the operation, and relay the operation to the appropriate client. The client, then, executes the operation, it received from the server.

Since most applications are simple web applications, the state of the application is already encoded in the URL of the website. Therefore, we used standard URL requests to encode our application state. For the non-web-based applications, we introduced new URL schemes (home:// and maps://). In addition, for the maps application, we encode the visible region of the map and any markers that are selected in the URL body.

EVALUATION

Using the prototype, we have described above, we have evaluated the Tangible Windows concept in three situations:

1. We asked users to set up an imaginative workspace with Tangible Windows to observe how users would arrange Tangible Windows and what applications they would consider useful running on these windows in the background.
2. We asked users to fulfill a simple planning task, where they had to select 4 locations from a map depending on associated information from an external reference (Wikipedia). This task was simple enough to be easily performed in a short time (10 minutes) but challenging enough to benefit from having multiple windows on both a desktop setting as well as using Tangible Windows.
3. We interrupted users during their work to test what effect Tangible Windows can have on the user behavior when addressing the interruption and when recovering from it.

Study Description

14 users participated in the study: 13 students of various disciplines and 1 architect. 12 of the 14 users were male, and 2 were female. The average age of the users was 24.5 years.

All user tests were recorded with two cameras: one that captured the whole scene with the user and the other that captured only the table with the Tangible Windows from above. In addition, the on-screen interactions on the reference system were recorded using a screen cast. Before the test, each user watched two introductory videos: the first explained Tangible Windows and the operations explained in section ; the second explained what applications are available and how to open web links in a new browser window or tab on the reference system.

The first situation was tested only on the Tangible Windows system, because virtual windows cannot be distributed in the environment of the user. The second and the third situation were tested on both the Tangible Windows system and a reference system to be able to directly compare results.

The test system consisted of 6 Apple iPads running the Tangible Windows client. The users had access to the following applications: Maps, Notes, Email, Calendar, Wikipedia, Google search, Facebook, ToDo list, News, Twitter, Calculator, Stocks. However, only Maps, Notes, Wikipedia, and Google search were fully functional. The other applications were mockups that only provided the functionality that was required for the user test. All Tangible Window clients supported the following operations: copy, home, back, forward, target.

The reference system consisted of an Apple MacPro desktop computer with a 23" Apple Cinema display. Users could use any of the following applications: Safari (web browser), Apple Mail (email client), iCal (calendar), and TextEdit (text editor).

Workspace Setup

In the workspace setup test, we have asked users to set up the room with Tangible Windows, as they would imagine setting up their future office. The users had two tables in front of them, where they could place the Tangible Windows, and a construction alongside the walls, where they could hang them. All Tangible Windows were in cases that allowed them to be laid down flat, in a slightly tilted position, or in a standing position.

We asked the test participants to open any applications they found useful and distribute them in the room. After the test, we interviewed the users about the arrangement of Tangible Windows and about other application they would have also considered useful but that were unavailable on the prototype.

Planning Task

In the planning task test, we asked users to plan a city tour in either Sydney (Tangible Windows) or Buenos Aires (reference). We ensured that users had no knowledge of either

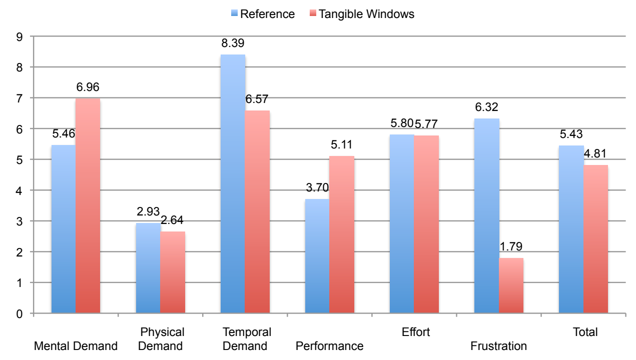

Figure 6. Comparison of the self-rated user experience of the Tangible Windows system and a reference Mac OS X desktop system for the task of planning a city tour with an interruption.

city. The users were supposed to select 4 sights out of a list of 30 sights, they wanted to visit, and order them by their preference. The list of sights was represented by markers in the Maps application (Tangible Windows) or Google Maps (reference). Each marker showed the name of the sight and a link to a Wikipedia article about the sight.

