Content Publishing Framework for Interactive Paper Documents

Moira C. Norrie
Department of Computer Science
ETH Zurich
CH-8092 Zurich, Switzerland
norrie@inf.ethz.ch

Alexios Palinginis
Department of Computer Science
ETH Zurich
CH-8092 Zurich, Switzerland
palinginis@inf.ethz.ch

Beat Signer
Department of Computer Science
ETH Zurich
CH-8092 Zurich, Switzerland
signer@inf.ethz.ch

ABSTRACT

Paper persists as an important medium for documents and this has motivated the development of new technologies for interactive paper that enable actions on paper to be linked to digital actions. A major issue that remains is how to integrate these technologies into the document life cycle and, specifically, how to facilitate the authoring of links between printed documents and digital documents and services. We describe how we have extended a general web publishing framework to support the production of interactive paper documents, thereby integrating paper as a new web channel in a platform for multi-channel access.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.11 [**Software Engineering**]: Software Architectures

General Terms

Design, Experimentation

Keywords

Interactive Paper, Publishing Framework

1. INTRODUCTION

Despite the ubiquity of computers, paper still persists as a universal medium for the capture and publication of information. While certain activities in the document life cycle, such as authoring, distribution and archiving, tend to have migrated to the digital world, others such as reading are still frequently based on paper [20]. The reading activity can often be a detailed and complex task that involves some form of annotation of a document and possibly combining and comparing information within or across documents [16]. Reading may be an activity that occurs away from our normal working place, for example, at a meeting or during our travels. Paper offers many advantages over digital media when these kinds of activities are considered. It is easy to annotate printed documents in several ways such as highlighting, writing comments in margins or augmenting with post-its [13]. Paper is cheap, light, foldable and requires no power, therefore making it an ideal medium in mobile environments [14]. It supports complex information analysis tasks due to the ability to easily view several documents simultaneously, altering spatial relationships as and when needed, and possibly also taking notes.

Researchers have recognised that paper documents play an important role in current work practices and, for many years, have addressed the challenge of how to link actions on paper documents to actions in the digital world. Thus we may want to digitally capture handwriting and document mark-up or to be able to invoke digital services directly from paper. One particular case of the latter is to consider extending the scope of the web to allow links to and from elements within printed documents. Recent technologies developed within the commercial and research sectors have made major advances in this direction and the concept of *interactive paper documents* is now a reality and no longer simply a dream. The issue has now become one of how best to exploit these technologies and integrate them into the document life cycle.

Emerging technologies for interactive paper open up a whole new world of possibilities for the integration of printed documents and digital services. However, if these technologies are to reach their full potential, it is important that supporting infrastructure be provided to enable applications to be developed and deployed with minimal effort. Just as we have seen with the web, it is not sufficient to provide content and link authoring tools alone. We have now reached a state where content management systems and web publishing frameworks are seen as a necessity to develop, operate and maintain a web site of any reasonable size. At the same time, the desire to publish the same content in different forms has seen content management systems take over the role of document production in some domains. We are therefore interested in developing, not only servers to support interaction from paper, but also cross-media content publishing frameworks.

In this paper, we describe how we extended a general web publishing framework (OMSwe) to support the generation of interactive paper documents. OMSwe [17] uses a special versioning mechanism to support multi-channel, context-aware web access and, in effect, interactive paper was added as

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DocEng'05, November 2–4, 2005, Bristol, United Kingdom.
Copyright 2005 ACM 1-59593-240-2/05/0011 ...$5.00.

another channel. It differs from other channels in that the publishing process for interactive paper involves the generation of both a PDF document to be printed and a special XML file with information on active areas that is used to create the definition of links in a special cross-media link server (iServer). To explain the technologies for interactive paper and the publishing process, we use an example of an interactive festival brochure that was developed for visitors to the Edinburgh Fringe festival in 2004.

Section 2 gives a brief introduction to technologies for interactive paper, including an overview of a general server framework that we developed to support links between paper and digital resources. The main features of the OMSwe web publishing platform are presented in Section 3 and Section 4 discusses how it was extended to support the publication of interactive paper documents. Section 5 then describes how OMSwe and iServer were coupled to support the interaction from these documents. Section 6 reports on our experiences with the publishing system and application described in the paper and how these have influenced our on-going and future work. Concluding remarks are given in Section 7.

2. INTERACTIVE PAPER

In this section, we start with a discussion on the potential benefits of interactive paper in tourist applications. We then follow this with a brief introduction to the technologies for interactive paper on which this work is based.

Tourism is a domain with considerable potential for the use of mobile technologies and a number of projects have developed PDA-based tourist guides, for example, Georgia Tech's Cyberguide [1], the Lanacaster GUIDE system [9] and Xerox Parc's electronic guide [26]. Although tourist guides for PDAs are now commercially available, studies of tourists show that paper maps and guides are still considered the essential tourist accessories. To understand why this is the case, it is necessary to study *how* tourists work with maps and guidebook. Our main interest is tourists on the move and therefore how tourists refer to maps, guidebooks and other forms of printed material during a visit to locate themselves, plan activities and find out more about particular places of interest and events. Tourists spend a lot of time combining and comparing information. It is therefore important that they can easily switch back and forth between options and display related information such as maps and descriptions simultaneously [6]. It is common to see tourists working with several documents simultaneously as shown in Figure 1 where one tourist holds a map while the other holds a guidebook and pointing is used to link the two.

The display option for electronic guidebooks is often a PDA. Comparison of information is rarely offered, mainly because of limited screen size. Further, the reduced size or coverage of maps can make it difficult for users to locate themselves relative to other areas of a city and to plan general directions of wandering. For this reason, some projects have opted for the use of Tablet PCs to enable them to provide much richer information environments to the tourists [7]. Clearly the problem here is that, with current technologies, Tablet PCs are still relatively heavy if they are to be carried for a significant amount of time and they require the use of both hands. It is important to remember that tourists also want to have hands free, enabling them to carry shopping or hold the hand of an accompanying child or partner.

