Interactive Paper as a Reading Medium in Digital Libraries

Moira C. Norrie, Beat Signer, and Nadir Weibel

Institute for Information Systems
ETH Zurich
8092 Zurich, Switzerland
{norrie,signer,weibel}@inf.ethz.ch

Abstract. In digital libraries, much of the reading activity is still done on printed copies of documents. We show how digital pen and paper technologies can be used to support readers by automatically creating interactive paper versions of digital documents during the printing process that enable users to activate embedded hyperlinks to other documents and services from printed versions. The approach uses a special printer driver that allows information about hyperlinks to be extracted and stored at print time. Users can then activate hyperlinks in the printed document with a digital pen.

1 Introduction

Even in digital libraries, paper still plays an important role as many users print documents before reading them. Various studies have analysed the affordances of paper behind this preference for reading on paper [1]. One notable reason is the ease with which paper documents can be annotated in various ways. However, once a document is printed, it stands in isolation having lost its connections to other documents and services that have become characteristic of digital libraries. Specifically, embedded hyperlinks in HTML, PDF or MS Word documents are no longer active and often invisible in the printed versions.

We have shown previously how digital pen and paper technologies can be used to create interactive paper documents that link the paper and digital worlds [2]. We believe that these technologies could also be beneficial for readers in digital libraries by automatically creating interactive paper versions of digital documents during the printing process. In this paper, we describe how we have achieved this by developing a special printer driver that allows us to extract and store hyperlink information at the time of printing. Users can then activate hyperlinks on paper using a digital pen. At the same time, handwritten annotations can be digitally captured and later accessed in the digital world.

The main issue was how to provide a general model and framework that allows hyperlink definitions for paper documents to be generated on demand at the time of printing. The anchors of the hyperlinks are shapes within a physical page defined in terms of (x,y) coordinates based on the physical layout of the printed document which may or may not correspond to that of the digital document.

B. Christensen-Dalsgaard et al. (Eds.): ECDL 2008, LNCS 5173, pp. 232–243, 2008.

Nowadays it is common for content management systems to provide special print formats for documents. Also, users may control certain features of the digital rendering through font selections, window re-sizing etc. We present a general approach for creating hyperlink definitions within paper documents based on an analysis of the digital source and describe how this has been integrated into a general framework for interactive paper documents.

Section 2 provides an overview of related work. Interactive paper technology is introduced in Sect. 3. Our approach and the underlying architecture are presented in Sect. 4, while the mechanisms used to define and generate hyperlinks within printed pages are described in Sect. 5. Concluding remarks are given in Sect. 6.

2 Related Work

The integration of actions performed on paper documents with digital media was investigated in the DigitalDesk project [3] where a camera mounted over a desk was used to track a user's hand movements on paper documents and translate them into digital actions. The Origami project [4] extended the DigitalDesk approach by dealing with interactions from printed web pages. In this system, HTML documents are rendered as postscript files and the positions and link targets of all embedded hyperlinks are recorded in the application's registry. If the printout of such an HTML document is used on the DigitalDesk and detected by the over-desk camera, links can be activated by pointing to the specific positions within the paper document and the corresponding digital web page is projected onto the desk next to the printed document. Both of these systems require a complex infrastructure and are limited in terms of mobility.

Other research projects and commercial products focus on the identification of, and interaction from, single elements within a paper document such as pages or paragraphs. Both the PaperClick[1] and the Wiziway[2] applications allow the creation of links on paper defined by means of printed place holders such as linear barcodes or pictograms. Other approaches, including the InfoPoint project [5] or more recent commercial products such as ScanLife[3], make use of 2D-barcodes placed on paper sheets which can be captured with regular digital cameras available in mobile phones.

Recent research projects such as PaperPoint [6] and Print-n-Link [7] have demonstrated how digital pen and paper technologies such as Anoto[4] can be used to turn paper documents into interactive objects. A hyperlink within a paper document is defined in terms of an active area and the action of touching the paper with a digital pen within this area activates the link. This technology provides the potential for users to interact with printed documents using the digital pen in a way that closely mimics interaction in a normal desktop browser using a mouse. PaperPoint allows PowerPoint presentations to be controlled and

[1] http://www.paperclick.com
[2] http://www.wiziway.com
[3] http://scanbuy.com
[4] http://www.anoto.com

annotated from printed handouts, while Print-n-Link allows users to search for scientific publications on the web and then print interactive paper versions that give information about citations through an audio channel and can activate web searches for cited documents.

