Designing an easy-to-use executive conference room system

Maribeth Back, Gene Golovchinsky,
Pernilla Qvarfordt, John Boreczky, Tony
Dunnigan, Scott Carter, William van Melle

FX Palo Alto Laboratory
Palo Alto, CA USA
back@fxpal.com

Abstract

The Usable Smart Environment project (USE) aims at designing easy-to-use, highly functional next-generation conference rooms. Our first design prototype focuses on creating a "no wizards" room for an American executive; that is, a room the executive could walk into and use by himself, without help from a technologist. A key idea in the USE framework is that customization is one of the best ways to create a smooth user experience. Since the system needs to fit both with the personal leadership style of the executive and the corporation's meeting culture, we began the design process by exploring the work flow in and around meetings attended by the executive.

Based on our work flow analysis and the scenarios we developed from it, USE developed a flexible, extensible architecture specifically designed to enhance ease of use in smart environment technologies. The architecture allows customization and personalization of smart environments for particular people and groups, types of work, and specific physical spaces. The first USE room was designed for FXPAL's executive "Ian" and installed in Niji, a small executive conference room at FXPAL.

The room Niji currently contains two large interactive whiteboards for projection of presentation material, for annotations using a digital whiteboard, or for teleconferencing; a Tandberg teleconferencing system; an RFID authentication plus biometric identification system; printing via network; a PDA-based simple controller, and a tabletop touch-screen console. The console is used for the USE room control interface, which controls and switches between all of the equipment mentioned above.

Introduction

As conference rooms add functionality, they often lose usability. A standard conference room is often quite literally crammed full of equipment: multiple displays, teleconferencing systems, meeting capture systems, and room controls (light, sound, screens, projectors) (Foote, 2004; Fox, 2000; Streitz, 1998). In one of the conference rooms at FXPAL, for example, there are four computers, three projectors, and a videoconferencing system, along with three SmartBoards (interactive whiteboards). Each of these items is controlled via its own unique interface (often a remote with dozens of small buttons) and none of these devices are interconnected (except at second hand, via the net). Using this room often requires the help of a technology expert – a "wizard" – who specializes in conference room systems.

The goal of the Usable Smart Environments (USE) project is to allow a user to make use of all this technology without having to pay any special attention to it. We based our design on observations, workflow studies and use scenarios for the executive's use of the room, and concluded that what was really needed was one big button that would "do the right thing" at any given point. That proved both impractical and undesired; however, extending the simplicity of this idea led us to a workable solution. In this chapter we detail the discovery and design process for

the USE ConsoleUI (a tabletop touchscreen kiosk) and its underlying system, which is our solution to the room and data control problem.

The USE control software has a number of web services that manage a collection of devices and applications that populate a meeting room. The control system that sits on top of these web services is designed to allow the user to select devices (e.g., video conference) or applications (e.g., whiteboard) as appropriate during a meeting. The software is controlled in the room through a console. The design goal of the console UI was an easy-to-use user interface that enables contextually appropriate interaction with the functions of the room. As is often the case, apparent simplicity often means enormous care and much work has been implemented "behind the scenes" to enable ease of use and a wizard-free conference room.

The current system allows customization and personalization of smart environments for particular people and groups, types of work, and specific physical spaces. The system architecture consists of a database of devices with attributes, rooms and meetings that implements a prototype-instance inheritance mechanism through which contextual information (e.g. IP addresses, application settings, phone numbers for teleconferencing systems, etc.) can be associated with specific devices for specific users, rooms and meetings.

Needfinding: an executive's workflow

As our goal was to fully understand an executive's use of meetings rooms, our approach was to investigate not only what happens during a meeting but also the general work process of an executive and how he and his staff prepared for the different types of meetings he attended. In particular, we explored the meeting practices of an executive of a research laboratory with 50 employees in Silicon Valley that is a subsidiary to a Japanese international company. We made direct observations, over a two-month period, of the executive's activities before, during, and after meetings, and tracked and analyzed the paper-based and computer-based scheduling and document handling systems involved. We also conducted interviews based on our observations with the executive, Ian, and with his support staff: an administrative assistant, Erica, and a Japanese market researcher, Mari, who also serves as Ian's personal assistant during travels to Japan.

Types of meetings within the company

Ian participates in a variety of types of meetings during a week: face to face informal meetings (sometimes including lunches and dinners), formal or semi-formal meetings with explicit or implicit agendas, video conferences, and occasionally offsite meetings. About 70% of the meetings are face-to-face informal meetings; 10-20% of the meetings are face-to-face formal or semi-formal meetings. Ian also frequently attends video conferences. When a video conference cannot deliver the appropriate level of personal contact, Ian goes to Japan to meet with executives and managers in the parent company; this happens a few times a year The least common type of meetings is offsite meetings. In the past, Ian has taken his management team for off-site meetings every year or two.

Still, the large majority of the meetings take place at the company's facilities. Ian has space for meetings in his office, although this space does not have any projection surface for sharing digital content. Adjacent to Ian's office is the conference room Niji. This room is used when Ian meets with a larger group of people, more than five or six, or if anyone in the meeting needs to use the projector for presentations or demonstrations. We identified this room as the space in which to develop our system.

Before the USE project began, Niji had an analog whiteboard, a data projector/screen for laptop hookup capability, a network connection, and a telephone conferencing system (audio only).

The most common meeting: informal face-to-face

Ian meets with researchers, consultants and managers in his office for informal face-to-face meetings. These meetings often have a stated purpose, such as discussion about the researchers' projects, or reviewing a presentation for a Japanese executive visiting the company. Some attendees bring their laptops to show a presentation or to make a demonstration. In order to get connected, many use the wireless internet connection while others plug in their computer to wired connections available for this purpose. However, since the word is out in the company that Ian likes to get a copy of the presentation on paper to take notes, many choose to give the presentation from paper rather from their laptops. Ian does not keep the annotations he makes on the printouts of the presentations; instead he gives them back to the presenter as a form of feedback.

The preparation for the face to face meeting mainly falls on Ian's assistant, Erica. She books the meeting and informs the attendee about the stated purpose. Depending on the purpose of the meeting the attendee prepares supporting documents. After the meeting, the task of the attendee is to implement eventual action items discussed.

The recurring semiformal meeting

Ian attends a number of more or less formal meetings during a week. The degree of formality shifts slightly between the meetings. For example, the Tuesday morning staff meeting which all employees are expected to attend has a pre-defined agenda that is followed throughout the meeting. This meeting is held in Kumo, a larger conference room across the hall. Other meetings have a more informal procedure that has developed over the years. Since these meetings have an implicit agenda rather than a pre-defined one, we call them semiformal. These semiformal meetings often have a few more attendees so they are often held in the conference room Niji rather than Ian's office.

Video conferences

Ian increasingly attends video conference meetings. Nearly all of his video conferences are with Japan, usually with executives at the Japanese parent company. Ian has two options for video conferencing; either he can use the one-person Tandberg video conference system on his desk or a larger system set up in a dedicated room near his office. Ian rarely uses any other documents than his handwritten notes during video conferences, although he thinks it would be nice to have supporting documents, such as emails and PowerPoint, in the meeting.

When the meeting has more than one attendee on Ian's side, another room, Yuki, is used. This room also uses a Tandberg video conference system in addition to an in-house system for sharing and annotating PowerPoint slides in distributed meetings. This system allows the people in the remote locations to see the slides in real time. When Ian needs to use this system, it is set up by a member of the support staff. Ian rarely has any documents prepared for these meetings. If PowerPoint or other documents are shared, other meeting attendees on his side or on the Japanese side provide them. Usually the documents have been shared between the meeting participants before the meeting starts, however, the shared version may not be the version used in the meeting.

