PaperPoint: A Paper-Based Presentation and Interactive Paper Prototyping Tool

Beat Signer and Moira C. Norrie
Institute for Information Systems, ETH Zurich
CH-8092 Zurich, Switzerland
{signer,norrie}@inf.ethz.ch

ABSTRACT
Recent developments in digital pen and paper solutions enable, not only the digital capture of handwriting, but also paper to be used as an interactive medium that links to digital information and services. We present a tool that builds on technologies for interactive paper to enable PowerPoint presentations to be controlled from printed slide handouts. Furthermore, slides can be easily annotated during presentations by simply drawing on the printed version of the slide. As well as discussing the advantages of such a paper-based interface and initial findings on its use, we describe how we were also able to exploit it to provide a general prototyping tool for interactive paper applications.

Author Keywords
Paper user interface, presentation tool, rapid prototyping, pen-based input.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
The affordances of paper make it superior to digital media for many tasks [14]. One particular advantage of paper is that it is easy to annotate and, even though handwriting can be captured on digital devices such as Tablet PCs, many users still prefer to write on paper. Other advantages are that paper is cheap, light, easily carried and can be viewed from different orientations. In contrast, digital displays still tend to be relatively heavy, expensive and awkward to carry and the user often has to position themselves directly in front of it to view it clearly. For these reasons, among others, several projects have investigated means of linking paper to digital media and services so that each medium may be used when and where appropriate for the task at hand [6, 17].

One particular scenario where we often see the two media used in parallel is in giving presentations. Presentation tools such as Microsoft PowerPoint have become one of the most commonly used computer applications and what we often see is the presenter referring to a printed handout overview during a presentation. Used alongside the laptop display, such a document helps the speaker keep track of the presentation and may be used to determine slides to skip under time pressure or locate relevant slides during discussions. Although PowerPoint now offers the possibility that the laptop shows a presentation overview while individual slides are displayed on an external screen, some users tend to find that too much information on the laptop screen is a distraction and prefer instead that it mirrors exactly what the audience is seeing. Notes are often written on a printed version when preparing a presentation to serve as reminders or prompts to the speaker. They may also be written during the presentation to record questions and comments from the audience or to note corrections to be made to the slides.

Recent technologies for the digital capture of handwriting have now been adapted to also allow users to interact with paper using a digital pen. Based on these technologies, it is possible to develop paper-based interfaces that allow applications to be controlled from paper. We have developed a very general and flexible framework to support the development of such applications and, in this paper, we describe a tool that we implemented to enable PowerPoint presentations to be controlled from printed slide overviews. Furthermore, slides can be annotated in a very natural way by drawing and writing on the printed versions with the digital pen. We describe the implementation, showing how the underlying framework can be used to provide paper-based interfaces to control other applications. In addition, we show how we could exploit PaperPoint to produce a general prototyping tool for such applications and report on our initial findings on the use of PaperPoint.

We begin with a discussion of related work and then provide an overview of our framework for interactive paper and the digital pen and paper technologies on which it is based. We then present the PaperPoint application in detail, including a description of how we have used it to support the rapid prototyping of interactive paper applications in general. After describing the implementation, we discuss user experiences and plans for future work.

RELATED WORK
A paper-based user interface addressing the issue of giving flexible presentations was investigated in *Palette* [9].

TEI'07, February 15-17, 2007, Baton Rouge, Louisiana, USA.

In preparation for a talk, presenters create their slides with PowerPoint as usual. After the digital slides have been prepared, a special paper card is generated automatically for each slide. The Palette converter program reads the document authored with the PowerPoint presentation software and creates a new document with the Palette cards containing a thumbnail view of the slide's image, some optional text notes, the slide's number in the sequence of the original presentation order and a visible barcode. Each digital slide can be accessed and displayed on the computer screen by holding the paper card under the Palette barcode reader. This paper card-based access to a digital presentation enables a flexible on-the-fly reorganisation of a talk by providing easy random access to single slides and simplifies the composition of new talks based on card sets created for and used in previous talks. The paper cards further offer space to directly write down questions and comments asked during a presentation. However, note that this paper card-based access to single slides may also create problems during a presentation in that it might sometimes be difficult to find the "next" slide.