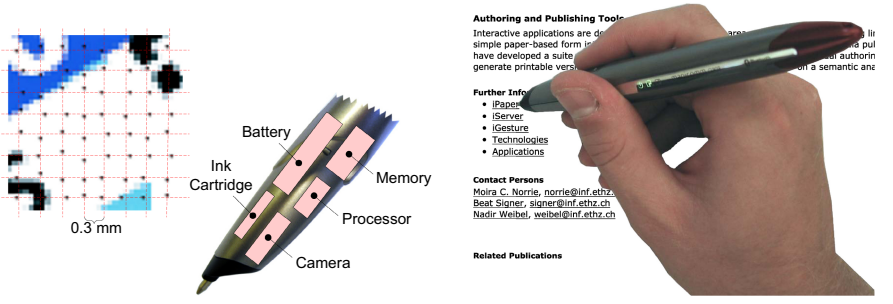
ProofRite [8] is another system based on Anoto technology that maps between digital and paper versions of the same document. It allows free-form annotations on the paper version of a text document to be included at the same physical location within the digital source document.

Ideally, users of digital libraries should be able to print any form of document such as web pages, PDF files and MS Word documents and immediately start interacting with these using a digital pen. While this may seem simple compared to some of the systems described above, it is important to consider the exact nature of the mapping between the digital and physical documents required in each case. In the case of PaperPoint, templates were manually authored that link specific areas on a presentation handout to fixed actions in the PowerPoint application such as showing the first slide or the next slide. So when a presentation handout is printed, all links are pre-defined and the system simply has to add the required Anoto pattern. Other systems such as Print-n-Link only deal with PDF documents where there is a direct correspondence between positions on paper and positions within the digital document. Similarly, in ProofRite, handwritten annotations are captured and simply embedded into the digital version of that document at the same positions. So in the case of ProofRite and Print-n-Link, there is a simple one-to-one mapping between the digital and paper documents.

Our goal was to be more general and not only handle cases where there is a one-to-one mapping between the digital and paper versions of documents. This happens more and more often nowadays as content management systems provide special print formats for digital documents or users want to control features of the digital or physical renderings. This means that we had to develop a general printing tool that would analyse documents during the printing process to extract information about hyperlinks and generate corresponding link definitions within an interactive paper framework.

3 Interactive Paper

A number of digital pens based on Anoto functionality are now available commercially. These pens work on an almost invisible dot pattern that encodes (x,y) positions within a vast virtual document space. A camera within the digital pen processes information encoded within the dot pattern in real-time resulting in up to 70 (x,y) positions per second. The technology was originally developed for the digital capture of handwriting. Multiple handwritten pages can be captured and stored within the pen before being transferred to a PC via a Bluetooth or USB connection. Hitachi Maxell and Logitech have recently released digital pens based on Anoto functionality that can also be used in streaming mode where position information is transmitted continuously. This



(a) Anoto technology (b) Pointing to a printed page

Fig. 1. Interactive paper web page

enables the pens to be used for real-time interaction as well as writing capture. Figure 1(a) provides an overview of the basic Anoto technology.

Various frameworks have been developed to deal with interactions from an Anoto-enabled paper document. The Anoto SDK [9] supports the simple post-processing of the data captured by the digital pen within a paper form, while the PaperToolkit framework [10] created at Stanford University, the PADD system [11] from the University of Maryland and the iPaper/iServer infrastructure [2] developed at ETH Zurich allow more complex real-time interactions to be managed. By using a general link server for interactive paper such as the iServer platform, it is possible to define active areas on paper and bind them to digital resources such as images, videos and web pages or to digital services. The digital pen can then be used in much the same way on paper documents as a mouse would be used during web browsing sessions to follow links to other web resources or trigger application calls.