Observations

Lack of technological support for everyday tasks

One of the most salient observations from this interview study is Ian's lack of technological support for his everyday tasks. Ian mainly uses the computer to read his email; the rest of his work is captured in the notes he makes in his calendar. The low use of technology can be explained by two factors; personal preferences and the essential nature of a CEO's work.

Over the years, Ian has developed methods and techniques to get his work done, which reflect his personal management style. Part of his style is to externalize his ideas as lists on paper, either in his calendar or from his notepad. The calendar provides him with a quick overview of important issues. It easily opens flat to the current week; it is lightweight and Ian can easily take it with him to meetings. A few notes on the pages remind Ian of the issues at hand. The characteristics of the calendar let Ian focus on what needs to be focused on in the meeting: both the issues that need to be solved or communicated, and the attendees of the meeting.

The technological tools that Ian uses reflect two essential parts of a CEO's work; communication and relationship building. This is particularly true in meetings. Besides the important information sharing and decision-making functions, the meetings are also an arena for building and maintaining relations with people critical for the success of the company. It is through communication and relationship building that the CEO conducts his leadership. The three computer-based tools that Ian uses, email, PowerPoint, and video conferencing, all play important roles in communication and relation building.

The technological tools Ian uses today support his way of working. However, the question is how to make the technology useful for him, so that he will use it. The key issues seem to be not only whether the new technology continues to support Ian's work practice, but if the technology can provide a tangible benefit for Ian's work process as well as the work process of his support staff. In addition, the technology must not intrude on the most important aspect of the meeting as a place for creating and maintaining relations. Technology introduced in the meeting room should be as unobtrusive as possible, melting into the background, yet easily accessible to allow spontaneous use by the attendees without interrupting workflow – that is, it should feel as though it is part of the basic infrastructure of the room (Star, 1999).

The persistence of notes and products

The meetings produce different kinds of products, with varying degrees of persistence and reusability. Some are notes in the attendees' personal notebooks or calendars. The meeting can also have more formal products, such as minutes. A third kind of product is annotations to presentations given during the meeting. In the setting we studied, the number of products was limited. The semiformal meeting produced the most, although generally only one product came out of the meeting: the minutes. On occasion, meetings were followed up by email communication. There is a relation between the kind of meetings and the kind of products produced during the meeting. The semiformal meeting tends to produce formal products, while the informal meeting produces annotations on presentation printouts (if any products are produced at all). Production of personal notes can occur in any kind of meeting.

The long-term usefulness of the products and the notes in the everyday work of the CEO varies. The (physical, paper-based) minutes, for example, are kept on file, but are rarely accessed. The reason might be that ongoing activities and future events require more attention than decisions about past activities and events. This fact is also reflected in how the CEO refers back to the notes in his calendar. These notes are often used – and more are created -- during an activity or during the planning of an event. After the event is past or the activity completed, the notes are no longer used. The lifespan of the notes is thus limited to the life span of the activity they support.

A meeting product not discussed is notes on the whiteboards in the meeting room. Whiteboard notes are the most uncommon type of product for Ian (though not for his employees). When we interviewed the CEO's administrative assistant, we noticed that Ian did have some notes on his whiteboard. Upon inquiry, however, it turned out these were about three months old and were not produced by the CEO. The main reason the notes were still there had less to do with their long term usefulness and more to do with that the whiteboard had not been in use since the notes were written down. Since the whiteboard has doors covering it, the notes could stay on the whiteboard without negatively influencing the overall appearance of the CEO's room.

PowerPoint presentations, when used within the company, are living documents. Although care is taken by the researchers to prepare them, notes and comments get added to them during the meetings with Ian and changes are made to the presentation of the meeting. Most of the presentations given to Ian are work in progress, such as research updates, presentation of research ideas and drafts of high profile presentations to executives of the parent company. Similarly, Ian's presentations are often works in progress as a step for preparing his Japan trips. The work in progress character of the presentation affects the sharing of the material. In particular Ian's presentations are confidential. He does not like people to get their hands on unfinished material, since he may still be working on what to present and how to present particular issues. For the researchers, the presentations generally do not have the same level of confidentiality.

Observations and discussion: the successful introduction of technology

From the execution and work process of the CEO's meetings we can draw several conclusions:

- The workflow this CEO has developed over the years includes very little technological support. When technology is needed, the primary users of it are his support staff.

- The meetings have few products. The semiformal meetings produce more products than other meeting types. These products mainly get archived to a location only known by the CEO's administrative assistant. Some of the products are returned to the original producer with added annotations.

- PowerPoint presentations are ubiquitous. They not only are shared from a computer, but very often communicated as printouts. The presenter most often has lower status in the organization than the person receiving the presentation. For example, researchers and managers present to Ian, while Ian presents to the executives he reports to in Japan.

- Presentation and material shown and discussed during a meeting may be confidential. Copies of confidential materials are disposed of after the meeting.

Introducing new technology to support the meeting culture of the CEO is challenging, in particular since the meeting culture has evolved without any significant amount of technology. This section summarizes in our view the most important factors for a successful introduction of new technology in the meeting room.

- The technology needs to fit with the current work practice. Although some changes to the current work are probably needed, these changes should be as minimal as possible and should fit into the current work practice as seamlessly as possible (Weiser, 1991). Any changes that are perceived as unnecessary or as added work compared to the current work practice may cause the user not to use the new technology.

- The technology needs to be perceived as lightweight. Ian uses lightweight tools today: pencil, eraser and paper. These tools are easy and quick to use and provides a tangible result of the actions taken. The new technology should share these characteristics. There should be

no need for figuring out how to use the technology, and the results of the user's actions should be immediately visible. In a meeting there is no time for figuring out how the technology works. In addition, the user risks losing face by showing a lack of understanding of the technology in the meeting, making adoption less likely (Preece, Rogers, & Sharp, 2002). Therefore, preserving meeting flow is both technically and personally important.

- The technology should work in the background so people in the meeting can focus on the participants of the meeting and not on the technology. For example, there should be no need for troubleshooting presentations. The technology should also be immediately accessible in the same way a whiteboard is immediately accessible when walking into a conference room.

- The technology needs to be developed with a holistic approach where the focus is not only on what is going on in the meeting room, but also on the activities preceding and following the meeting. Extra work the user needs to do while creating or transferring the presentation to a meeting room may influence how often he or she uses the available technology.

USE case scenarios: Implications for design and implementation

Based on findings from the workflow study above, we developed six use case scenarios for a USE conference room that fits into the work flow of the executive. This section gives examples of one of the scenarios and our exploration of different solutions to the scenarios and some implications of the design of the user interfaces and the system architecture.

Scenarios and roles: task-level definition, transition points

The example scenario describes how the CEO, Ian, prepares and gives a presentation for his staff. Although we created several scenarios to match the workflow task discoveries, we describe only one here due to space constraints. This scenario was not common, only occurring around six-seven times a year, but it highlights the constraints of designing for Ian and the different aspects of his work process that the USE system we were designing needed to support.

In developing the scenarios, we first wrote a rich description to match with the current work process, in order to introduce changes only where it was technologically necessary. We then broke down the scenarios in table format. The first (Table 1) describes each step of the process and different design alternatives. The second table (Table 2) describes the order in which actions needed to be performed, and the responsibilities of the users and the system.

Scenario 1: Before the meeting

Ian needs to have a meeting debriefing six lab members about a presentation he is to give to the CEO of the parent company. Ian wants to have input on the presentation before he finalizes it.

Ian asks Erica to schedule the meeting. He tells her who he wants to attend and what the purpose of the meeting is. Erica schedules the meeting into Niji, since there will be more than six people in the meeting and since Ian wants to show slides. When Erica reserves Niji, she records the time and date of the meeting, and a title of the meeting.