A drawback of the Palette system is that the intrinsic mobility of the paper cards becomes restricted by the barcode scanning device. To access a specific slide, the Palette cards have to be placed under the barcode reader which has a fixed position. *PaperButtons* [12] overcomes this problem by extending the Palette system with "paper buttons", i.e. electronic buttons which can trigger different actions and are attached to a piece of paper. Each paper card has one button that replaces the barcodes used in the Palette system. After pressing this button, a unique identifier is sent over a radio frequency connection to a receiver that transfers the information to the Palette application manager showing the appropriate slide. The wireless solution solves the problem of restricting the presenter to being in the vicinity of the barcode reader.

The same idea of having fast random access to digital information based on a paper user interface is addressed by the *PaperIcons* project [13]. In PaperIcons, a printed clip art book is used for fast random access to a digital clip art gallery. While browsing a paper document, a user can pick printed icons and drop them onto a pen-sensitive computer screen. This pick-and-drop operation is suitable for applications where a user can, for example, select clip arts from a physical book and integrate them into a digital presentation. The PaperIcons project uses CyberCodes, a form of 2D barcodes, to uniquely identify different document pages. To track a user's selection defined by the pen's position within a single page, the document has to be placed on a pen-sensitive tablet. An over-desk camera is used to read the CyberCodes attached to document pages encoding the page numbers. The page number resolved by this camera feedback is used in combination with the positional information delivered by the pen-sensitive tablet to access the corresponding digital object. Similar to the Palette system, PaperIcons introduced some limitations in terms of mobility since the clip art book has to be placed on a fixed pen-sensitive tablet to be identified by the over-desk camera and tracked by the tablet.

Real-world graphical user interfaces (RWGUIs) were introduced by Masui and Siio [7] as paper-based interfaces for arbitrary physical devices. Customised paper-based remote controls can be defined by the user and accessed via the *FieldMouse* [16], a special reading device based on mouse tracking technology measuring a user's pointing position relative to a reference location within a paper document. Masui and Siio present an example of a paper-based TV volume control, where a special barcode has first to be scanned and then the volume of the TV can be controlled by simply moving the FieldMouse. The goal of the paper-based control is to combine functionality from different control devices and provide a flexible real-world graphical user interface which can be easily customised.

More recently, various Tablet PC and PDA-based solutions try to enhance the presentation experience by providing a pen-based user interface for digital ink annotations. The *Classroom Presenter* system [2] is based on a Tablet PC and enables presenters to annotate existing PowerPoint slides. However, in terms of mobility, the weight of the Tablet PC is still an issue. Other solutions integrating presentations with digital annotations include the *ZenPad* [1] that was developed within the *Classroom 2000* project as well as the *Pebbles Remote Commander* application that was implemented as part of the *Pebbles* [8] project.

INTERACTIVE PAPER FRAMEWORK

Our interactive paper framework (iPaper) enables active areas, in the form of arbitrary shapes, to be defined on paper documents. These active areas can then be linked to supplementary digital information or services. Each time a user selects a position within one of these active paper regions, the corresponding link is activated and the associated link target accessed.

The interactive paper framework is based on a client-server architecture as shown in Figure 1. On the client side, a special input device, for example a digital pen, is used to detect (x,y) coordinates within an interactive paper document and send these to a computing device such as a regular PC or a PDA. In addition, the input device has to identify the document it is used on and the page number within this document. The document's identifier (ID) and page number together with the positional information are transmitted from the client to the server component responsible for further data processing via an HTTP request.

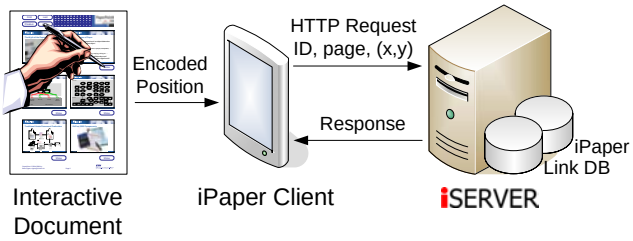

Figure 1. Interactive paper framework

The server side consists of our cross-media information management component, called iServer, with an interactive paper resource plug-in (iPaper). It is out of the scope of this paper to describe the details of the iServer platform which basically allows the definition of links between arbitrary digital or physical resources. Further information about the iServer platform and the interactive paper framework is provided in [15].

It is important to note that all operations on the server side are completely independent of a particular paper, printing or input device technology. The only information required by the server is the document identifier, a page number and an (x,y) position.