In order to select active areas on paper, these areas first have to be authored and the paper document has to be Anoto-enabled by covering it with the corresponding dot pattern. In Print-n-Link [7], interactive paper versions of scientific papers available in PDF can be created automatically at the time of printing by simply selecting a special printer driver that will add Anoto pattern allocated dynamically from the available pattern space. The necessary active areas are automatically generated by analysing the PDF document at printing time for information about citations. In the rest of the paper, we describe how we have generalised this approach to deal with cases where there is not a simple one-to-one mapping between the digital document and the physical rendering on paper.

4 Printer Driver for Interactive Paper Documents

To embed hyperlinks within the printed instance of a document and define the corresponding active areas, we have to identify the exact position on paper of all link elements (e.g. a word, a sentence or an image). We therefore intercept the publishing process, extract all link information before printing and map

these to shapes within the paper instance. The exact positions on paper are determined from the physical representation of the document specified in a Page Description Language (PDL). The main objective of PDLs is to enable the independence of a document from the software used to render it on the screen, the operating system and the printing device: A document should not differ when displayed on different machines or printed on different printers. The best known PDLs are Adobe's PostScript (PS) [12] and Portable Document Format (PDF) [13], but there exist other emerging formats such as the Microsoft XML Paper Specification (XPS) [14], an XML-based paginated representation of a document that is based on the Microsoft Extensible Application Markup Language (XAML) and released with Microsoft Vista and Office 2007. Our approach may be applied to different PDLs, however, in this paper, we describe how it was implemented based on XPS.

To examine the process in detail, we will consider the case of printing HTML web pages since nowadays it is frequently the case that the printed versions differ significantly from the digital versions and there is not a simple one-to-one mapping between them. We note, however, that the approach and architecture are general and different types of documents can be handled by integrating appropriate plug-ins. Our PaperWeb plug-in enables the printing of interactive web pages in such a way that the paper documents are automatically augmented with Anoto functionality and all hyperlinks can be activated using a digital pen. Thus, a user may point to a *paper-link*—a hyperlink on paper—activate its target and open the linked page within a desktop browser. Currently the system works with the Mozilla Firefox browser.

The basic architecture of PaperWeb is outlined in Fig. 2. The browser first locates the hyperlinks and then sends the complete XHTML Document Object

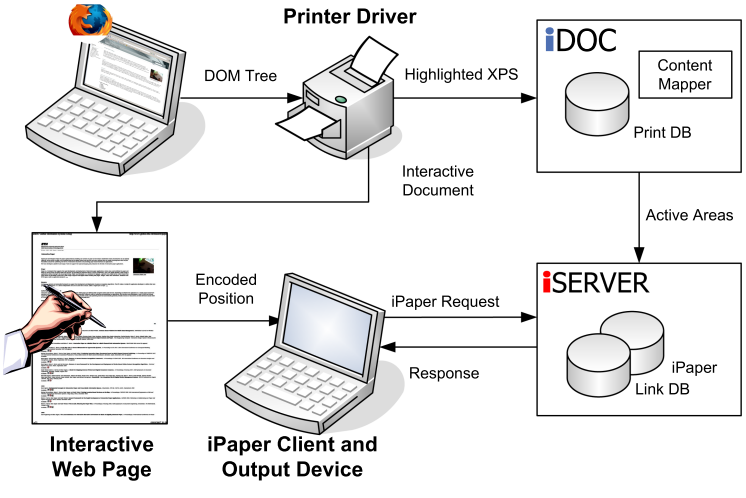


Fig. 2. PaperWeb architecture

Model (DOM) to our special printer which generates an XPS version of the web page.

In order to extract the hyperlink information, we implemented a Firefox extension which is activated when the user prints an interactive web page. The extension analyses the DOM tree of the XHTML document and retrieves all hyperlinks. In the current implementation, we look for `<a href>` (anchor) and `<area>` tags, but the system can easily be extended to support other elements. During the printing process, the printer driver augments the document with Anoto pattern and sends it to the physical printer. The XPS representation that contains the retrieved hyperlinks is forwarded to the iDoc framework [15] which extracts both the information about the position and target of each link. We implemented a specific content analyser for the iDoc *content mapper* that is responsible for mapping physical elements rendered in the XPS document to their digital counterparts within the web page. By analysing the XPS document in terms of the the shapes and tags that it contains, the semantic mapper creates a link between the printed instances of every linked element and its representation within the digital source document. For every hyperlink, a paper-link is created by publishing its position and size together with the target address to the interactive paper plug-in (iPaper) of the iServer platform.