To prepare his draft, Ian asks Erica to print out several older presentations which each have some of the content he'd like to show. He uses these as well as blank sheets of paper to design his presentation. He also meets with a few people in his office to gather ideas again on paper, by hand. In addition, he asks some researchers to send him their slides on a particular topic.

Erica authors the presentation in PowerPoint from the sheets of paper and prints Ian gives her. When she is done she prints the presentation and gives it to Ian to check over. This cycle repeats a few times, until Ian is satisfied with the content of the presentation. Erica prints a final version for him to mull over the day before the meeting. These tasks can be performed also by Mari, the other assistant; however the process is basically the same. Erica uploads the presentation so that it is accessible from the equipment in the conference room.

Scenario 1: Day of the meeting

Ian tells Erica in the morning that he wants some changes to the presentation before the meeting. She makes them and uploads it again. In addition, she also prints out a final clean copy on paper for Ian to take into the meeting.

Up to this point, we did not envision any major changes to the work processes when introducing the USE system. But when Ian enters the room, we envisioned four different alternatives for the sign in process. The main difference between these first four alternatives is which devices are used for signing in. Alternative 1 uses a contextual aware PDA, Alternative 4 uses a specific console at the door. Alternative 2 and 3 uses device inside the room close to the executive's chair; a console or a table pc. Table 1 illustrates the differences in a unified manner.

Table 1. Read down each of the 4 columns in this table to walk through the first 4 alternatives in the "Ian presents" scenario.

<table>
<tr><th>Alternative 1</th><th>Alternative 2</th><th>Alternative 3</th><th>Alternative 4</th></tr>
<tr><td>PDA: When Ian is ready to head to Niji, Erica gives him a PDA set up for the kind of meeting Ian is going to.</td><td colspan="3">No PDA: Ian goes without PDA to Niji, where he will use in-room devices.</td></tr>
<tr><td></td><td></td><td></td><td>When Ian comes to the room he walks up to a sign-in device attached to the wall beside the door and identifies himself</td></tr>
<tr><td colspan="4">Ian walks into the room where the rest of the attendees are already seated.
He puts down his papers [and the PDA], and greets everyone.</td></tr>
<tr><td></td><td colspan="2">Ian identifies himself to the system</td><td></td></tr>
<tr><td></td><td>System fires up and one of the screens shows the first slide of his presentation. The other screen is set to whiteboard mode.</td><td></td><td></td></tr>
<tr><td colspan="4">Ian starts up the meeting with a few short announcements. When ready to present...</td></tr>
<tr><td>...he takes the PDA and presses the start button.</td><td></td><td colspan="2">... he launches presentation from a device in the room and presses the start button</td></tr>
<tr><td colspan="4">The presentation is shown on one of the displays. The other display is set in whiteboard mode. (already done in [2])
Ian proceeds thru the presentation slides using...</td></tr>
<tr><td>... the PDA.</td><td>... a device on the table</td><td colspan="2">... same device that launched presentation</td></tr>
</table>

The presentation comes to an end, Ian and the attendees have a discussion about it, and actions items are assigned.	
Ian leaves and takes the PDA with him. When Ian leaves the room, his presentation disappears from the screens. Ian returns the PDA to Erika.	Ian prepares to leave the room and signs out/exits the presentation so that the presentation and the controls for it are not accessible to the meeting attendees.
Some of the other attendees linger in the conference room and continue the discussion. After Ian left/signed out, both screens became digital whiteboards.	

Roles for Scenario 1

Next we worked out the actions of, and the causal relationships between, the various roles (*Presenter* (Ian), *Admin* (Erica), and *System*) required to fulfill the scenario in which Ian presents. The roles and the order events was summarized in tables. Table 2 shows the roles for Scenario 1. This table should be read left to right, top to bottom to understand the causal relationships.

Table 2. Roles for Scenario 1

#	Presenter (Ian)	Admin (Erica)	System
1	Sets meeting	Schedules meeting in Outlook; creates a new meeting for presenting PowerPoint with USE.	Creates the meeting object; sets owner (Ian), date.
2	Creates presentation sketch	Creates PowerPoint presentation	
3	Specifies changes	Makes changes to presentation	
4	Approves final copy	Uploads to system; configures primary display to be PowerPoint & other display to be blank.	Associates presentation with meeting; records meeting configuration.
5	Enters room; swipes card or uses biometric ID to start meeting (or carries PDA, which room notices)		Authenticates Ian, loads configuration for meeting, pre-loads PowerPoint presentation onto primary display; blanks second display; switches console interface to PowerPoint control mode.
6	Advances through slides		Advances through slides
7	Advances past last slide of presentation (the black screen)		Returns console to its main menu (Whiteboard, Teleconference, PowerPoint, End).
8	Presses “End” button		Clears displays; resets room to its default state.
9		Checks that room has been reset; if not, presses “End” button on the console.	Clears displays; resets room to its default state.
10			Clears displays and resets room to its default state if steps 8 or 9 have not been performed, the meeting time has elapsed, and there is no other room activity.

Lessons Learned: the Scenarios

One central technology from the user’s point of view in these scenarios is the device to control the conference room. This device needs to be connected the rest of the system so that it can send commands to the technology installed in the room. It also needs to be aware of the system's status so that it can inform the user of it. The connections must be fast and reliable. For example when the user shows a presentation, the slide control must be instantaneous in order to give appropriate feedback to the user; if the slide does not change immediately the user will most likely hit the "forward" button repeatedly, confusing the system (and the user). It is essential that the user interface (UI) of the control device communicate to the user clearly and immediately the complex settings of the room: a glanceable, immediately updated state display. In the next section below, we discuss in detail the iterative design of the console.

The UI does not rely on a particular form factor or type of device. The UI is designed so it can be accessed and easily used on a wide range of devices such as PDA, touch screens in the room,

personal laptops, or cell phones. For the room Niji, we use a touch screen tablet PC as a tabletop kiosk, dedicated to the Console UI.

The conference room envisioned in these scenarios includes several separate technologies accessible through a single UI. To accomplish this, we designed a conference room control system that can connect with authentication equipment, projectors, video conference equipment and whiteboard applications. This system acts like a mediator between the adaptable UI on the device in the room and the technologies such as video conference equipment installed in the room. In addition to supporting the capabilities during an ongoing meeting, the system also needs to support the preparations and the activities following up the meeting.

The USE system uses card readers and/or fingerprint readers to allow Ian to sign in securely. When Ian signs in, a successful authentication starts a number of processes, adapts the UI on the device, starts appropriate applications etc. Ease of use and good feedback in the authentication process are essential. One important issue with the authentication is that it must succeed every time for high profile users, such as Ian the CEO. For these people failure is often unacceptable; for example, the CEO could lose face if failure with authentication occurred in the presence of others. For this reason we chose RFID authentication over biometrics such as fingerprint readers; the successful read rate on off-the-shelf fingerprint readers is much lower than RFID card readers, often requiring multiple tries.

Since the room includes video conference equipment, the system needs to be able to deal with the capabilities of the remote partner in a video conference. The simplest scenario is that the remote room uses a similar installation as our conference room with the same kind of control system architecture. In that case, the system can communicate with the remote room to launch applications. Also, if the system regularly connects to a particular remote location, a profile of that remote location can be set.

Designing the Console: meeting the requirements of the executive

Underlying design principles guiding the ConsoleUI design

The current interface to the USE system is a standalone application known as the ConsoleUI. A few key ideas guided its design, based on the needfinding and use scenarios noted above. Most importantly, we wanted to let the user focus on the communication tasks at hand in the meeting, and specifically avoided approaches that exposed any procedural issues involving technology control (Ponnekanti, 2001). On the technical side, we wanted to make sure the pace of the user's interaction matched the response times of devices in the room, in order to reduce frustration and the perception that "this thing isn't working." Finally, we wanted the interface to be flexible and extensible, to accommodate changes in system capabilities.