The input devices used for our most recent interactive paper applications, including the PaperPoint presentation tool, are based on *digital pen and paper* functionality offered by the Swedish company Anoto [3]. The basic idea is to detect a digital pen's absolute position on a paper document. Therefore, positional information is directly encoded on each piece of paper using a special pattern of tiny visual dots. The dot pattern results in a slightly grey page background with minimal interference with the document's content.

Any digital pen to be used in combination with Anoto encoded paper has to be equipped with a camera in addition to the writing stylus to track the pen's movement relative to the paper surface. The pen's buffer memory can then be used to digitally recreate what a user has written on a paper document. The *Magicomm G303* digital pen that we used for the PaperPoint application is the first commercially available Anoto pen that innately offers the possibility to continuously stream its data to a computer. Thus, we can use it as an interaction device as well as for writing capture. Note that our system does not use any existing Anoto interface elements (pidgets) and is based purely on functionality provided by the iPaper framework [10]. The Anoto technology was originally designed for writing capture and form processing and the authoring tool misses some of the features for interactive paper such as multi-layered active areas that we consider important and are provided by our framework. The only component provided by Anoto that we use is the Acrobat plug-in to generate and print the pattern. All the mappings from active paper areas to digital services are handled by iServer and its iPaper plug-in.

As outlined in the previous section, there exist other technologies that can be used to integrate paper and digital material. For example, *DataGlyphs* [4] encode information in printed artwork. This implies that the glyphs can only be embedded in paper regions covered by some artwork or they will interfere with empty (white) areas. Furthermore, in contrast to Anoto's solution, DataGlyph technology is mainly used to encode information on paper documents and is less suited to handwriting capture due to its limited resolution. Note that our interactive paper framework is not restricted to Anoto functionality and can work with any position encoding technology by implementing a few input device interface classes. In addition to Anoto technology, we have worked with regular barcodes and RFID tags as well as a technology for position encoding based on conductive ink developed within the European *Paper++* project [6].

In addition to the PaperPoint presentation tool introduced in this paper, we have developed a variety of interactive paper applications based on the iPaper architecture, including an interactive worksheet for the Natural History Museum in London, mobile tourist guides based on interactive paper maps and event brochures and also the *Print-n-Link* system [11] for the enhanced reading of scientific publications. The set of realised applications covers solutions for enhanced reading, enhanced writing, paper-based interfaces and artistic installations. Detailed information about many of these applications can be found in [15].

PAPERPOINT APPLICATION

In this section, we present the functionality of the PaperPoint presentation tool. Early in the design of the PaperPoint application, we decided that it should not replace existing presentation tools such as PowerPoint but rather provide a new user interface for existing applications. Since PowerPoint is the most widely used presentation tool, we decided to use it as the basis for our paper-based presentation tool. It is important that no additional authoring effort is required to use the tool and users can easily switch back and forth between the paper-based and digital control of the application. The paper-based interface is therefore provided as an optional add-on and users are free to choose whether they want to use it for a given presentation and they can also use it alongside the regular PowerPoint presentation system.

We first describe the steps involved in the process of creating a PaperPoint presentation and then present specific functionality available from the interactive paper handouts. The first step in preparing a paper-based PaperPoint presentation is to create the slides with PowerPoint as usual. The user does not have to add any additional elements to the slides or choose specific commands to make the slides ready to be used with the paper-based presentation tool. It is only when it comes to printing the handouts that an additional step is required. Instead of printing the slides on blank paper, special preprocessed paper sheets have to be loaded into the printer tray. These PaperPoint template pages are numbered in ascending order and have to be loaded into the printer tray in the correct order. On each of these PaperPoint template pages, we print a unique Anoto pattern together with button shapes, which later can be used to control the presentation, as shown on the left hand side of Figure 2. The use of these template pages is the only thing that has to be changed in the production process of a PaperPoint presentation. The handouts are printed on the PaperPoint template page resulting in the interactive paper handouts presented on the right hand side of Figure 2. Note that, currently, we only support this specific layout with six slides but, as we discuss in the next section, it would be a simple task to extend the range of supported handout layouts.