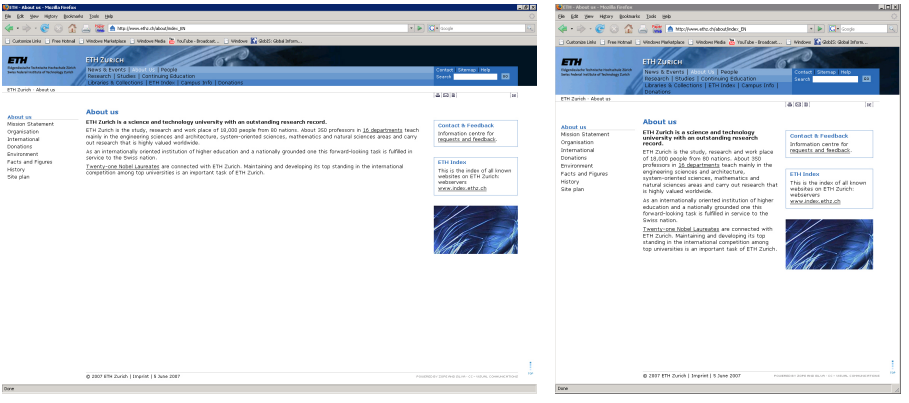
Once the document is printed, users may immediately interact with the paper version using a digital pen. Pen positions on paper are transmitted via Bluetooth to the iPaper client which forwards a request to iServer. If the position is within a paper-link active area, then the link target will be displayed in the user's browser. If the pen position is elsewhere on a page, it can be assumed that the document is being annotated and a special capture component called. The capture is completed when the pen leaves the paper for a fixed time or touches a paper-link. One of the issues is what happens to such annotations. A minimal approach is to simply record them and attach them to a page. More advanced solutions may attach them to specific elements within a page or to even interpret them as commands as is done in the paper-based editing system PaperProof [16]. Further, the issue of where they will be stored and how they will be visualised will very much depend on the type of the document and also the general architecture of the system. For example, in the case of a PDF document, it would be possible to attach the annotation to the corresponding position within the digital version. In the case of a web page, it assumes that a browser is used that supports annotation, for example Amaya[5]. It is beyond the scope of this paper to discuss these options in detail, but we emphasise that while it is a straightforward matter to capture handwritten annotations with a digital pen, the complexities come from how they are then handled within a particular digital library system.

5 Paper Hyperlinks

Web designers often define specific layouts of a web page allowing web authors to add text, graphical content or animations in predefined regions. Web page

[5] http://www.w3.org/Amaya/

layouts help to define the appearance of a page in a way that is independent of a web browser, the client's display resolution or the size of the windows. However, this implies that the information contained within the predefined regions may reflow in order to adapt to a client's preferences. Figure 3 shows the same web page rendered in two different sizes within a Firefox web browser. Note that while the left and right columns remain more or less the same, there is a reflowing of the text in the middle column.



(a) Maximised web page (b) Resized web page

Fig. 3. Different page representations

When printing a web page, its representation changes again depending on the web browser used and the paper size. The physical representation of the page on paper normally differs from the one displayed on the screen, since some elements such as the background or menus may be removed and the font size adapted to the paper size. A printed version of the web page in Fig. 3 is shown in Fig. 4 and, as one can see, there is a simpler version of the header component without graphics and links, and no menu column.

In order to access the hyperlinks from paper, we have to determine their physical position at the time of printing. Since we cannot rely on the representations of the web page displayed on the screen, we need to be able to map elements within a digital representation of a document to areas within the printed instance. We use a two-step process that consists of first highlighting link elements within the digital document to create shapes that can later be retrieved to define the active areas within a physical page.