Our first design is for a user who is generally interested in the goals of a meeting rather than the engineering of a conference room's systems. For example, a phrase like "I want to call Tom" means much more to most users than "I want to switch my video source from the presentation computer to the video conferencing system and initiate a video conferencing call via an ISDN line …" . In its simplest form the USE Console UI presents only potential end results to the user, for example: "call Tom," or "present my PowerPoint file." This form of the UI is intended to insulate the user from the complex processes that are necessary to carry out their requests. By presenting all options in a manner in which the user is comfortable, our hope is that the user will come to trust the system; this will allow them to focus on their tasks, not on using the UI.

This trust allows the USE console to overcome one of the use system's perpetual HCI problems: latency. Some of the devices in the room take a few seconds to react to user input. Since these devices can be affected by any number of factors (e.g. heat and network congestion) the duration

of these delays can be very difficult to predict. As a consequence, the USE ConsoleUI deliberately slows the user interaction down. This is accomplished by slowing down any animations or transitions in the UI. Even objects like animated buttons have slower than normal animation cycles.

Finally, conference rooms are constantly evolving. Because of this, the USE system is designed to be open-ended, allowing integration of new applications and equipment that might come into a conference room. This meant that we needed a highly adaptable UI. We accomplished this by keeping the ConsoleUI almost purely devoted to interface logic—the actual code for controlling devices in the room is distributed throughout a number of web service applications. All control elements within the UI can be automatically configured, within some range of possible configurations.

Contextually aware UI: a One Button Interface?

It took extensive user studies and several meetings, but the basic UI that we needed to develop became pretty clear. We needed one big button that always did the right thing at any given time. We came to think of it as a contextually aware "next" button: “Next, I want to see some slides." “Next, I want to place a conference call."

Our initial design came quite close to the one button goal. The user’s activities were heavily scripted via a configuration UI. The configuration UI allows the user to add presentations or set up a videoconference ahead of time. In this early version it also determined the order in which events occurred. When the user completed a task they pressed the done button and were presented with the next task. If a task was completed naturally, the next task would be presented automatically. While this UI was certainly simple, it didn’t allow for much flexibility. The flexibility that it did allow took several steps to achieve. One feature from this design that did survive into the final design is the division of the screen into two distinct areas that reflect the two displays in the prototype conference room.

After several iterations we finally arrived at the design that became our first working prototype (Figures 1, 2, and 3). This design stayed fairly faithful to the one button esthetic but did provide a great deal more flexibility.

Figure 1. First working interface: Running a video conference on the left screen and viewing a presentation list on the right screen.

Figure 2. First working interface: Running a video conference on the left screen and a presentation on the right screen.

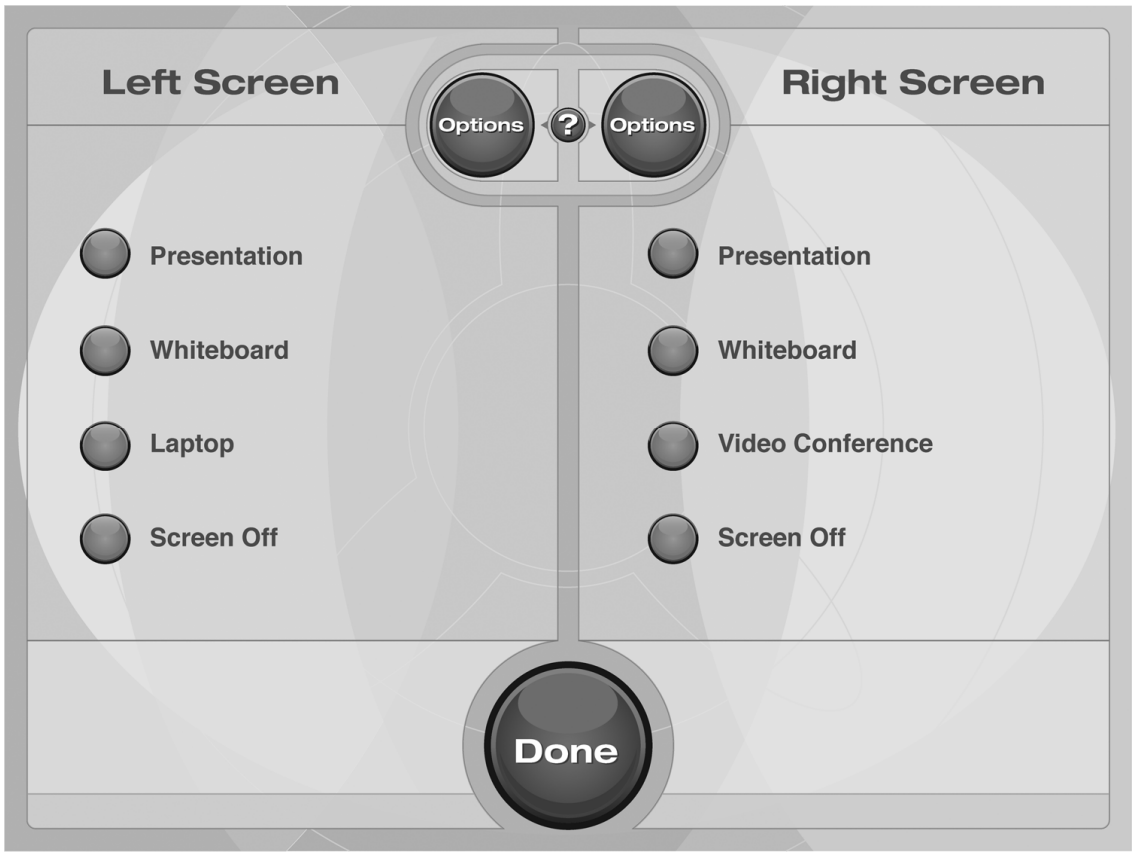

Figure 3. First working interface: Viewing option lists on both screens.

This new design incorporated option buttons, one per side, which allowed the user to select specific tasks (Figure 3). This change offered the user a great deal of freedom without adding much complexity.

Our initial prototype functioned quite well, but testing did reveal one flaw in our design logic. In order to switch between displaying two presentations the user needed to perform three actions and visit three screens. This was an unacceptably complex interaction within the "One Button for one task" paradigm.

Our quick solution to the problem was to add more buttons to each screen. This solved the “three clicks” problem but at the expense of clutter and ease of use. These new buttons were just too small to select on our touch screen.

Our current solution is a compromise. Switching between presentations requires two clicks, but a bridging animation makes the operation “feel” seamless. From the “Presentation Options Screen” (Figure 4) the user selects a presentation. The graphics that embody the selected presentation’s “button” transform into the presentation’s controls (Figure 5, left). Clicking the green “List” button reverses the animation, presenting the user with the original list of available presentations.

Our original prototype had a similar problem dealing with multiple videoconference selections. The user was continually forced to switch between the “Video Conference” screen and its “Options” screen. In our current prototype this is no longer necessary (Figure 5, right). This

interface also supports a number of simultaneous connections (the number is determined by device capabilities). Each can be initiated and terminated independently.