Figure 1: Tourists consulting Map and Guidebook

There are therefore requirements that any mobile devices be light and easy to place in pockets etc. Further, since collaboration is a key aspect of tourism, whether traveling in groups or just asking another person for directions or recommendations, another requirement is that more than one user can simultaneously view and interact with the display. Collaboration around current mobile digital devices is awkward because of both display size and required positioning for clear reading of the display.

For the above reasons, we chose to investigate the use of emerging technologies for interactive paper in mobile tourist environments and, particularly, to support visitors to the Edinburgh Festival Fringe. The Fringe is the world's largest public arts festival with more than 300 venues and 1700 shows per day. With so many events on offer, visitors often select events during their visit based on contextual factors such as location, time and the weather as well as ticket availability and reviews. Several sources of printed information are available for tourists, including an official Fringe brochure which lists all events ordered alphabetically under category, a Fringe map which shows all the venues and daily event listings published in newspapers. By using interactive paper for these documents, these information sources can be *augmented* with digital information and services rather than *replaced* with digital devices and media. Thus what we provide is an optional, value-added service and users can easily migrate from the pure printed documents to the cross-media version if and when desired.

There are numerous research projects and commercially available products related to interactive paper and while they tend to vary widely in terms of their goals and technologies, most are based on some means of detecting user actions on paper documents and linking them to actions in the digital world. Products include LeapPad [11], an educational toy which supports learning through interactive books that play sound files in response to the child pointing to areas within a book, and PaperClick [18], a system for linking to web pages from paper documents. A number of products focus on writing capture rather than interaction and these include PC NoteTaker [19] and the Nokia Digital Pen based on Anoto technologies [3]. Regardless of the purpose, the underlying technology used to detect user actions on paper is based on either the physical environment of the paper or the paper itself. For example, in the case of LeapPad, the book is fixed on a special tablet and it is the relative position

of the "magic pen" over this tablet that determines the position pointed to on a page. In the DigitalDesk research project at Xerox EuroPARC [24], a video camera mounted above an office desk is used to detect user actions on documents and is therefore another example where the physical environment of the paper document rather than the document itself is used to track user actions.

If paper documents are to be interactive independent of their physical position or attachment to a special device, then the mechanism for detecting user actions on paper must rely on some sort of encoding of link or position information on the paper itself. One approach is to print some sort of link identifier on the paper. For example, barcodes or special icons could be used to encode unique identifiers that are mapped to digital media files or web pages to be displayed, see for example WebStickers [12]. Another approach is to encode position information across pages which can be used to track pen position. The advantage of this approach is that it can be used to capture writing as well as for interaction. Further, since links are based on relative position within a document rather than simple identifiers, it is possible to find all the elements within a document that link to a specific target. Anoto is a company that developed a technology to track the movement of a pen on paper. A special digital pen has a camera situated alongside the writing stylus to capture images of an almost invisible pattern of infrared absorbing dots printed on paper, as indicated in Figure 2.

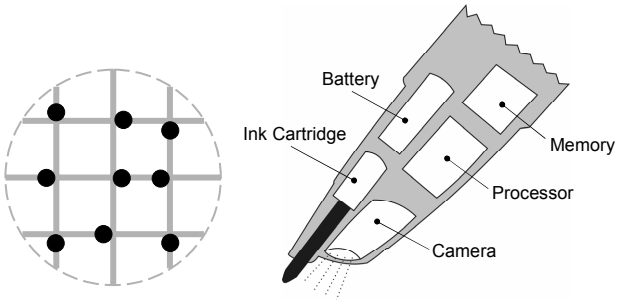

Figure 2: Anoto technologies

The pattern of dots encodes x and y positions in a vast virtual document space. Camera images are recorded and processed in real-time giving up to 100 x and y positions per second. The technology was developed for the digital capture of handwriting and several pages of handwriting can be captured and stored within the pen before being transmitted to a PC. In the case of the Nokia Digital Pen which is based on this technology, data transmission occurs either when the pen is placed in a cradle or via Bluetooth when a special send operation is activated. While data transmission on demand is sufficient for writing capture, immediate transfer of position data is required if the digital pen is to be used as an interaction device. Using special prototype pens from Anoto, we are able to switch the pen to streaming mode and use it for both real-time interaction and writing capture. We note that while these pens are not yet available in the marketplace, pens with similar functionality are expected to appear as products in the near future.

While technologies to track the movement of a pen on paper provide the basis for realising interactive paper documents, a server is required that can interpret the position information and invoke the appropriate digital services. Most projects tend to have developed simple servers dedicated to a specific application or set of applications. Our goal was to instead develop a general server framework that could be used to support any type of application and support both writing capture and interaction. The key to achieving this was to develop a general cross-media link server, called iServer, based on an advanced link metamodel that also includes user access modelling. Links are stored as database entities separately from the source and target resources. This means that all link properties, including the sources and targets that they reference, can be changed dynamically without requiring changes to the resources that are linked. Also a link can have multiple sources and targets if desired and these can easily be updated either by the users or automatically by the system using event triggers. For example, a time trigger could be used to change the information linked to a festival event after its last performance.