Before giving a presentation with the PaperPoint tool, the user has to specify the location of the PowerPoint file to

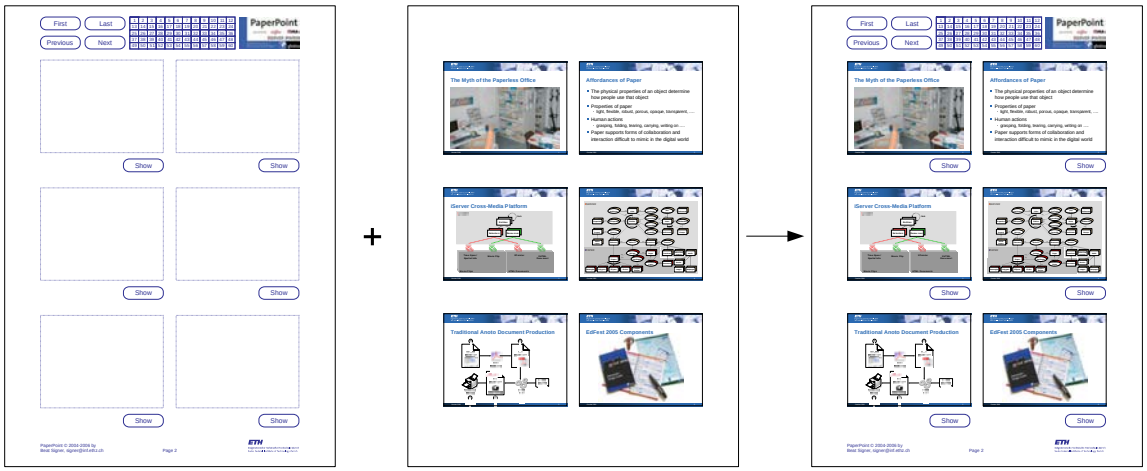

Figure 2. PaperPoint printing process

be used in combination with the interactive paper handouts. After starting the PaperPoint client, the corresponding PowerPoint presentation is started and the iPaper client waits for input from the digital pen. The user has a number of possibilities as to how they can use the paper handout to control the digital presentation. First of all, below the thumbnail of each slide, there is a 'Show' button providing easy access to the corresponding digital slide instance as shown in Figure 3. The printed slide handouts provide an overview of the presentation with each slide embedded in the context of its surrounding slides. When a user selects a slide from the paper handouts, they always have a visual clue about the slides to come next which helps in building smooth transitions between succeeding slides.

By touching the 'Show' button, the presentation immediately switches to the selected slide, enabling the user to quickly skip to any slide regardless of the presentation order. Compared to the conventional PowerPoint user interface, the paper-based PaperPoint control therefore supports a more flexible way of giving presentations. The printed handouts provide a perfect overview of the available slides and, without switching to PowerPoint's slide sorter and interrupting the flow of the presentation, slides can easily be presented in a non-linear order. The presenter can switch from one slide to any other slide by simply selecting the corresponding 'Show' button on their paper handouts without the audience being aware that the slides are not presented in their original order. This is very convenient not only in the case that some slides have to be skipped because of time constraints, but also for accessing specific slides while answering questions coming from the audience or during discussions in meetings.

At the top of each PaperPoint handout page, there are additional 'Next' and 'Previous' buttons for giving linear presentations and a 'First' and 'Last' button to jump to a presentation's first or last slide, respectively. In fact, one could use only the 'Next' and 'Previous' buttons resulting in a slide controller similar to many wireless presentation remote controls which only allow the user to go one step forward or backward in their presentation. In addition, the 'Next' and 'Previous' buttons can be used to control the different steps within a PowerPoint animation. There is also a rectangular area with numbers ranging from 1 to 60 providing direct access to a specific slide.

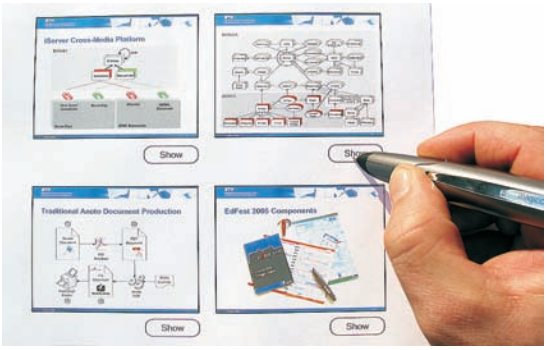

Figure 3. Interactive slide handouts

An evident benefit of the PaperPoint presentation aid is the fact that the presenter no longer has to stand next to a computer to control the slide presentation. The slides can be navigated remotely by pen and paper via a Bluetooth connection and this leaves the presenter with greater freedom in terms of how they move with the space and interact with the audience. However, many other devices such as wireless mice or special presentation aids also allow presentations to be controlled remotely and therefore this is not a unique feature of the paper-based user interface. The fact that it also offers an easy way to annotate slides during the presentation, even remotely, is an additional major benefit.