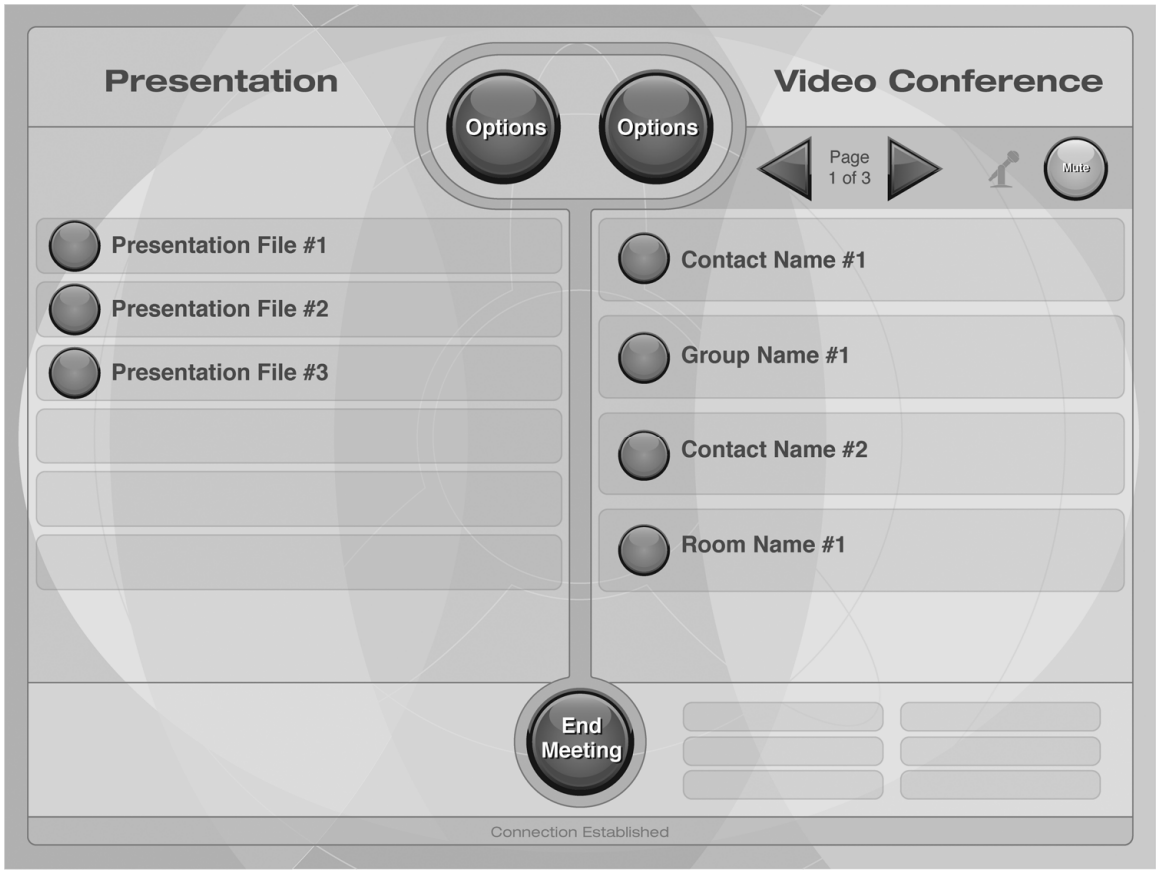

Figure 4. Current solution: Viewing a list of loaded presentations on the left screen and the video conference "address book" on the right screen.

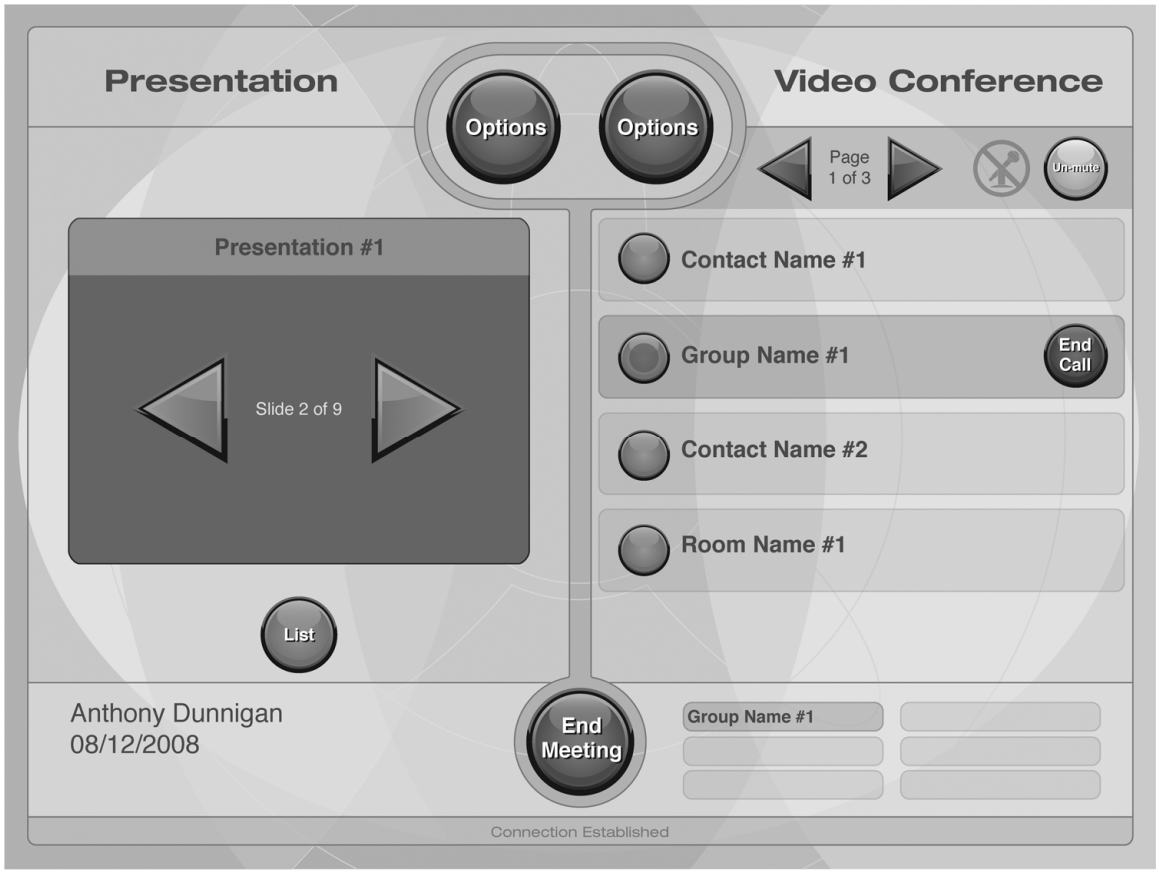

Figure 5. Current solution: Running a presentation on the left screen and a video conference on the right screen.

The ConsoleUI design implementation

Communicating with the USE system

The current design of the Console UI allows for the manipulation of several physical devices from one touch screen. Currently the interface is an Adobe Flash MX graphical interface that communicates with other components of the system via messages passed through a web service (the ConsoleService). The UI communicates with the ConsoleService via an XML socket. The socket communication is bi-directional and asynchronous: either end may initiate an exchange of messages, but user interaction is not automatically blocked during these exchanges.

The XML messages sent to the user interface describe the setup of the room, the devices and applications available, and objects available to those devices and applications. For example, PowerPoint is one application running on a particular display, and its objects are the set of PowerPoint files assigned for the current meeting. Another example is a video conference device with contact objects that specify who can be called, and which ISDN number or IP address to use. The interface sends command messages to the ConsoleService; the ConsoleService in turn forwards them to the RoomControl service, where commands are executed. Commands include selecting a particular device or application, and manipulating them by advancing slides, making calls, etc. as appropriate to each device or application.

After the ConsoleUI initializes, users can invoke actions by pressing various buttons on the touch-screen. All actions are asynchronous and have the following logic: the user requests an action via button-press; the request is passed to the ConsoleService; the ConsoleService responds with a "working" message and queues the request for processing by the RoomControl; the ConsoleUI displays a waiting state (in the button) -- most of the interface is still live at that point, though some selected options may be disabled; the ConsoleService receives a response from the RoomControl (error, or success with further details), and sends a "completed" message to the ConsoleUI; the ConsoleUI updates its state (either by displaying an error or by making some change in visible state).