After the test, we discussed the employed strategies with the user to determine the underlying reasons and motivations.

Interruption

Users were informed before the test that, if they received an email, they should stop working on the task and respond to it immediately. This email was sent to every user some time during the planning task. In the email, the user was requested to determine the postal code of a distant city, and to find an appointment on the calendar. The postal code could be found from the Google search application, the calendar appointment could be found from the calendar application by browsing to the following week.

After the test, we discussed the interruption with the user, especially what windows they used to solve the task and what they did after the interruption to resume their previous work.

Questionnaire

All users were asked to fill out a questionnaire about their experience with the reference and the test system. This questionnaire was designed in accordance with the NASA TLX questionnaire [11] and inquired the users mental demand, physical demand, temporal demand, performance, effort, and frustration.

Results

All test participants commended the Tangible Windows system and reported on average significantly lower frustration levels while using them compared to virtual windows on the desktop.

Figure 6 shows the mean of weighted workload from NASA TLX questionnaire. The frustration is significantly lower in Tangible Windows (Median=1.5) than the reference set

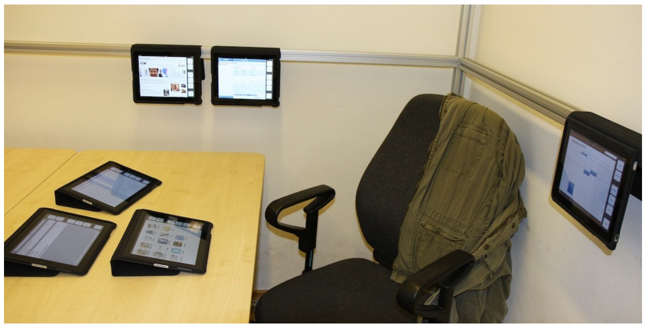

Figure 7. User setup of the workspace. The Tangible Windows on the desk show the Todo list, Mail, and the home screen; the Tangible Windows on the wall show News, Stocks, and the calendar.

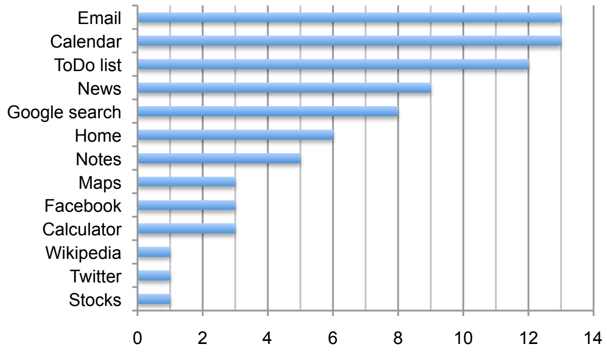

Figure 8. Number of users that opened one of the following applications somewhere in the environment of the workspace during the workspace setup task.

(Median=6.0). Wilcoxon ranked-sum test shows that the result is significant with medium effect size ($Z = 2.073, p < .05, r = 0.392$).

Workspace Setup

Users were very diverse in the arrangement of their virtual offices. Some users hung all Tangible Windows on the walls, while others arranged them neatly on the tables. Most users, however, did a mixture of both. Figure 7 shows such a mixed user setup.

All but one user had the Email and the Calendar open somewhere in their environment. 12 out of 14 users had either the Google search or the Home screen reachable, for quick entry into their system. 5 people had the notes application open on their desk to serve as a scratch pad. Two users dedicated one pad for personal entertainment and put it on the other side of the room to avoid too much distraction. Figure 8 shows how many users had each application open.