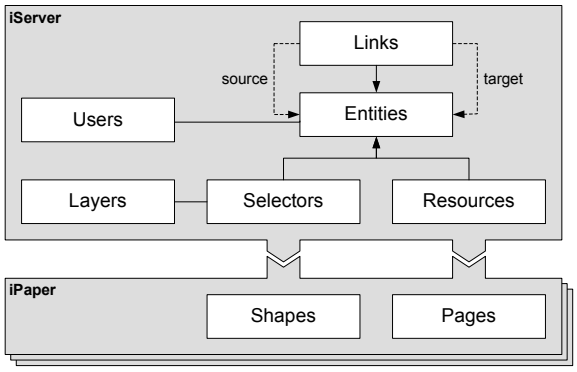

Figure 3: iServer architecture

To ensure full flexibility, a plug-in mechanism is used to support specific media types and links can be between resources, or elements within resources, of any media type as indicated in Figure 3. An element within a resource is specified using a selector mechanism for that media type. For example, in the plug-in for interactive paper, we use geometrical shapes within pages as selectors to define active areas within paper documents. In the case of an XHTML plug-in, we use XPointer expressions [25] to select elements, while for a video plug-in we can use time sequences as selectors. To create links to dynamic application information stored in a database, a plug-in can be used where the resource is a database and a selector is a database query.

The link metamodel is a generalisation of link models found in hypermedia systems [4], including that of the XML standard, XLink [25]. This means that we can support concepts such as multi-headed and multi-source links, links over links and also link annotations. In addition, we introduced a layer concept to support nested links. Layers can be activated, deactivated and re-ordered dynamically enabling layers to be used to provide context-dependent access. For example, layers may be used to provide zooming capabilities or alternative system functionalities. The former has been used in a natural history application to zoom in on digital versions of printed images by repeatedly selecting the image on paper. The latter occurs in an interactive map application where different layers provide different categories of information, such as cultural, entertainment or shopping, about

a city. Details of iServer, including the link metamodel, can be found in [21].

Technologies for interactive paper open up many possibilities for new forms of cross-media publishing. Many of the applications that we have so far developed have involved linking together independent paper and digital resources, or developing customised applications. An authoring tool has been developed to support the creation of links between existing paper documents (either in printed format or PDF versions of them) and digital media files, web pages or even other paper documents. Currently, a more general tool is under development to support the authoring of links between elements of all supported media types and also to various applications. However, while the manual authoring of links between existing resources is satisfactory for some applications, it is too time-consuming and tedious for others where there is a large volume of highly-linked cross-media content. In such cases, a content publishing approach where documents are generated from content stored in a database is more appropriate. Furthermore, since such systems are widely adopted nowadays for web publishing on multiple devices, it enables interactive paper to be integrated into a framework that supports multi-channel access to information.

Consider for example the case of introducing interactive paper as an information resource for tourists at the Edinburgh festivals. The idea is that event brochures printed on paper can be augmented with digital services to obtain information such as the events starting nearby in the next half hour, ticket availability or ratings of events by other tourists. To support mobility, we decided that additional, dynamic information supplied by these services should be delivered through a voice output channel based on a text-to-speech engine. Therefore the interaction devices that a user requires are the paper documents, a digital pen and an earpiece. In addition, they require some form of computing device, ideally with GPS. We have been experimenting with various mobile computing devices and also using laptops to carry out some user studies in fixed locations. We have also been using separate GPS devices. However, with the current rate of developments in smart phones and also wearable computers such as the QBIC [2], which is integrated into a belt, it is envisaged that we can start experimentation with true mobile solutions in the near future. Alternatively, there is the possibility of the computing device being integrated into the digital pen as is the case in the FLY Pentop computer from LeapFrog [**?**] due for release in the second half of 2005. The FLY Pentop computer is based on Anoto technologies, which we are also using for most of our demonstrator applications. Given the rapid development of new technologies for interactive paper and mobile computing, the important thing is that our software systems and publishing framework are independent of specific technology solutions and, for the moment, we focus on the information systems aspects and the modes and devices for interaction.

The Fringe Festival has thousands of events of various categories such as theatre, comedy and dance. Event information is currently published as a brochure, daily event lists published in newspapers and also on the web. The basic content is stored in a database of events and exported in various formats to support these publishing channels. Our goal is to extend this approach to content publishing to include support for interactive paper documents, which requires, not only the generation of the PDF to be printed, but also the automatic generation of link definitions for importation in iServer and the coupling of iServer and the content publishing system to support interaction. In the next section, we describe the main features of the web publishing framework on which this work was based and then, in later sections, go on to explain how it was extended for interactive paper.

3. OMSWE PUBLISHING FRAMEWORK

Nowadays a web publishing framework has to support access through, not only desktop browsers, but also a range of mobile devices such as PDAs and mobile phones. Multi-channel access may involve adaptation of, not only the format of information delivered, but also its content, structure and presentation.

Although a large variety of tools and technologies are available to support the publishing of static and dynamic data on the web, many of these lack a well-defined declarative model. Within the research community, this problem has been recognised and a number of model-based approaches have been proposed that exploit knowledge and experience gained over many years in the database and hypertext fields. In contrast to a number of existing model-based approaches to web site development such as WebML [8], our focus is system-based rather than tool-based. By this, we mean that instead of developing tools on top of existing database technologies, we wanted to investigate how new concepts could be integrated into the core of database systems to support web engineering and content publishing. Further, in addition to providing support for the operation of dynamic web sites, we wanted to support the development process itself in terms of rapid prototyping, system evolution, reusability and both static and dynamic web site generation. All of these requirements are coupled with those of multi-channel and context-dependent access mentioned before.

OMSwe [17] is the resulting web publishing framework that we developed as an extension of the OMS Pro object-oriented database management system. OMS Pro handles all data and metadata uniformly and both the architecture and system were metamodel-driven. As a result, the system not only offers very powerful and flexible data management facilities, but also by revising or extending the metamodel, it is relatively easy to integrate new concepts into the system and maintain orthogonality.

To achieve maximum flexibility, all of the information that defines a web site is managed in a publishing database. This means not only the application data such as festival events and their categories, but also the definition of document structures and their presentations. Note that this does not mean that all information, inclusive of multimedia materials, actually has to be stored in the database. It may be the case that application data is stored in one or more external databases and multimedia stored in external files. However, the publishing database manages references to all external data and also includes mechanisms to support the synchronisation of these sources with published information. In the case of external files, this involves the automatic copying of updated files to the relevant web server directories at predetermined or manually invoked synchronisation points.