But why use Flash?

Flash is very good at presenting and animating graphics. The ConsoleUI relies on custom graphics and animations to impart important information to the user; such non-standard UIs are fairly easy to design and iterate in Flash, whereas they require sophisticated and time-consuming development in conventional user interfaces. Many platforms support Flash, allowing one UI to be deployed on many dissimilar devices. With little or no alterations the ConsoleUI can be run from a laptop computer (PC or MAC), a tablet PC or a handheld mobile device. This is another feature of the ConsoleUI that has proven very valuable during the development of the USE system. It is often helpful to control the USE system from within the Flash development environment.

Flash makes it possible to exchange XML over a socket connection, which makes it possible for a Flash program to act as a peer for another program, such as a web service. While it may be possible to implement a Web Service client directly in Flash, we required bi-directional communication among the services in our system. We found it easier to have a WebService front end to mediate communication with the rest of the system, leaving the complexity of SOAP/WSDL communication to .NET. We are now exploring the possibility of embedding Flash and a COM component in a .Net Windows form.

Yes, but is it configurable?

Because different conference rooms can have different capabilities, and even different culturally determined decor, we have set up the ConsoleUI application so that it can be auto-configured, within some range of possible configurations. This includes room capabilities, the user's preferred language, UI colors, on-screen messages and all labels. So, for example, one conference room might have two display computers, and another conference room might have three. Similarly, a user might choose one set of colors for a ConsoleUI running in a corporate boardroom in the USA, and an entirely different set of colors for the same UI running in the conference room of a small design firm in Japan. Configuration information is passed to the ConsoleUI from the ConsoleService in an initialization message, which gets it in turn from the user's preferences, the room's preferences and entries made in the Configuration UI.

Lessons Learned: ConsoleUI Design

It was surprisingly easy to develop this UI in Flash. Much of this surprise was due to a lack of documentation concerning Flash's ability to pass data via an open socket, especially XML data. Once we were able to write the code necessary, Flash proved to be quite stable, sending and receiving hundreds of thousands of messages in one particular test.

So that Flash could have a predictable partner to communicate with, a web service was created in C# that contained a built-in socket communication module that sent and received data. In our method, Flash connects to this web service; after that, data flows both ways. There was some trial and error involved in getting clean messages into and out of Flash. Again, once the proper method was discovered, Flash proved very stable.

Our choice of a Slate tablet PC to host the console UI is one we will change. This tablet's limited video capabilities have sometimes interfered with the UI's animation effects. This poses a real problem, since so much important information is contained in those animations. When they move at the wrong speed or stutter, their meaning becomes corrupted. It also looks bad, which lessens a user's confidence in the system. We are now testing a touch screen driven by a more capable PC that should solve this and other performance problems.

System Architecture

The USE system consists of several components: a configuration repository, one or more orchestration components (for each physical space that needs to be controlled), an optional user console for each space, and (for each physical space) one or more devices to be controlled. Figure 6 shows a diagram of one instantiation of this architecture. It should be understood that more than one serial device and more than one TCP/IP device can be controlled simultaneously; similarly, more than two applications can be run on each display machine.

Component Configuration

The USE system coordinates multiple devices based on a given configuration to put each device into an appropriate state for the specific task. The user can exercise control over the system by selecting task-oriented actions; the system performs the required device orchestration.

Unlike canned room control environments that are designed to work with specific sets of devices, this architecture supports dynamic plug-and-play capability: devices can be swapped out without rebooting the system or affecting existing configuration scripts. New devices can be introduced without requiring any code changes in the existing orchestration framework.

Although all components of the system can run on the same computer, typically the components are distributed among several machines, some of which may be co-located, while others may be remote. Communication among components is mediated by web services calls, although other communications mechanisms such as RPC may also be used. Figure 6 illustrates the architecture that consists of a configuration repository, a room control PC, and two Display PCs. The Display PCs have their displays projected onto Smartboard electronic whiteboards through overhead projectors. The video switch controls the routing of displays, including guest laptops that can be plugged into a VGA cable in the room. The system may be controlled through any number of devices shown on the right.

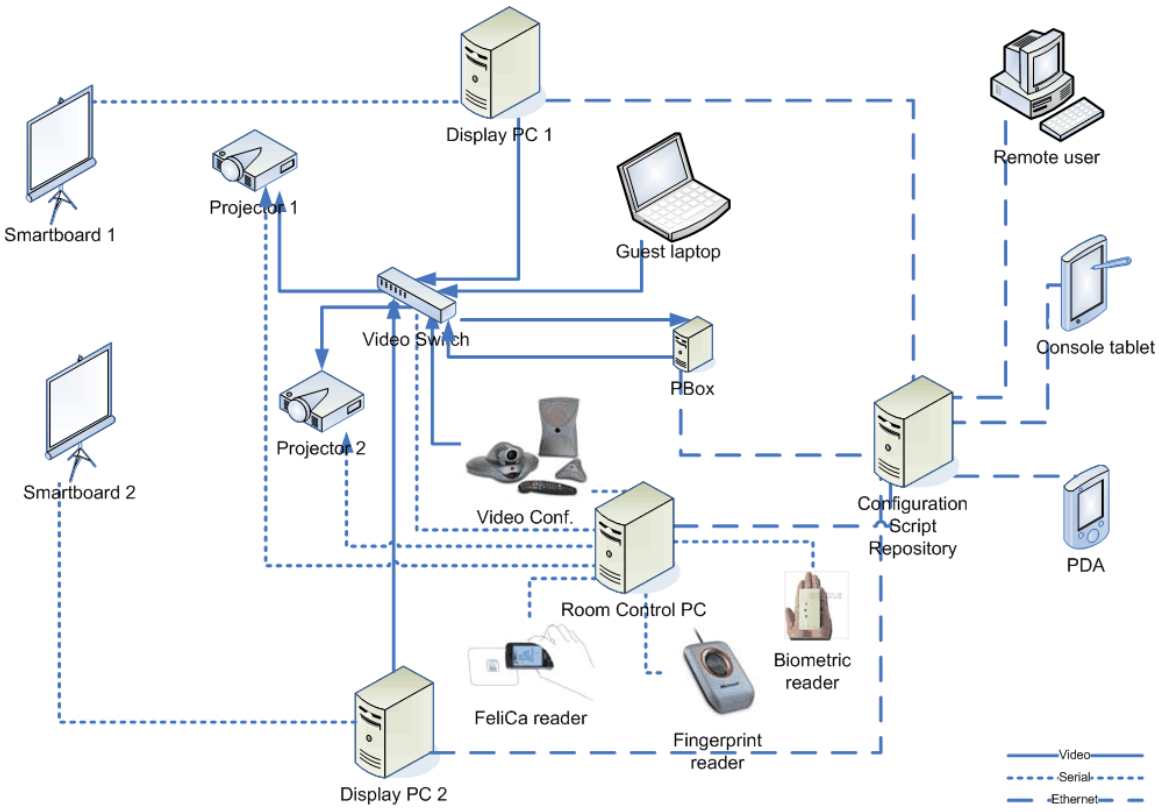

Figure 6. Devices and their connections.

Devices and device states

The configuration repository manages a database of devices, device states, rooms, users, meetings, and configurations. Each room contains one or more devices, and each meeting involves the use of a configuration that orchestrates the use of some subset of the room's devices. Each object (Device, DeviceState, Room, Meeting, etc.) has a unique id, and zero or more named attributes. A configuration orchestrates devices by collecting a set of DeviceStates, each of which represents some device in a particular context. For example, a room may contain two identical projectors. The database would contain one Device object that characterizes the type of projector, and a meeting configuration would contain two DeviceState objects, one for each of the projectors. The attributes of each DeviceState would specify how the particular projector should be connected to the system, including how it is connected to the video switch device, and what serial port should be used to control it.