In highly interactive situations such as meetings and tutorials, it is often useful to be able to annotate existing content or record new content as part of the discussion. This may form the basis for brainstorming or reacting spontaneously

Annotations on paper handouts

Resulting digital slide

Figure 4. Annotations

to questions raised. The blackboards and whiteboards which support such interactions are increasingly being replaced with screens designed to show PowerPoint presentations that are typically prepared in advance and rather static. With the PaperPoint presentation tool, we bring some of the functionality or properties available on a traditional blackboard back to digital presentation tools.

As soon as a presenter starts to annotate one of the slides on the paper handouts by writing within the rectangular box surrounding the slide, the presentation immediately switches to the corresponding digital slide with a response time of less than half a second. The written information is digitised and immediately integrated into the digital presentation by adding it to the appropriate slide. This functionality can be used, not only to highlight existing parts of a slide, but also to add textual information or sketches. An example showing the annotation of a slide is shown on the left hand side of Figure 4. The presenter has written some comments next to the image describing the different components, i.e. 'earpiece', 'booklet', 'map' and 'digital pen'. All the strokes are captured by the digital pen and integrated into the corresponding digital slide in real-time as outlined on the right hand side of Figure 4. We have observed that some presenters also use the annotation functionality as a substitute for the 'Show' buttons by just pointing somewhere within the slide area without leaving any further traces but forcing the annotation service to switch to the corresponding digital slide version.

Note that technically the ink traces left on paper buttons are not a problem since they do not interfere with the Anoto pattern. The digital pens based on Anoto functionality work with an infrared light source to detect the black Anoto dot pattern. This implies that any non-black colour can be used to draw onto the Anoto pattern and the infrared camera will still detect the pattern on colour covered areas. The same robustness is available for the blue ink provided by the pen's writing stylus. Therefore, the user cannot destroy any paper-based functionality by writing on the command buttons since the pen can also detect its position on ink covered regions. However, if a presentation is given several times and marks or annotations become disruptive, a new handout set can easily be printed before the next instance of the presentation.

Although the highlighting of specific parts of a slide could also be done directly in the digital slide version, a major advantage of the paper-based presentation control is the fact that spontaneous annotations can be done more easily and naturally than by switching to the PowerPoint authoring view. Of course, it is also possible to draw on slides using existing technologies such as a computer mouse. However, not many people use this functionality since the mouse was not really invented as a writing tool and it is almost impossible to write text on a slide using a mouse. While other solutions based on a writing stylus and screens offer a significant improvement over the use of a mouse, it is still the case that many users find writing on a screen much more awkward than on paper and these tools are not widely used. Our PaperPoint application brings this functionality back to the appropriate tool for creating handwritten annotations—pen and paper.

While it is convenient to annotate an interactive paper handout lying on a flat surface, it becomes much harder to write on the flexible paper sheets while walking around in a conference room or lecture theatre. Therefore, we have found that many PaperPoint users put their slides on a robust clipboard, as shown in Figure 5, enabling them to write on the slides while on the move. However, the clipboard is an optional tool and we have used different formats in different situations, for example individual paper sheets in meetings and cardboard in mobile environments.

Figure 5. Clipboard with paper controls

In Figure 5, we can see another advantage of the paper user interface. Users can cut out some of the paper widgets and use them to build a customised interface by rearranging the paper snippets and sticking them on a flat surface. For example, the user of the clipboard shown in Figure 5 has added some general controls to their clipboard which allows them to navigate a presentation even if no printed overview is available by just clicking on the 'Previous' and 'Next' paper buttons. This works because the active areas defined on the paper templates are bound to specific functionality such as going to the next, first or last slide. In this way, users can build their own customised interfaces without even having to use a computer or do any programming.