The key concepts introduced into the database system to support web engineering were the notion of a reusable web component, called a `webElement`, and also a versioning

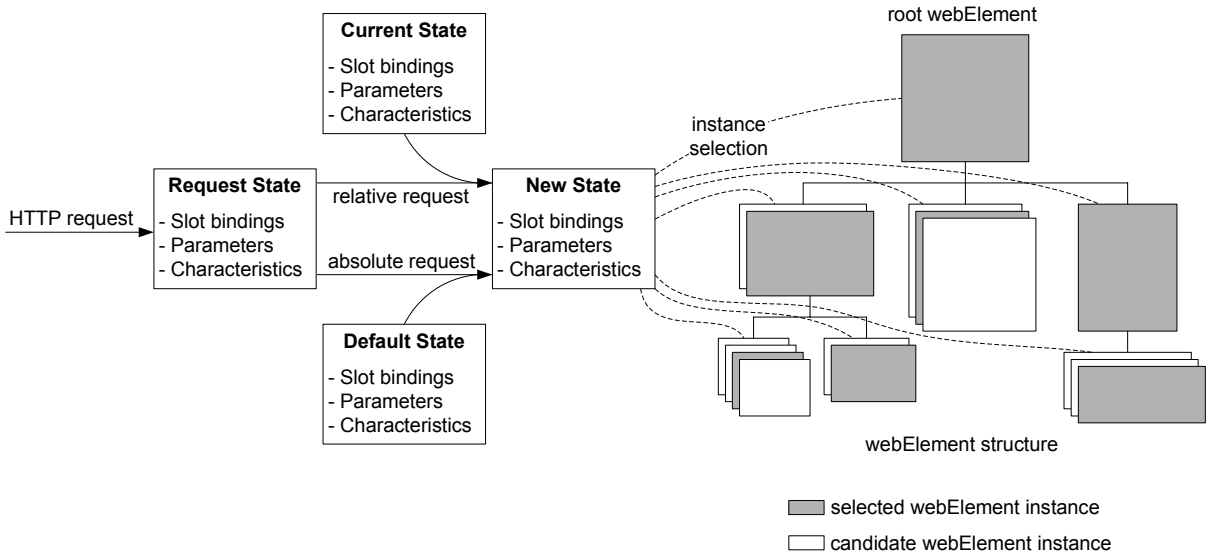

Figure 4: OMSwe request processing

mechanism for handling context-dependent delivery of information.

A web page is structured in terms of logical components such as the header, menu bar and main content. Components can be nested arbitrarily and a web page will often have a deeply nested structure. Components are defined in terms of content and presentation. In OMSwe, these components are represented as objects of type `webElement`. These objects define the dynamic information contained within a component and this can be a combination of information retrieved or generated from objects of the database and references to subcomponents. In this way, the content is defined in terms of a content hierarchy where each node is bound to a database operation that generates data. In generating the response to an incoming request for a web page, the dynamic content is evaluated through recursive calls according to the component structure. At each stage, the content generated is represented in XML and stored in an internal XML repository.

The presentation of a component is defined as an XSLT template and this is linked to the corresponding `webElement` through an association construct of the OM model. It is at this level that the developer specifies whether XHTML, WML, VoiceXML or some other markup should be generated. Just as the recursive evaluation of the dynamic content of a web page assembles all data as XML in an internal repository, it also assembles the XSLT templates in a separate repository. At the end of processing, the system then calls an XSLT transformer to generate the web document to be returned as a response to the request. We note that while this dynamic composition of web pages is very flexible, clearly it comes at a price in terms of the time to process a request. For that reason, caching plays an important role in the system and occurs at all levels of the system. Thus, it is not only possible to cache entire web pages, but also components of web pages and even the dynamic content resulting from queries to application data.

To handle context-dependent access to data, a notion of state had to be integrated into the database system to represent both interaction and context states. Context-dependent information delivery is then controlled by matching the context state of a request to context-specific versions of the objects involved.

The general scheme of processing incoming requests is illustrated in Figure 4. Data is extracted from incoming HTTP requests to form the request state. From the form of the request, the system detects whether or not it is relative to the current context. If it is relative, then the new state is formed by an update of the current state according to the request state. Otherwise, the current state information is lost and the new state is formed by a merge of the default state and the request state. Starting at the root `webElement`, the system then recursively generates the XML content and XSLT templates for the requested web document as described previously. However, at each stage of the processing in the content hierarchy, the system selects the `webElement` instance that best matches the new context state. Since a `webElement` specifies its dynamic content in terms of any subcomponents, this means that not only the content, but also the structure may adapt according to context. Similarly, the presentation may also be context-dependent based on versions of the associated template objects.

The model of context that we use is similar to that proposed by others for HTML [23] and also semi-structured data [22]. Context is defined in terms of a set of *characteristics* which specify the various factors to be taken into account in deciding on both the content and presentation of a document. These characteristics can include anything from user-related factors such as language preference and location to system-related factors such as the request protocol and client device. Thus multi-channel access is supported through a general context mechanism.

To support this in OMSwe, a system type `context_obj` was introduced to define version descriptors and objects inherited from this type are eligible for version operations. Our model and system support both multiple inheritance and schema evolution and this enables objects to be made

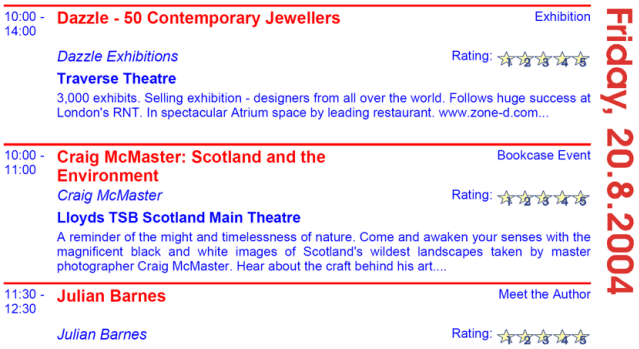

Figure 5: Part of brochure page

versionable without effecting the application-specific type hierarchy. Thus a non-context-aware application can easily be evolved into a context-aware application.