Meetings

Prior to starting a meeting, the meeting owner configures it by selecting the applications and devices that will be used, and by specifying which documents and video contacts (if any) will be used during the meeting.

When the system is started, the orchestration component requests from the configuration repository the configuration appropriate to the space being controlled, the time of day, and (optionally) the person making the request. The information returned by the repository is translated into a configuration for the meeting. The configuration consists of a collection of DeviceState objects that describe how each device and application that participates in the meeting should be configured. This allows the system to put the devices into known states

regardless of their earlier state. Thus manual operation of devices (such as switching video sources or changing projector settings, for example) during gaps between meetings will not affect system behavior during a scheduled meeting.

Applications

Devices controlled by the framework include serially-controlled devices such as projectors, video switchers, and some video conference devices, and network-controlled devices such as other computers, some video conference devices, etc. From the perspective of the orchestration architecture, applications (e.g. PowerPoint) that run on computers in the space being controlled also constitute devices to be orchestrated. Applications are controlled by a Controller component that receives input from the orchestration engine and selects the corresponding application for display. Multiple applications can be run simultaneously on each display machine; the system places no restrictions of what kinds of applications can be run in this manner.

Each application can receive commands through the Controller component. There are three classes of application in the system: built-in applications, opaque applications, and applications with some automated control. The whiteboard application is an example of a system-specific application. It can be controlled on a fine-grained level by injecting custom messages into its message queue. An opaque application (such as a web browser) can be opened, closed, hidden or displayed, but cannot be controlled in any custom way by the system. Finally, the third kind of application is one that has a published API for controlling its execution. PowerPoint, for example, has a COM API for advancing slides and performing other remote control actions. A special IDevice is created for controlling PowerPoint programmatically. This allows users of the system to use a console user interface to drive a presentation on a public display without physically interacting with the computer that controls the display. Thus remote users can present to local audiences.

Interaction

The system's use of scripts allows much of the interaction with the system for control purposes to be reduced to a few simple selections, or to be entirely unnecessary.

One class of interactions involves logging into the system. This can be accomplished through an RFID (FeliCa) badge, through a fingerprint or other biometric reader, or by typing the userid and password into a trusted application. A successful swipe of a thumb or an RFID is sufficient to open a document or an electronic whiteboard application on one screen, and to initiate a teleconference call on another. In each example, the user focuses on the task – sketching, presenting, and communicating – rather than on the mechanics of controlling devices or figuring out which remote control to use.

Once a person is logged in, he does not need to use remote controls or buttons on the various devices to control the room. Most of the device state is configured automatically; the few cases where some input is required (e.g., choosing which contact to call to establish a videoconference) is accomplished with a single selection on a touchscreen console used to control the room. The system automatically takes care of switching the projector to video mode, determining the appropriate way to initiate the connection, and connecting to the remote cite. This avoids one of the pitfalls of un-orchestrated spaces in which some devices can be left in unknown states, making it difficult to know what needs to be done to accomplish the desired task.

Component structure

The system can be decomposed into a stack of components, as shown in Figure 7. Starting from the bottom, the system contains databases for storing configurations and authentication information. These databases are wrapped by abstraction layers that manage database access and make the rest of the system independent from the particular choice of databases. In addition, the persistence layer makes it possible to manipulate database records as first class objects,

further isolating persistent storage details from the system. The Generic Objects layer represents the schema of the objects stored in the database. The Business Objects layer contains objects that directly control the various devices and applications that make up an instance of the system; these are the IDevice objects created by the RoomControl component described below.

Authentication and access control is performed by the security component. This consists of authenticating users who want to sign in, generating session tokens for signed-in users, distributing theses session tokens to other components of the system, and validating session tokens on request. This component maintains user accounts, but will use Windows Domain authentication to authenticate users, unless the login credentials do not correspond to a Windows domain login. Thus the authentication component allows both domain users and remote users to use the system. In addition, trusted authentication devices are used to generate authentication requests. This hardware includes RFID and biometric readers.

The application logic layer corresponds to the network of distributed components that deliver the appropriate configuration information to the right room at the right time, and orchestrate the meetings. Finally, multiple user interfaces (such as the meeting console, the configuration editor, scheduler etc.) allow users to create, modify, and use the information stored in the system.

Related work

The USE system bears some similarities with other systems for managing devices in smart environments, including Patch Panel (Ballagas, 2004), Metaglue (Coen et al., 1999), Gaia (Román et al., 2002), and Aura (Sousa & Garlan, 2002). Unlike Patch Panel, we are concerned with integrating complex existing devices such as video switchers, video conferencing systems, and applications rather than “primitives” such as joysticks and buttons. We do not require applications to be designed specifically for our system, as is done in Metaglue and Gaia.

The system is designed to be predictable and controllable by the user, rather than autonomous, as are other systems (Coen et al., 1999; Román et al., 2002; Sousa & Garlan, 2002). This means that there is a clear relationship between users’ actions and system responses in our system than in autonomous ones. We focus on short-term meeting support rather than on long-running tasks that may be moved from one location to another (Sousa & Garlan, 2002). Our task focus means that the system behaves quite differently from (Hanssens, Kulkarni, Tuchida, & Horton, 2002) or (Sousa & Garlan, 2002): if a projector is turned off, the system of this invention turns it on, whereas (Hanssens, Kulkarni, Tuchida, & Horton, 2002) will re-route the display to another projector. While this adaptive routing may be useful in a room with many redundant displays, it is not a useful solution in most cases.

We also distinguish our work from presentation management systems such as Epic (Rui & Liu, 2004). Epic is designed to orchestrate the presentation of slides on multiple screens in the course of a presentation. The system described here, on the other hand, orchestrated multiple applications and devices.

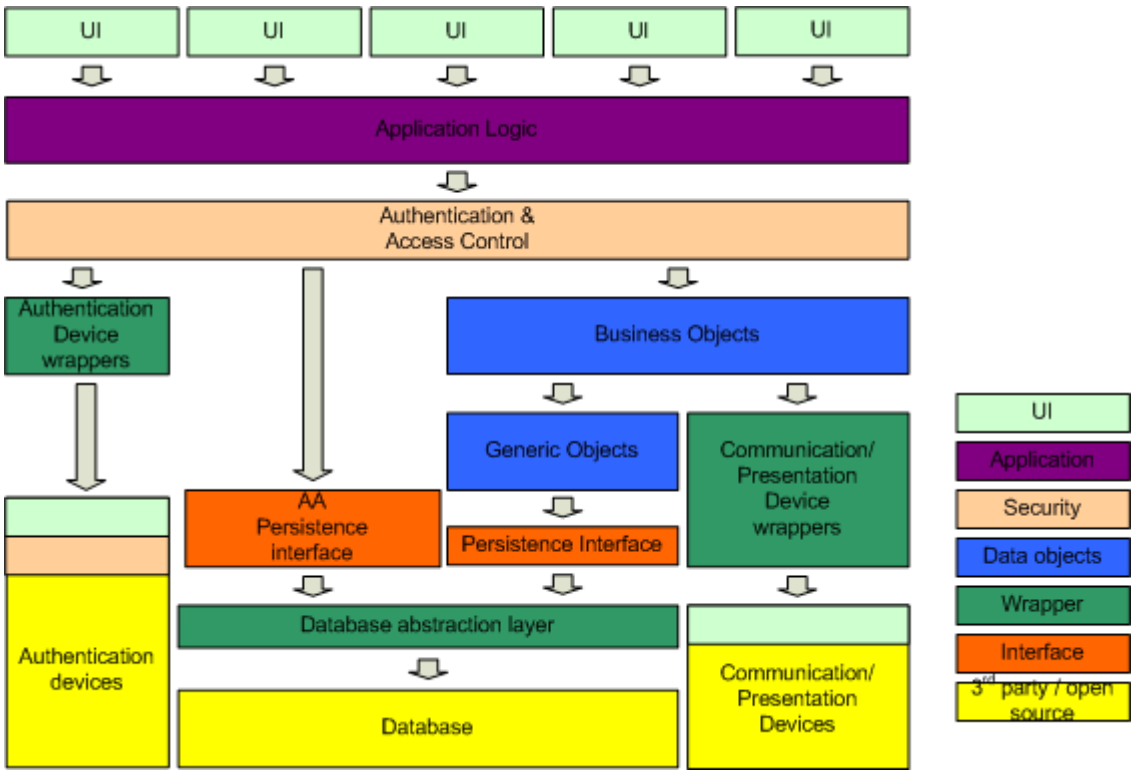

Figure 7. Architecture stack.