In addition to the provided applications, users wished for the following applications to be open in the background in their environment:

- instant messenger, Skype, phone
- television, movie player, music player

- dictionary
- favorite websites
- clock
- photo gallery
- bus timetable

Users also told us about applications they thought would benefit from having multiple Tangible Windows:

- spreadsheet, show different sheets or views of the data in different windows
- slides authoring, show different parts of the slideshow in different windows
- video editing, place clips in different windows to rearrange and sort
- photo viewer, each window to show a photo
- social video network, where videos are distributed across different windows
- tutorials to be viewed alongside the application they teach
- integrated development environment
- musical instruments that combine multiple windows to a single instrument (full-scale piano)
- document viewer, one document per window
- games, especially 3D games with different viewpoints visible on different windows
- phone and messaging, where it is possible to send typed messages on one window, while talking (oder video-calling) on another
- drawing application, where different parts of the sketch are distributed among different windows

Overall, the users rated the usefulness of having Tangible Windows showing applications placed in their environment very high (on average 6.57 out of 7.0).

Planning Task
All users kept copies of either the Map showing the sight or a Wikipedia article about the sight on a spare Tangible Window to remember their selection. 8 out of 12 users arranged these copies by positioning the Tangible Windows in their order of preference on the desk. One user, stacked the tablets on top of each other to get them out of the way. Figure 9 shows an example arrangement of Tangible Windows after the planning task.

Interruption
To cope with the interruption, every user took one or two fresh Tangible Windows and limited their research to these windows. After the interruption was done, the windows were placed aside and the original task was resumed with very low effort. One user explicitly said that the arrangement of Tangible Windows helped her understand where she left off her previous work.

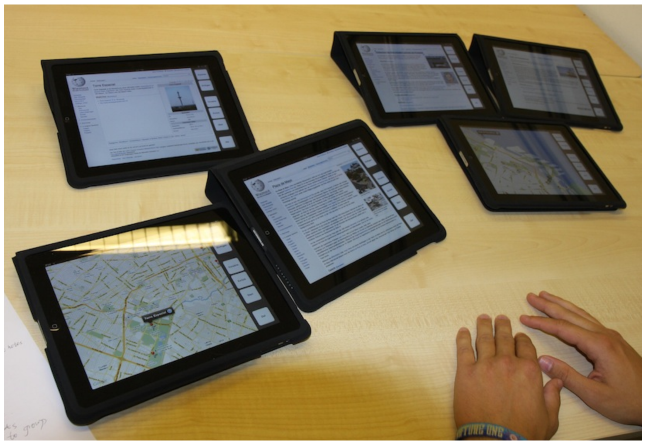

Figure 9. Arrangement of Tangible Windows for the planning task. Both groups of 3 Tangible Windows represent 2 sights, which are both shown in the opened Wikipedia articles. The markers on the map could be identified using the back and forward operation.

CONCLUSION & FUTURE WORK
We have introduced a new concept for interacting with ubiquitous applications called Tangible Windows. Users interact with desktop applications through virtual windows that can be arranged on the computer screen. Tangible Windows are virtual windows brought into the physical world. They allow users to interact with applications through them and arrange them in the real world. We have developed a prototype of a Tangible Windows system that supports 5 window operations: home, copy, remote action, control, and link. We have tested these operations and the general experience of working with tangible windows in a user study. The results show that users prefer working with Tangible Windows over working on a reference system using virtual windows.

In the future, we plan to improve our Tangible Windows prototype in three ways:

1. We want to remove the server and convert the communication architecture to a peer-to-peer architecture to better support ad-hoc usage of Tangible Windows.

2. We want to implement more applications and test their long-term usage.

3. We want to evaluate the use of Tangible Windows in a collaborative setting.

For a Tangible Window system to be deployed on a larger scale, there are still some key technologies missing:

- Applications must be able to encode and restore their complete state during runtime. Currently, there is no support for the developer to implement this functionality and there are no means to do so automatically, except for very specialized classes of applications.

- User data must be accessible uniformly from all Tangible Window devices. Storing user data on each device is unfeasible considering Weiser's vision that pads should not be personalized but rather a service provided by the environment.