Hyperlink Highlighter

The XHTML DOM model stores all elements and objects contained within a web page as a tree. The hyperlinks are also stored within the DOM tree and are accessible by parsing the tree. Regardless of the object representing the hyperlink

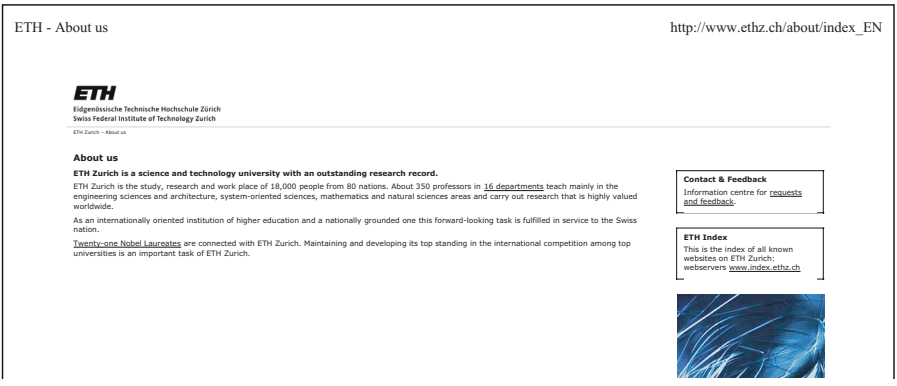


Fig. 4. Printed web page version

(text, graphic, video, etc.), it is always possible to locate it and extract the target address of that hyperlink.

Since information about hyperlinks is lost during the printing process, we introduce the concept of a *highlighter* to generate graphical shapes around link elements. Our Firefox extension analyses the DOM tree during the publishing process, extracts information about the document structure and defines bounding boxes for each hyperlink based on the JavaScript code shown in Figure 5.

For each link element, a PaperWeb division element (`<div>`) is created and special style commands are applied to create the highlighting box. The bounding boxes are invisible for the user but are rendered into the paginated XPS version of the web page. The hyperlink element is copied from the DOM tree and included in the PaperWeb `<div>` element. We also need the target address of the original web page hyperlink which is extracted from the DOM anchor element and inserted as a new division element into the existing PaperWeb `<div>` element. This second division element contains a textual, visible representation of the target address and is positioned in the top-left corner of the parent `<div>` element. After the PaperWeb division element has been defined, it is placed before the original anchor element within the DOM tree and the anchor element is then removed.

After the highlighting phase, the web page is transformed into a physical representation through XPS. Since the highlighting shapes are part of the web page, they are not lost during the printing process and may be retrieved from the physical web page representation. Figure 6 shows an extract of a web page during the highlighting phase. Note that, in the figure, the highlighted shapes have been emphasised by the light blue colour but normally they are not visible to the end user.

Hyperlink Retriever

Having generated a physical representation of the web page containing the highlighted hyperlinks, we still need to retrieve their position and target

```
function paperweb(){
  var hrefs = window.content.document.getElementsByTagName('a');
  var areas = window.content.document.getElementsByTagName('area');
  highlight(hrefs);
  highlight(areas);
}

function highlight(els){
  for(var i = 0; i < els.length; i++){
    var href=els[i];
    var target = href.href;
    var targetNode= window.content.document.createTextNode("idocRes:"+ target);
    var my_node = href.cloneNode(true);
    var div_paperweb = window.content.document.createElement('div');
    var div_link = window.content.document.createElement('div');
    div_paperweb.setAttribute('class', 'paperweb');
    div_paperweb.setAttribute('style', 'position:relative; display: inline;
                          background-color:lightblue;');
    div_link.setAttribute('class', 'paperweb_link');
    div_link.setAttribute('style', 'display:inline; position:absolute; color:red;
                        font-size:1px; left:0; top:0;');
    div_link.appendChild(targetNode);
    div_paperweb.appendChild(my_node);
    div_paperweb.appendChild(div_link);
    href.parentNode.insertBefore(div_paperweb, href);
    href.parentNode.removeChild(href);
  }
}
```

Fig. 5. PaperWeb Firefox extension

addresses in order to generate link definitions that can be imported into the iPaper/iServer platform for interactive paper. We first locate the highlighted hyperlinks by analysing the XPS, then calculate their exact position on the page and finally extract the information about the target address.