A `context_obj` object has 3 attributes that define when it is appropriate to use the object. The first specifies the set of characteristic-value pairs that define contexts in which this version is appropriate. The second and third attributes together define a valid time period for the version. This allows time-dependent components of a document to be specified. For example, a conference web site may have a paper submission component that is valid for only a specified period of time. By having two versions of this component for the time period before and after the final submission date, respectively, the system will automatically close the submission by selecting the "submission closed" component once the initial period has expired.

It is beyond the scope of this paper to describe our context model and mechanisms in detail, but it is important to note that, given a request, versions are selected according to a best match rather than exact match. This avoids having to specify versions for each possible context state. Characteristics such as language can be specified as ordered lists of preferences and defaults are defined for cases where no match is found. On the other hand, for cases where the characteristic value is critical, such as choosing the appropriate template for a particular client device, then the characteristic value can be specified as mandatory. Details of the context model and versioning mechanism are given in [15].

4. PUBLISHING PROCESS

In this section, we describe the changes that had to be made to the OMSwe system to support an application, such as the one for the Edinburgh festival, where information is published and accessed through interactive paper as well as possibly through normal web channels such as desktop browsers.

It is important to note that essentially there are two stages to be supported. First, there is the actual process of publishing a paper document. Second, there is the interaction from paper documents where the output is published over a separate channel such as a visual display or audio. We deal with each of these in turn describing the publishing process in this section and the interaction process in the next section.

In Figure 5, we show part of the interactive brochure that we developed for the Edinburgh festival as part of the EdFest project. The interactive digital services are intended to be an optional add-on service and therefore the brochure is designed to look much the same as a normal printed brochure with list of events. Users working with the digital pen and earpiece can obtain additional audio information about events, their location and also average ratings by pointing to appropriate areas on the page. In addition, they can add their own rating on an event, write comments about an event or set reminders so they are notified about the start of an event shortly before it begins. The comments about an event can either be written in a separate Anoto notebook or alongside the event listing. The capture of comments requires that the system recognises a switch in the use of the pen from interaction mode to writing mode. This is based on a combination of factors such as the area where the writing takes place and also timing factors. Comments are sent to the server and can be shared with other users. Presently, these can only be accessed through visual displays but in a future version we will use a combination of optical character recognition (OCR) technology and text to speech synthesis to enable audio access.

Clearly, current web publishing systems are designed to handle the publishing of such information on the web and this could be done quickly and easily using a system such as OMSwe. In addition to publishing festival event information for desktop browsers, we have also developed interfaces for mobile phones (WML and CHTML) and also voice based on VoiceXML. The question was how we could adapt the framework so we could integrate support for interactive paper. Each event has a fixed structure in terms of title, venue, category, description, ratings etc. and this enables the system to automatically generate the definition of active areas as geometrical shapes on specific pages. In other words, the content publishing system has to calculate the x and y coordinates of the boundaries of printed elements on a page. As with any standard web publishing system, it also determines the link information in terms of the URL of the corresponding link targets. Active areas may overlap. For example, the writing capture area for an event is the entire box where that event information is stored. The system manages this by defining the areas on different layers which is a concept not supported in most web publishing systems.

OMSwe already had the basic mechanisms to support the publishing of content as a printed document. It only required that XSL:FO templates be used to generate a PDF document which could then be printed. Since we require a certain precision over the physical placement of information on a page, we did encounter some problems with current XSL:FO processors. After some experimentation, we elected to use the Apache FOP engine [10]. At the moment, the printing of the document has to be performed in two stages — first printing the Anoto pattern and then the PDF document. This is a problem of restricted access to Anoto publishing software, but it is something we are working on and soon we will be able to print documents in a single step. This is particularly important for a second stage project concerning the Edinburgh festival where users will be able to print personalised interactive brochures on demand at public kiosks.

Alongside the generation of the PDF documents, we also need an automated export of the link information for the

iServer component in order that it can process future requests from that document. We therefore had to implement a special output channel in the content publishing framework that generates an XML description of the link information for the paper document as shown in Figure 6.

```
<?xml version="1.0" encoding="UTF-8"?>
<iserver>
<individual id="we">
  <name>OMS WebElements</name>
  <description>Publishing Framework</description>
  <login>omswe</login>
</individual>
<document id="festivalBooklet" creator="we">
  <name>Festival Booklet</name>
  <id>Edfest Booklet</id>
  <size>
   <width>148</width><height>210</height>
  </size>
  <content>http://edfest.org/Edfest/booklet.pdf</content>
</document>
<page id="p1" creator="we" document="festivalBooklet">
  <name>Page 1, Festival Booklet</name>
  <number>1</number>
</page>
<layer id="l1">
  <name>Link Layer</name>
</layer>
<rectangle id="r42" creator="we" layer="l1" resource="p1">
  <name>Performance Info Shape p-o21</name>
  <upperLeft>
   <point><x>24</x><y>29</y></point>
  </upperLeft>
  <size>
   <dimension>
    <width>103</width><height>8</height>
   </dimension>
  </size>
</rectangle>
<omswe id="po21" creator="we">
  <name>Performance Info p-o21</name>
  <description>Link to OMSwe</description>
  <content>
   http://edfest.org/oms?db_anchor=p_info&amp;db_p=p-o21
  </content>
</omswe>
<link id="lpo21" creator="we" sources="r42" targets="po21">
  <name>Link Performance Info p-o21</name>
</link>
</iserver>
```

Figure 6: XML link definition

The XML file uses a general format that can specify links of any form supported by the link metamodel. The XML document therefore specifies the document, pages, layers, geometrical shapes within pages and links separately. This allows links to be defined over arbitrary, and possible multiple, resources and resource selectors. Thus, it is possible to have a geometrical shape within a page be used as either a link anchor or a link target, or both. Additionally, links are associated with a creator and, in this case, that is the OMSwe system.