To support this customisation of paper-based interfaces, we provide special empty PaperPoint stickers, covered only with the Anoto pattern, that are linked to specific digital services. Figure 6 shows a sheet of PaperPoint stickers that are all linked to the command that shows a specific slide. Similar stickers are available for the 'Next', 'Previous' and all other commands. A user can design their own user interface by pasting the empty stickers on a surface—for example a blank paper sheet. In a second step, the paper interface may be decorated to highlight the functionality of the paper stickers.

Figure 6. PaperPoint stickers

Another application of the PaperPoint stickers in combination with PowerPoint is the early stage prototyping of interactive paper applications without the need for any programming. Since some of our project partners have no Java programming skills, we were looking for a tool that would enable them to quickly build mockups of interactive paper interfaces. PowerPoint is a widely used application that allows for the integration of different types of media, including movies, sounds, links to web pages or screenshots of arbitrary applications and requires no programming skills. A mockup of an interactive paper interface can easily be built by attaching the PaperPoint stickers to parts of different paper documents or other physical objects and then providing a mockup of the required "functionality" by adding the content to the PowerPoint slide linked to by the corresponding sticker.

Using this PaperPoint sticker-based prototyping approach, it takes only a few minutes to build physical mockups for a given application scenario. Due to the simplicity of the tool and the PowerPoint-based authoring process, our experiments with the use of PaperPoint in project workshops have shown it to be an highly effective tool to physically manifest the results of brainstorming sessions and present them to other team members.

Finally, another powerful feature of the PaperPoint application is the fact that it can be controlled by multiple pens concurrently and not only by a single pen. The system can therefore be used to support collaboration between multiple users. For example, the participants of a group meeting can collaboratively control a presentation and select which slides have to be shown. Each user has a version of the printed handouts and a digital pen allowing them to interact with the PaperPoint application and control the presentation. In addition, all annotations that are written on specific slides can be stored in a database and later accessed in digital format. Especially in decision-making tasks, the collaborative presentation navigator can improve the overall performance by providing fast shared access to the relevant resources.

If we consider starting with blank slides, we get another application scenario where the PaperPoint tool can be used in brainstorming sessions. Each user can produce their own personal ideas and sketches on paper and share them with other users by projecting them onto a common screen. Such a session could start either from totally blank slides or prepared slides with initial thoughts or topics.

IMPLEMENTATION

The first step in implementing the PaperPoint application on top of the iPaper framework was the definition of the active paper regions to be linked to digital information or services. A printed overview slide is shown on the left hand side of Figure 7 whereas the right hand side shows the active areas, defined by rectangular shapes, that have to be stored in the iServer database. For each slide, we have an active area covering the whole slide region that is linked to the annotation functionality as described later. Other active areas are defined for the paper buttons used to access specific slides and control the presentation (e.g. 'Show', 'Next', 'Previous' etc.).

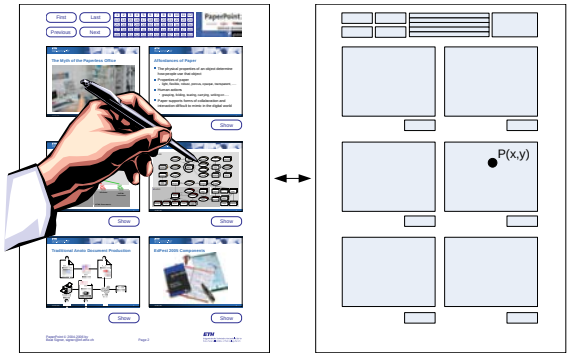

Figure 7. Active paper areas

As mentioned earlier, our interactive paper framework is based on a general cross-media information platform that enables links between arbitrary digital or physical resources based on a resource plug-in mechanism. We now introduce *active content*, a form of resource that was developed for the iServer platform to support the design of complex interaction components as required, for example, by the PaperPoint application to communicate with PowerPoint. While regular

iServer links just return a single piece of information such as an HTML page, a movie or a database object, active content is represented by *active components* which are bound to a piece of Java code that can be executed on either the server or client side. By introducing this simple but effective idea of active components, it becomes possible to link and activate any piece of program logic. Furthermore, it is very easy for developers to implement new functionality by simply providing a specific implementation of the given active component interface. This allows them to concentrate on the specific functionality that they require and extend the framework without having to have any detailed knowledge of the framework itself.