An XPS file is basically a ZIP archive containing a set of XML files—one file for each physical page—describing the structure of the page on a graphical level. All fonts and embedded images are included in the ZIP archive. `Glyphs` elements are used within XPS to represent a sequence of uniformly formatted text and `Path` elements are used to add vector graphics and raw images to an XPS page. Figure 7 shows an example of the XML representation of a `Glyphs` and a `Path` XPS element. The position of a `Glyphs` element is characterised by the `OriginX` and `OriginY` attributes. The `FontUri` attribute defines the font to be used, `Indices` represents the distances between single characters composing a glyph and `UnicodeString` contains the glyph's actual text content. The geometry and the position of graphics is handled by the `Data` attribute of `Path` elements.

The blue boxes visible in Fig. 6 are rendered in XPS with a `Path` object and defined as four points within the `Data` element as shown in Fig. 7(b). The hyperlink's target is rendered as a `Glyphs` object containing a special `iDocRes`

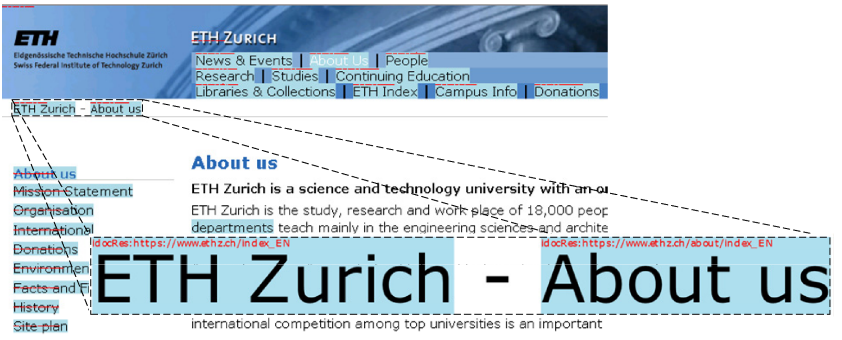


Fig. 6. Highlighted hyperlinks

```
<Glyphs
  Fill="#ff000000"
  FontRenderingEmSize="4.32017"
  FontUri="/Documents/1/Resources/Fonts/
    083FE3E4-4F2E-85D9-B082CEBD4F5E.odttf"
  StyleSimulations="None"
  OriginX="54.08"  OriginY="105.92"
  Indices="40;55;43;3;61;88;85;76;70;75"
  UnicodeString="ETH Zurich"/>
```

(a) `Glyphs` element

```
<Path
    Fill="#ffACDCE8"
    Data="F1 M 54.08,101.6 L 78.08,
          101.6 78.08,106.72 54.08,106.72 z"
/>
```

(b) `Path` element

Fig. 7. XPS element representation

identifier. By analysing the position and the size of the blue boxes, it is easy to find the objects that it contains. Every box contains exactly two elements: the hyperlink target rendered as a glyph, which may be identified through the `iDocRes` identifier, and a second element which is either a `Glyphs` or a `Path` element, depending on the hyperlink's source element (text or graphics). By iterating over all boxes, we can identify all hyperlinks defined within the source web page and extract the position and size of their bounding boxes. This information is finally exported to iServer which is responsible for mapping the active paper area defined by the bounding box to the corresponding digital service which will open the target web page within a web browser.

For this purpose, three iServer/iPaper entities are created: a rectangular *shape*, an *active component* and a *link*. The shape element defines the exact dimensions and position of the box within the printed document page. The active component specifies the service to be called if a user touches a shape with the digital pen. We use a `BROWSER` active component specifying the target web address to be opened within the default browser. Finally, a link entity is used to associate the shape with the corresponding active component. Figure 8 shows an XML extract of a single PaperWeb link definition within the iServer/iPaper framework.

```
<rectangle id="paperweb_shape_eth" creator="paperweb" layer="linkLayer"
            resource="paperweb_eth_1">
  <name>PaperWeb Rectangle for 'ETH Zurich'</name>
  <upperLeft>
    <point><x>14</x><y>27</y></point>
  </upperLeft>
  <size><width>6</width><height>1</height></size>
</rectangle>

<activeComponent id="paperweb_ac_eth" creator="paperweb">
  <name>PaperWeb AC for 'ETH Zurich'</name>
  <properties>
    <parameter>
      <key>org.ximtec.iserver.ac:uri</key>
      <value>https://www.ethz.ch<index_EN/value>
    </parameter>
  </properties>
  <identifier>org.ximtec.iserver.activecomponent.BROWSER</identifier>
</activeComponent>

<link id="paperweb_link_1" creator="paperweb" sources="paperweb_shape_eth"
      targets="paperweb_ac_eth">
  <name>PaperWeb Link for 'ETH Zurich'</name>
</link>
```