Once this XML file has been imported into the iServer, the link information is stored in its database. In the next section, we describe how requests generated by the activation of these links are handled by describing the EdFest architecture and system operation.

5. INTERACTION PROCESS

The overall architecture of the system that we developed for trials at the Edinburgh festival in 2004 is shown in Figure 7.

The mobile user is equipped with a copy of the interactive brochure, a digital pen, an earpiece with microphone and a GPS device, as well as a mobile computer. In addition, we also supported a voice-only interface and one that used a head-mounted display as a visual output channel. A desktop browser interface was also available for pre- or post-visit access to information.

The system is based on a client-server infrastructure with iServer and OMSwe as the two main server-side components. We note that for initial trials we wanted to keep the architecture relatively simple and therefore allowed the system to be used in one of two modes — either with a single remote server accessed via general internet connections or the user would have a local server as well as client. Clearly the former has major disadvantages in terms of performance and networking problems, while the latter restricts information sharing. However, this architecture was simple enough to test the basic concepts and requirements of interactive paper documents in mobile environments and in a second stage we are now developing a distributed architecture.

On the client-side the system consists of several components that offer special functionalities. The pen client is responsible for communication with the digital pen, while the voice engine is responsible for processing the VoiceXML files included in the response from the server and also supports speech interaction with the system. An HTML browser was also included on the client side for experimentation with head-mounted displays.

The digital pen works in streaming mode and sends x and y coordinates to the iServer component via the pen client in real-time. In the case of interaction, the user points to various areas within the document and it is only when it makes contact with the paper that the data is sent. When the user enters a capture area and starts to write, the pen remains on paper and a stream of x and y coordinates (together with pressure information) is sent to the server. iServer calls an active component that handles writing capture and the authored content is stored as an XML document. Rendering software can then be used to generate images of the captured content or, as stated previously, OCR software can be used to convert it to text form. With the Anoto pens currently available in the market place, rendering software is provided for the captured content. However, since we are using a prototype pen in streaming mode and sometimes want to render data in real-time, we had to develop our own rendering software.

When an active area in the brochure is pointed to, a request is sent to the iServer component including the pen's x and y coordinates. The iServer resolves the activated link to the appropriate target which is specified as a URL. For example, an active area within the booklet could be mapped to a URL `http://edfest.org/oms?db_anchor=getRating`. The iServer acts as a proxy for OMSwe, which receives

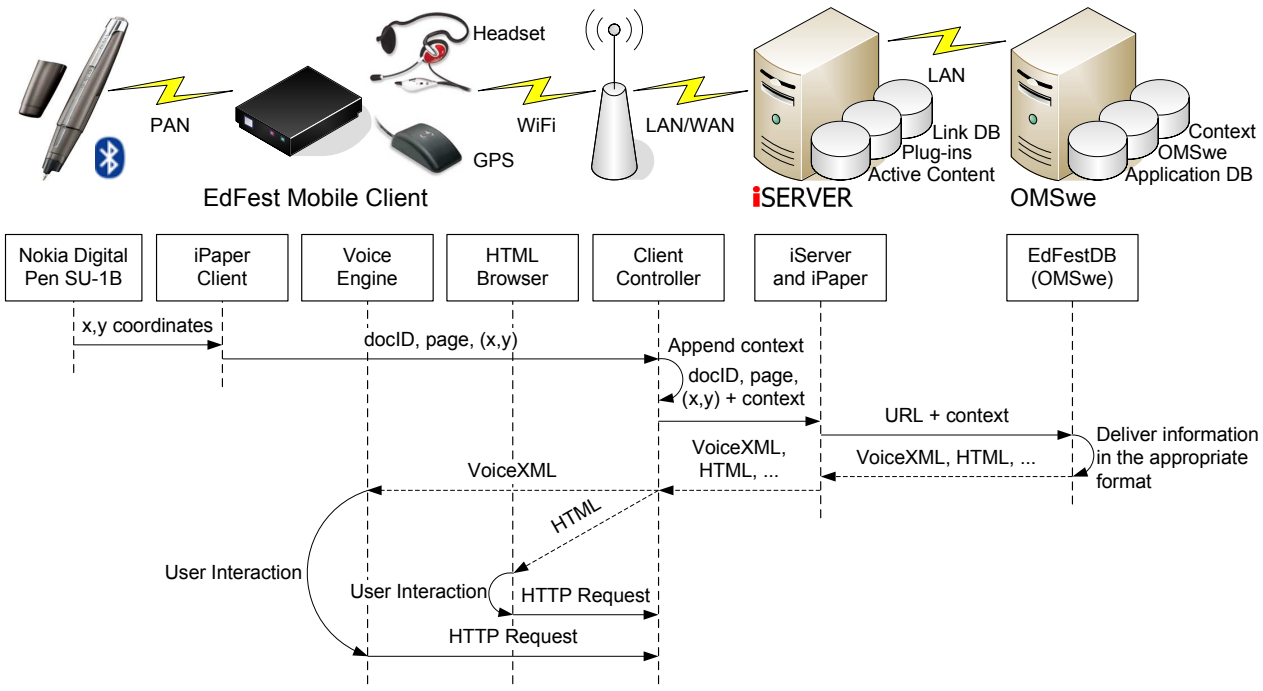

Figure 7: EdFest system architecture

the URL and returns the appropriate VoiceXML document. This document is generated dynamically from content stored in the application database.