After some input has been captured by a client device—a digital pen in the case of the PaperPoint application—the client-side component has to check for any running active components. Therefore the client distinguishes two working modes: a *browsing mode* where no active component is running on the client side and an *active mode* where an active component has been instantiated and is currently running. Let us assume that no active component is currently running on the client side. An incoming request from the digital pen is sent to the server and handled by iServer. The resolved link target may either be a simple digital resource or an active component. In the case that a link to an active component has been selected, the Active Component Manager will load the corresponding component on the server side. Each active component stored in the database has an identifier and some additional parameters which are used to initialise the Java object at instantiation time. The Active Component Manager will use the active component's identifier to find the corresponding client (*stub*) and server (*logic*) Java class named in the *Active Component Name Directory* and these are then used to create the corresponding Java instances using the Java Reflection API.

Before an active component can be used within the iServer framework, it has to be registered with the Active Component Name Directory. For each active component identifier, a stub and logic binding has to be provided that defines the Java classes to be loaded when an active component is activated. After an active component logic has been loaded on the server side, its XML representation is sent to the client side where an instance of the corresponding active component stub is created and initialised. From then on, all pen input is delegated to the running active component for further processing until the active component is terminated by invoking its `setDone` method.

The PaperPoint application was realised by implementing a set of active components which interact with PowerPoint and no changes to the underlying iPaper framework were required. To access the functionality offered by the PowerPoint Windows application, we used the Java/Win32 (Jawin) [5] integration project. Jawin is an open source architecture for interoperation between the Java programming language and components exposed through the Component Object Model (COM) or Dynamic Link Libraries (DLLs).

DISCUSSION AND FUTURE WORK

The PaperPoint application is a prototype with only the core functionality for controlling PowerPoint presentations accessible from the paper-based presentation tool to date. Currently, the system is in use by members of 4 different institutions for lectures, conference presentations, meetings and early mockups of new interactive paper-based applications.

A number of users have now been working with PaperPoint for more than a year. In informal discussions, multiple users reported that the new flexibility provided by the tool also changed the way in which they prepare their PaperPoint presentations. Since the implicit linearity of PowerPoint presentations no longer exists when using the PaperPoint handouts, users tend to add additional slides in preparing their presentations even if they are not sure whether they will finally show them. Some users already integrated these slides at the positions where they would probably show them, while others just added them at the end of their presentations as a kind of backup slides that could easily be integrated on demand at presentation time.

Users also reported favourably on the use of the system as a means of providing flexible presentations of projects based on large collections of related slides. Instead of having to prepare presentations in advance by copying and pasting selected slides into a new presentation suited to a particular audience, presenters could easily select slides from a printed overview of the collection at presentation time. Similarly, when presentations had to be prepared that involved more than one presenter, they could simply integrate their slides into a single file and then it was easy to dynamically switch between presenters and their chosen slides during a presentation. In such situations, the transition between slides was much smoother than the typical actions of searching for and skipping over slides seen in presentations controlled from a laptop and the audience was often unaware that the presentations were being adapted to the current context on the fly. However, while this worked well when users were very familiar with the collection of slides and more experienced in giving presentations, in some cases, it resulted in the presenters being under-prepared in terms of planning the structure and content of a presentation. Also, sometimes instead of adding a few optional or extra slides to a presentation, they would have far too many slides and spend too much time selecting slides during the presentation.

We plan more elaborate user studies to analyse not only how the application is used during presentations, but also how it effects the preparation and quality of presentations.

We are currently extending the set of available PaperPoint active components, including paper widgets to control embedded movies and sound clips. The same approach of using the Jawin library can be used to access other Windows applications and we already have other active components to control Microsoft Access, web browsers etc. Note that the idea of customisable paper-based user interfaces is not limited to a single application. In fact, functionality offered by

different applications may be accessed from a single paper document, thereby integrating paper with digital functionality provided by various applications.

The printing process is another area that could be improved in the future. At the moment, the user has to load the correct PaperPoint template sheets in their printer tray before printing the handout generated from PowerPoint. We are currently developing a special virtual printer driver which will allow us to print the handouts together with the automatically generated Anoto pattern in a single step.

Last but not least, we plan to use some of the features of the underlying iServer platform for collaborative information sharing. One possibility is to support the sharing of PaperPoint annotations in an educational setting. While the teacher presents the lecture material using the PaperPoint presentation tool, each student will be able to have a copy of the interactive handouts and a personal PaperPoint instance running on their laptop. During the lecture, students could annotate their own slides as usual but, since all information is also captured electronically, they will be able to access digitised notes of other students after the lecture. Not only the students could profit from this exchange of annotations, but also the teacher who could use this information to get a better understanding of upcoming questions or potentially unclear parts of their lecture material.