Looking to the future

New technologies like sensor networks, tangible interaction devices, virtual environments, and an array of mobile devices will have an impact on meetings and meeting rooms. In particular we plan to focus on two new areas as we extend the USE system.

Mobile interaction

Increasingly distributed work processes along with the ubiquity of mobile devices provide motivation and means for mobile meeting support. But what does it mean to be in a meeting while mobile? Mobile users face higher demands on their attention and thus it may be difficult for them to maintain the level of awareness necessary to follow a meeting in real time (Oulasvirta, 2005). By the same token, it can be difficult for non-mobile participants to understand the disposition of a mobile user (Cheverst, 1999). However, systems should not merely make allowances for mobile participants but also allow them to contribute in ways non-mobile participants are not able to. Should we create a section of the room-based UI for enabling mobile clients?

We see design and development possibilities rooted in the following questions: How do people who are mobile contribute to a meeting? How can they interact seamlessly with other mobile users as well as participants in augmented environments? How can systems support mobile meeting participants who may be distributing their attention across multiple activities? What is the right level of awareness to convey between mobile participants and those in augmented environments? How can a system support activities unique to mobile users, such as data capture in the field?

Sensing

In order to support fluid communication and promote rapid recovery from conversational breakdowns in a meeting, it is necessary to convey a rich representation of participants' context. To achieve this, a system must be able to collect and convey a wide spectrum of information (Marshall & Bly, 2005), while still providing mechanisms for participants to reveal and potentially alter sensing policies (Lederer, 2004). Furthermore, a system may need to resolve conflicting policies between meeting sites and participants.

Interesting aspects of sensing for smart meeting environments include: In what ways can information sensed implicitly about meeting participants be used to augment awareness or in some other way support participants in situ? What are the appropriate interfaces or mechanisms to access captured information? What is the right information to capture and how should it be modeled? What straightforward methods can be employed to regulate privacy settings on sensed data? And, how do all these ideas inform the design of the next USE system?

Conclusion

We have found that the USE system as designed has allowed not only Ian, the principal user, but also multiple users with varying degrees of technical skill to control our conference room. In each case the limited number of options the UI provides was sufficient for their needs. This has led us to believe that our design approach, initially targeted to one specific user type, might be extensible to most users.

We have noted some focus issues: users seem not to, and probably should not, lavish any more attention on the ConsoleUI than is necessary to accomplish their goals. We have observed users pressing a button in the UI and then looking at the appropriate room display almost immediately. Because of this behavior, much of the feedback that the UI provides to users is ignored. A solution to this problem might involve incorporating environmental cues into the USE system. Lighting cues or subtle graphical overlays on the displays in the room might provide the feedback necessary. At a minimum, users must trust that their command was sent and is being executed. They must also know when a command has been completed or has failed. Our current UI provides all of this information, as well as some visual cues as to the amount of time a command will take to complete; however, often this information is missed by the user.

Our next interface will certainly be more customizable, but our initial prototypes have indicated a definite range for that customizability. Our understanding and the basis for our design iterations will be further enhanced by design reviews, use studies, and the extension of the USE system into a larger conference room with more users and different, less constrained requirements.

Acknowledgements

We would like to thank our colleagues at FXPAL for supporting the USE project and tolerating the constant flux of our conference rooms. Special thanks to our informants, and to John Doherty, Jonathan Cohen, Paul MacEvoy, and Gerry Filby, who provided various technical underpinnings for the USE system.

References

Ballagas, R., Szybalski, A., Fox, A. (2004). *Patch Panel: Enabling Control-Flow Interoperability in Ubicomp Environments*. Paper presented at the Conference Name|. Retrieved Access Date|. from URL|.

Cheverst, K., Blair, G. S., Davies, N., and Friday, A. . (1999). The Support of Mobile-Awareness in Collaborative Groupware. *Personal and Ubiquitous Computing, 3*(1/2), 33-42.

Coen, M., Phillips, B., Warshawsky, N., Weisman, L., Peters, S., & Finin, P. (1999). Meeting the Computational Needs of Intelligent Environments: The Metaglue System. *Proceedings of MANSE'99*.

Foote, J., Liu, Q., Kimber, D., Chiu, P., and Zhao, F. (2004). *Reach Through the Screen: A New Metaphor for Remote Collaboration*. Paper presented at the Conference Name|. Retrieved Access Date|. from URL|.

Fox, A., Johanson, B., Hanrahan, P., and Winograd, T. . (2000). Integrating Information Appliances into an Interactive Space. *IEEE Computer Graphics and Applications 20*(3), 54-65.

Hanssens, N., Kulkarni, A., Tuchida, R., & Horton, T. (2002). Building Agent-Based Intelligent Workspaces. *International Conference on Internet Computing*, 675-681.

Lederer, S., Hong, J., Dey, A. K., and Landay, J. A. . (2004). Personal privacy through understanding and action: Five pitfalls for designers. *Personal and Ubiquitous Computing, 8*(6), 440-454.

Marshall, C. C., & Bly, S. (2005, April 2-7). *Saving and Using Encountered Information: Implications for Electronic Periodicals.* Paper presented at the Conference of Human Factors in Computing Systems, New York.

Oulasvirta, A., Tamminen, S., Roto, V., and Kuorelahti, J. (2005). *Interaction in 4-second bursts: The fragmented nature of attentional resources in mobile HCI*. Paper presented at the Conference Name|. Retrieved Access Date|. from URL|.

Preece, J., Rogers, Y., & Sharp, H. (2002). *Interaction Design: Beyond Human-Computer Interaction*. New York: John Wiley & Sons.

Román, M., Hess, C. K., Cerqueira, R., Ranganathan, A., Campbell, R. H., & Nahrstedt, K. (2002). Gaia: a middleware platform for active spaces. *Mobile Computing and Communications Review, 6*(4), 65-67.

Rui, Y., & Liu, Z. (2004). ARTiFACIAL: Automated Reverse Turing test using FACIAL features. *Multimedia Systems, 9*(6), 493 - 502.

Sousa, J. P., & Garlan, D. (2002). Aura: an Architectural Framework for User Mobility in Ubiquitous Computing Environments. *Software Architecture: System Design, Development and Maintenance, IFIP 17th World Computer Congress - TC2 Stream / 3rd IEEE/IFIP Conference on Software Architecture (WICSA3), August 25-30, 2002, Montréal, Québec, Canada, 224*, 29-43.

Star, S. L. The Ethnography of Infrastructure. *American Behavioral Scientist, 43*(3), 377-391.1999.

Streitz, N., Geisler, J., Holmer, T. (1998). *Roomware for Cooperative Buildings: Integrated Design of Architectural Spaces and Information Spaces*. Paper presented at the Conference Name|. Retrieved Access Date|. from URL|.

Weiser, M. (1991). The computer for the 21st century. *Scientific American, 265*(3), 94-104.