Fig. 8. iServer XML link definition

6 Conclusions

We have described a general approach for creating interactive paper versions of digital documents during the printing process based on digital paper and pen technologies. This allows users to read documents on their preferred medium of paper without losing the links to other documents and services typically represented by embedded hyperlinks. Details were given of a specific plug-in that enables web pages to be printed as interactive paper documents. The demonstration of the PaperWeb solution includes the presentation of the different processing steps as well as digital pen-based interaction with interactive printed web pages.

References

1. Sellen, A.J., Harper, R.: The Myth of the Paperless Office. MIT Press, Cambridge (2001)
2. Norrie, M.C., Signer, B., Weibel, N.: General Framework for the Rapid Development of Interactive Paper Applications. In: Proceedings of CoPADD 2006, 1st International Workshop on Collaborating over Paper and Digital Documents, Banff, Canada, pp. 9–12 (November 2006)

3. Wellner, P.: Interacting With Paper on the DigitalDesk. Communications of the ACM 36(7) (July 1993)
4. Robinson, P., Sheppard, D., Watts, R., Harding, R., Lay, S.: Paper Interfaces to the World-Wide Web. In: Proceedings of WebNet 1997, World Conference on the WWW, Internet & Intranet, Toronto, Canada (November 1997)
5. Kohtake, N., Rekimoto, J., Anzai, Y.: InfoPoint: A Device that Provides a Uniform User Interface to Allow Appliances to Work Together over a Network. Personal and Ubiquitous Computing 5(4), 264–274 (2001)
6. Signer, B., Norrie, M.C.: PaperPoint: A Paper-Based Presentation and Interactive Paper Prototyping Tool. In: Proceedings of TEI 2007, First International Conference on Tangible and Embedded Interaction, Baton Rouge, USA, pp. 57–64 (February 2007)
7. Norrie, M.C., Signer, B., Weibel, N.: Print-n-Link: Weaving the Paper Web. In: Proceedings of DocEng 2006, ACM Symposium on Document Engineering, Amsterdam, The Netherlands (October 2006)
8. Conroy, K., Levin, D., Guimbretière, F.: ProofRite: A Paper-Augmented Word Processor. In: Demo Session of UIST 2004, 17th Annual ACM Symposium on User Interface Software and Technology, Santa Fe, USA (October 2004)
9. Anoto, A.B.: Development Guide for Service Enabled by Anoto Functionality (February 2006)
10. Yeh, R.B., Klemmer, S.R., Paepcke, A.: Design and Evaluation of an Event Architecture for Paper UIs: Developers Create by Copying and Combining. Technical report, Stanford University, Computer Science Department (2007)
11. Guimbretière, F.: Paper Augmented Digital Documents. In: Proceedings of UIST 2003, 16th Annual ACM Symposium on User Interface Software and Technology, Vancouver, Canada, pp. 51–60 (November 2003)
12. Adobe Systems Inc.: PostScript Language Reference Manual
13. Adobe Systems Inc.: PDF Reference, Adobe Portable Document Format. 5th edn. Version 1.6 (February 2006)
14. Microsoft Corporation: XML Paper Specification, Version 1.0 (October 2006)
15. Weibel, N., Norrie, M.C., Signer, B.: A Model for Mapping between Printed and Digital Document Instances. In: Proceedings of DocEng 2007, ACM Symposium on Document Engineering, Winnipeg, Canada (August 2007)
16. Weibel, N., Ispas, A., Signer, B., Norrie, M.C.: PaperProof: A Paper-Digital Proof-Editing System. In: Proceedings of CHI 2008, ACM Conference on Human Factors in Computing Systems, Florence, Italy (April 2008)