OMSwe sends its responses to the client controller which is then responsible for dispatching it to the appropriate visualisation component. In the case of a VoiceXML document, the client controller forwards the information to the voice engine which provides audio output to the user.

The client of the EdFest system is composed of several interface components including the pen client and voice engine. Other components can be included to support other devices such as head-mounted displays. These are all either off-the-shelf components or customised components developed with other application setups in mind. By default, the components are autonomous and do not know anything about each other. In the case of the EdFest platform, we wanted to combine the components to form a multi-modal interface that could provide more functionality than the sum of the parts. The client controller is the component that takes care of this integration and is therefore the central component on the client side. It acts as an HTTP proxy for the pen client and the voice engine. Instead of communicating directly with the server components, the HTTP connection goes through the client controller which is able to alter or even replace both the HTTP request from the interface component and the response from the server, and also to trigger side-effects where required. The client controller also integrates additional components such as the GPS sensor.

One of the most important tasks of the client controller is the dispatching of HTTP responses. In contrast to most web systems, our multiple channels are not independent of each other. We need to be able to trigger a request using one modality e.g. the pen and display the response in another modality e.g. voice. The client controller analyses the MIME type of the responses and based on this information dispatches the result to the appropriate component.

Another issue in multi-modal interfaces is the consistency and synchronisation of the various input and output channels. For example, it should be possible to interrupt an audio message if the user selects another area on paper. Therefore the client controller also analyses requests and decides on actions to be taken such as, for example, sending a message to the voice engine to stop the current voice dialogue.

Summarising, one can see that the responsibility for the multi-modal aspects of the interface and the synchronisation between channels lies with the client controller rather than the web publishing component OMSwe. The problem is that current web publishing systems are designed to handle multi-channel access, but they still adopt a basic single request/single response model. This means that they do not support any form of synchronisation between channels, nor do they typically support multiple responses. For example, in the case of EdFest, one might want to return control information to the pen to activate its vibration mode or a display to a public or mobile display screen as well as audio information. We are therefore now investigating how we can best extend OMSwe to take over some of these responsibilities from the client controller. As a first step, when a multiple response is required, OMSwe generates an XML document that contains the various responses as subdocuments. The client controller can then dispatch the different responses to the relevant client devices.

6. EXPERIENCES AND FUTURE WORK

Initial tests and trials of the EdFest system took place in Edinburgh during August 2004. In the weeks before the trials, the design of the interactive event brochure was being carried out in parallel to the development of the application

system and extensions to the software infrastructure to cope with the requirements of context-awareness and multi-modal interaction. Since interactive paper is a new medium for access to complex information systems, the design of the brochure was far from trivial and went through many iterations and often quite radical changes. Initial paper prototypes based on mock-ups were used to gain user feedback on designs, but still the effect of actually testing the system functionality through interactive paper documents proved to be extremely important as we found that user reactions to the real system were often quite different from the reactions to mock-ups. We were keen to have a high-level of interactivity, meaning that we wanted to provide lots of functionality and augment almost all elements within the brochure with additional information or services as described in Section 4. This of course meant that there were a large number of active areas and links to be authored. Although the initial version of the brochure had only 10 pages, the ability to generate the corresponding PDF files and link definitions from a content publishing system was essential in enabling us to experiment with and refine the designs. In a second version of the system developed for 2005, we are working with real data provided by the Festival Fringe organisers. With 341 venues, 1890 performers and 27046 performances, it is clear that the authoring process is something that needs to be supported by a publishing framework.

Based on our experiences last year, we have also made some basic changes to the design of the documents and interface for the trials to be carried out in 2005. This is in addition to changing the basic set-up of the system in line with a 3-year project plan to develop and test different aspects of the system. The main focus in 2005 is to work with multiple interactive documents rather than having all functionality integrated in a single booklet. We are also introducing the concept of kiosks where users can access additional visual information and also print out personal daily event lists as interactive paper documents.

Usability trials of the initial version of the system were carried out during a three-day period during August 2004 and involved a mix of semi-structured interviews, video-based observations and user questionnaires. It is beyond the scope of this paper to describe these studies in detail, but we mention some issues raised that caused us to significantly change our design this year. As stated earlier, the basic idea is that the brochure can be used on its own and therefore the layout and information content should be similar to that of the normal Fringe brochure. We therefore chose to simply have activate areas of text so that by pointing, for example, to the name of a venue, audio information would be given about how to get to the venue. This caused some problems in the trials because users were unsure where to point and what the associated information and functionality would be. Further, many users were reluctant to point to the texts with the pen in case they marked the document. Unfortunately, since we are using a digital pen for both interaction and writing capture, and the pen does not have a retractable stylus, we cannot avoid this. In response to both of these issues, we have decided this year to carry out trials with an icon-based document design, where only icons are active and they indicate the associated functionality. Again we emphasise that the main contribution of the publishing framework presented here is that we are able to quickly and easily generate alternative designs and therefore to carry out studies comparing different styles of interfaces in realistic settings.

With respect to the publishing framework, there are a number of issues that we want to investigate with regard to supporting the production of cross-media information materials. First of all, it is important that the actual authors of content, editors and layout designers can work with familiar tools that support their specific tasks. We are therefore planning to provide export/import facilities for publishing tools such as QuarkXPress. Secondly, the OMSwe system has evolved over the years from a general database management system to a system with integrated support for web publishing. Based on our experiences with OMSwe, we are developing a new system, XCM, that extends the functionality of OMSwe to increase support for context-aware systems. This is achieved by augmenting the concepts of content, structure and presentation with an orthogonal context dimension [5].