CONCLUSION

We have presented a paper-based presentation tool that successfully integrates paper and digital services by combining the strengths of both media. While the interactive PaperPoint handouts provide a good presentation overview and can be easily annotated, the PowerPoint application provides a rich set of functionality for dynamic presentation effects and digital media can be used to store, access and share annotations. In addition, we have introduced the concept of PaperPoint stickers and active components which can support the rapid prototyping of paper interfaces for arbitrary digital applications, thereby bridging the paper-digital divide.

REFERENCES

1. G. D. Abowd, J. Brotherton, and J. Bhalodia. Classroom 2000: A System for Capturing and Accessing Multimedia Classroom Experiences. In *Proc. of CHI '98, ACM Conference on Human Factors in Computing Systems*, pages 20–21, Los Angeles, USA, April 1998.
2. R. Anderson, R. Anderson, B. Simon, S. A. Wolfman, T. VanDeGrift, and K. Yasuhara. Experiences with a Tablet PC Based Lecture Presentation System in Computer Science Courses. In *Proc. of SIGCSE '04, 35th Technical Symposium on Computer Science Education*, pages 56–60, New York, USA, March 2003.
3. Anoto AB, http://www.anoto.com.
4. D. L. Hecht. Printed Embedded Data Graphical User Interfaces. *IEEE Computer*, 34(3):47–55, March 2001.
5. Jawin - A Java/Win32 Interoperability Project, http://jawinproject.sourceforge.net/.
6. P. Luff, C. Heath, M. C. Norrie, B. Signer, and P. Herdman. Only Touching the Surface: Creating Affinities Between Digital Content and Paper. In *Proc. of CSCW 2004, ACM Conference on Computer Supported Cooperative Work*, pages 523–532, Chicago, USA, November 2004.
7. T. Masui and I. Siio. Real-World Graphical User Interfaces. In *Proc. of HUC 2000, 2nd International Symposium on Handheld and Ubiquitous Computing*, pages 72–84, Bristol, UK, September 2000.
8. B. A. Myers, H. Stiel, and R. Gargiulo. Collaboration Using Multiple PDAs Connected to a PC. In *Proc. of CSCW '98, ACM Conference on Computer Supported Cooperative Work*, pages 285–294, Seattle, USA, November 1998.
9. L. Nelson, S. Ichimura, E. R. Pedersen, and L. Adams. Palette: A Paper Interface for Giving Presentations. In *Proc. of CHI '99, ACM Conference on Human Factors in Computing Systems*, pages 354–361, Pittsburgh, USA, May 1999.
10. M. C. Norrie, B. Signer, and N. Weibel. General Framework for the Rapid Development of Interactive Paper Applications. In *Proc. of CoPADD 2006, 1st International Workshop on Collaborating over Paper and Digital Documents*, pages 9–12, Banff, Canada, November 2006.
11. M. C. Norrie, B. Signer, and N. Weibel. Print-n-Link: Weaving the Paper Web. In *Proc. of DocEng 2006, ACM Symposium on Document Engineering*, pages 34–43, Amsterdam, The Netherlands, October 2006.
12. E. R. Pedersen, T. Sokoler, and L. Nelson. PaperButtons: Expanding a Tangible User Interface. In *Proc. of DIS 2000, Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, pages 216–223, New York City, USA, August 2000.
13. J. Rekimoto. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proc. of UIST '97, 10th Annual Symposium on User Interface Software and Technology*, pages 31–39, Banff, Canada, October 1997.
14. A. J. Sellen and R. Harper. *The Myth of the Paperless Office*. MIT Press, November 2001.
15. B. Signer. *Fundamental Concepts for Interactive Paper and Cross-Media Information Management*. PhD thesis, ETH Zurich, 2006.
16. I. Siio, T. Masui, and K. Fukuchi. Real-world Interaction using the FieldMouse. In *Proc. of UIST '99, 12th Annual ACM Symposium on User Interface Software and Technology*, pages 113–119, Asheville, USA, November 1999.
17. P. Wellner. Interacting with Paper on the DigitalDesk. *Communications of the ACM*, 36(7), July 1993.