REFERENCES

1. C. Andrews, A. Endert, and C. North. Space to think: large high-resolution displays for sensemaking. *Proc. CHI 2010*, pages 55–64, 2010.
2. L. Balme, A. Demeure, N. Barralon, J. Coutaz, and G. Calvary. CAMELEON-RT: A Software Architecture Reference Model for Distributed, Migratable, and Plastic User Interfaces. 3295:291–302, 2004.
3. R. Bandelloni and F. Paternò. Flexible interface migration. *Proc. IUI 2004*, page 148, 2004.
4. A. Blouin and O. Beaudoux. Improving modularity and usability of interactive systems with Malai. *Proc. EICS 2010*, pages 115–124, 2010.
5. M. Blumendorf, G. Lehmann, and S. Albayrak. Bridging models and systems at runtime to build adaptive user interfaces. In *Proc. EICS 2009*, pages 9–18, 2010.
6. D. Dearman and J. S. Pierce. It's on my other computer!: computing with multiple devices. *Proc. CHI 2008*, pages 767–776, 2008.
7. W. K. Edwards, M. W. Newman, and E. S. Poole. The infrastructure problem in HCI. *Proc. CHI 2010*, pages 423–432, 2010.
8. W. K. Edwards, M. W. Newman, J. Z. Sedivy, and T. F. Smith. Experiences with recombinant computing: Exploring ad hoc interoperability in evolving digital networks. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 16(1), 2009.
9. K. Gajos and D. S. Weld. SUPPLE : Automatically Generating User Interfaces. *Information Systems*, pages 93–100, 2009.
10. G. Ghiani, F. Paterno, and C. Santoro. On-demand Cross-Device Interface Components Migration. In *Proc. Mobile HCI 2010*, pages 299–307, 2010.
11. S. G. Hart and L. E. Staveland. *Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research*, pages 239–250. North Holland Press., Amsterdam, 1988.
12. D. Holman, R. Vertegaal, M. Altosaar, C. Kl, N. Troje, and D. Johns. PaperWindows : Interaction Techniques for Digital Paper. In *Proc. CHI 2005*, pages 591–599, 2005.
13. G. Mori, F. Paternò, and C. Santoro. Tool support for designing nomadic applications. *Proc. IUI 2003*, page 141, 2003.
14. T. Nakajima. How to reuse exisiting interactive applications in ubiquitous computing environments? *Proc. SAC 2006*, page 1127, 2006.
15. D. Olsen. Evaluating user interface systems research. In *Proc. UIST 2007*, 2007.
16. D. Raneburger. Interactive model driven graphical user interface generation. *Proc. EICS 2010*, pages 321–324, 2010.
17. B. Reeves and C. Nass. *The Media Equation: How People Treat Computers, Television, and New Media like Real People and Places*. Cambridge University Press, 1996.
18. M. Román and R. H. Campbell. A middleware-based application framework for active space applications. *Middleware Conference*, pages 433–454, 2003.
19. S. Sathish and C. D. Flora. Supporting smart space infrastructures: a dynamic context-model composition framework. *ACM International Conference Proceeding Series; Vol. 329*, 2007.
20. A. Savidis and C. Stephanidis. Software refactoring process for adaptive user-interface composition. *Proc. EICS 2010*, pages 19–28, 2010.
21. L. Terrenghi, A. Quigley, and A. Dix. A taxonomy for and analysis of multi-person-display ecosystems. In *Personal and Ubiquitous Computing*, volume 13, page 583, 2009.
22. V. Tran. UI generation from task, domain and user models: the DB-USE approach. *Proc. EICS 2010*, pages 353–356, 2010.
23. G. Vanderhulst, D. Schreiber, K. Luyten, M. Muhlhauser, and K. Coninx. Edit, inspect and connect your surroundings: a reference framework for meta-UIs. *Proc. EICS 2010*, pages 167–176, 2009.
24. M. Weiser. The computer for the 21st century. *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(3):3, 1991.