7. CONCLUSIONS

We have presented a content publishing framework that supports the creation and use of interactive paper documents. By integrating support for interactive paper in a general web publishing framework, we are able to publish rich content sources on different channels, including interactive paper. Although, in theory, the creation of interactive documents can be easily incorporated in existing content publishing frameworks, we have shown that some care must be taken as the system must have control and knowledge of the physical positioning of printed elements on document pages. Further, it needs to automatically generate and export link information as well as the document to be printed.

The creation of the interactive paper document is only part of the process, as digital information may need to be generated dynamically when the user interacts with the document. Since current technologies for interactive paper provide only an input channel, responses need to be displayed on other output channels such as audio or visual displays. To show the issues involved and how they can be addressed, we have presented the system architecture for a mobile tourist information system that was based on the coupling of a cross-media server for interactive paper with a web publishing framework. The experiments that we carried out with this system have validated the general approach and also the merits of interactive paper in mobile environments. They have also helped us refine the requirements of both the system and the design of interactive documents in terms of both functionality and layout. Based on these we are now developing a second version of the system for the Edinburgh festival in 2005.

8. ACKNOWLEDGMENTS

We thank the present and former members of our research group who have contributed to the EdFest project. They are Rudi Belotti, Corsin Decurtins, Marco Dubacher, Michael Grossniklaus, Ljiljana Vukelja and Nadir Weibel.

9. REFERENCES

[1] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: A Mobile Context-Aware Tour Guide. *Wireless Networks*, 3:421–433, 1997.

[2] O. Amft, M. Lauffer, S. Ossevoort, F. Macaluso, P. Lukowicz, and G. Tröster. Design of the QBIC Wearable Computing Platform. In *Proceedings of ASAP 2004, 15th IEEE International Conference on Application-specific Systems, Architectures and Processors*, Galveston, Texas, 2004.

[3] Anoto AB, http://www.anoto.com.

[4] H. Ashman and R. M. Simpson. Computing Surveys' Electronic Symposium on Hypertext and Hypermedia: Editorial. *ACM Computing Surveys*, 31(4):325–334, December 1999.

[5] R. Belotti, C. Decurtins, M. Grossniklaus, M. Norrie, and A. Palinginis. Interplay of Content and Context. *Journal of Web Engineering*, 4(1), 2005.

[6] B. Brown and M. Chalmers. Tourism and Mobile Technology. In *Proceedings of ECSCW 2003, 8th European Conference on Computer Supported Cooperative Work*, Helsinki, Finland, September 2003.

[7] B. Brown, M. Chalmers, M. Bell, I. MacColl, M. Hall, and P. Rudman. Sharing the Square: Collaborative Leisure in the City Streets. In *Proceedings of ECSCW'05, 9th European Conference on Computer-Supported Cooperative Work*, Paris, France, 2005.

[8] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, and M. Matera. *Designing Data-Intensive Web Applications*. Morgan Kaufmann, 2003.

[9] K. Cheverst, N. Davies, K. Mitchell, A. Friday, and C. Efstratiou. Developing a Context-Aware Electronic Tourist Guide: Some Issues and Experiences. In *Proceedings of ACM CHI 2000, Conference on Human Factors in Computing Systems*, The Hague, 2000.

[10] Apache FOP, http://xml.apache.org/fop/.

[11] LeapPad Learning System, LeapFrog Enterprises, Inc., http://www.leapfrog.com.

[12] P. Ljungstrand, J. Redström, and L. E. Holmquist. WebStickers: Using Physical Tokens to Access, Manage and Share Bookmarks to the Web. In *Proceedings of DARE'2000, Designing Augmented Reality Environments*, Elsinore, Denmark, April 2000.

[13] C. C. Marshall. Toward an Ecology of Hypertext Annotation. In *Proceedings of ACM Hypertext'98, 9th Conference on Hypertext and Hypermedia*, Pittsburgh, USA, June 1998.

[14] M. Norrie. Paper on the Move. In *Proceedings of Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS 2004)*, Springer LNCS 3272, 2004.

[15] M. Norrie and A. Palinginis. Versions for Context Dependent Information Services. In *Proceedings of COOPIS 2003, Conference on Cooperative Information Systems*, Catania-Sicily, Italy, 2003.

[16] K. O'Hara and A. Sellen. A Comparison of Reading Paper and On-Line Documents. In *Proceedings of ACM CHI'97, Conference on Human Factors in Computing Systems*, March 1997.

[17] OMS Web Elements (OMSwe), http://www.globis.ethz.ch/research/oms/platforms.

[18] NeoMedia Technologies, http://www.paperclick.com.

[19] PC NoteTaker, http://www.pc-notetaker.com.

[20] A. J. Sellen and R. Harper. *The Myth of the Paperless Office*. MIT Press, November 2001.

[21] B. Signer and M. C. Norrie. A Framework for Cross-media Information Management. In *Proceedings of EuroIMSA 2005, Grindelwald, Switzerland*, February 2005.

[22] Y. Stavrakas and M. Gergatsoulis. Multidimensional Semistructured Data: Representing Context-Dependent Information on the Web. In *Proceedings of CAiSE'02, 14th International Conference on Advanced Information Systems Engineering*, Toronto, Canada, June 2002.

[23] W. Wadge, G. Brown, M. Schraefel, and T. Yildrim. Intensional HTML. In *Proceedings of 4th International Workshop on Principles of Digital Document Processing*, March 1998.

[24] P. Wellner. Interacting with Paper on the DigitalDesk. *Communications of the ACM*, 36(7), July 1993.

[25] E. Wilde and D. Lowe. *XPath, Xlink, XPointer and XML: A Practical Guide to Web Hyperlinking and Transclusion*. Addison Wesley, 2002.

[26] A. Woodruff, P. Aoki, A. Hurst, and M. Szymanski. Electronic Guidebooks and Visitor Attentio. In *Proceedings of ICHIM, 6th International Cultural Heritage Informatics Meeting*, Milan, Italy, 2001.