Diss. ETH No. 16218

Fundamental Concepts for Interactive Paper and Cross-Media Information Spaces

A dissertation submitted to the
SWISS FEDERAL INSTITUTE OF TECHNOLOGY
ZURICH

for the degree of
Doctor of Technical Sciences

presented by

Beat Signer

Dipl. Informatik-Ing. ETH
born August 20, 1973
citizen of Appenzell, AI, Switzerland

accepted on the recommendation of

Prof. Dr. M. C. Norrie, examiner
Prof. Dr. C. Heath, co-examiner

2005

To my parents

Abstract

While there have been dramatic increases in the use of digital technologies for information storage, processing and delivery over the last twenty years, the affordances of paper have ensured its retention as a key information medium. Despite predictions of the paperless office, paper is ever more present in our daily work as reflected by the continuously increasing worldwide paper consumption.

Many researchers have argued for the retention of paper as an information resource and its integration into cross-media environments as opposed to its replacement. This has resulted in a wide variety of projects and technological developments for digitally augmented paper documents over the past decade. However, the majority of the realised projects focus on technical advances in terms of hardware but pay less attention to the very fundamental information integration and cross-media information management issues.

Our information-centric approach for a tight integration of paper and digital information is based on extending an object-oriented database management system with functionality for cross-media information management. The resulting iServer platform introduces fundamental link concepts at an abstract level. The iServer's core link management functionality is available across different multimedia resources. Only the media-specific portion of these general concepts, for example the specification of a link's source anchor, has to be implemented in the form of a plug-in to support new resource types. This resource plug-in mechanism results in a flexible and extensible system where new types of digital as well as physical resources can easily be integrated and, more importantly, cross-linked to the growing set of supported multimedia resources. In addition to the associative linking of information, our solution allows for the integration of semantic metadata and supports multiple classification of information units. iServer can, not only link between various static information entities, but also link to active content and this has proven to be very effective in enabling more complex interaction design.

As part of the European project Paper++, under the Disappearing Computer Programme, an iServer plug-in for interactive paper has been implemented to *fully integrate paper and digital media*, thereby gaining the best of the physical and the digital worlds. It not only supports linking from physical paper to digital information, but also enables links from digital content to physical paper or even paper to paper links. This multi-mode user interface results in highly interactive systems where users can easily switch back and forth between paper and digital information. The definition of an abstract input device interface further provides flexibility for supporting emerging technologies for paper link definition in addition to the hardware solutions for paper link definition and activation that were developed within the Paper++ project.

We introduce different approaches for cross-media information authoring where information is either compiled by established publishers with an expertise in a specific domain or by individuals who produce their own cross-media information environments. Preauthored information can be combined with personally aggregated information. A distributed peer-to-peer version of the iServer platform supports collaborative authoring and the sharing of link knowledge within a community of users.

The associations between different types of resources as well as other application-specific information can be visualised on different output channels. Universal access to the iServer's information space is granted using the eXtensible Information Management Architecture (XIMA), our publishing platform for multi-channel access.

Our fundamental concepts for interactive paper and cross-media information management have been designed independently of particular hardware solutions and modes of interaction which enables the iServer platform to easily adapt to both new technologies and applications. Finally, the information infrastructure that we have developed has great potential as an experimental platform for the investigation of emerging multimedia resources in general and interactive paper with its possible applications in particular.

Zusammenfassung

Obwohl die Verwendung von digitalen Technologien zur Speicherung, Verarbeitung und Verteilung von Information in den letzten 20 Jahren stark zugenommen hat, konnte sich Papier dank seiner speziellen Eigenschaften als wichtiges Informationsmedium behaupten. Entgegen allen Vorhersagen des papierlosen Büros ist Papier bei der täglichen Arbeit präsenter denn je, was sich auch im kontinuierlich steigenden weltweiten Papierkonsum widerspiegelt.

Anstatt Papier durch digitale Medien zu ersetzen, haben sich viele Wissenschaftler für die Erhaltung von Papier als Informationsresource und eine bessere Integration von Papier und digitalen Informationssystemen engagiert. Dies führte in den letzten zehn Jahren zu einer Vielzahl von Forschungsprojekten und technologischen Entwicklungen, die sich mit dem Thema des interaktiven Papiers befassen. Die Mehrzahl der Projekte fokusiert jedoch auf technische Fortschritte im Bereich der Hardware und vernachlässigt die grundlegenden Probleme der Informationsintegration sowie die der medienübergreifenden Informationsverwaltung.

Unser informationsbasierter Ansatz für eine enge Integration von Papier und digitaler Information beruht auf einer Erweiterung eines objektorientierten Datenbanksystems mit Funktionalität für eine medienübergreifende Informationsverwaltung. Die von uns realisierte iServer Informationsplattform führt fundamentale Konzepte für eine medienübergreifende Linkverwaltung zwischen beliebigen Medien ein. Um neue Medientypen in das System zu integrieren, muss jeweils nur der medienspezifische Teil dieser Konzepte, wie zum Beispiel die konkrete Verankerung eines Links innerhalb eines bestimmten Mediums, implementiert werden, da die iServer Kernfunktionalität für alle multimedialen Ressourcen benutzt werden kann. Dieser ressourcenbasierte Plug-in Mechanismus führt zu einem flexiblen und erweiterbaren System, in welches neue digitale sowie physikalische Ressourcen einfach integriert und mit bereits unterstützten Medien verlinkt werden können. Zusätzlich zur assoziativen

Informationsverknüpfung unterstützt unsere Lösung die Integration semantischer Metadaten und die Mehrfachklassifikation von Informationseinheiten.

Als Teil des Europäischen Forschungsprojektes Paper++, im Rahmen des Disappearing Computer Programmes, haben wir ein iServer Plug-in für interaktives Papier entwickelt, um *Papier und digitale Medien zu verbinden* und damit die Vorteile der physikalischen und digitalen Welt zu vereinen. Wir unterstützen nicht nur das Verknüpfen von Papier und digitaler Information, sondern auch Verbindungen von digitaler Information zu Papier und selbst direkte Veknüpfungen zwischen Papierdokumenten. Die dabei verwendete multimodale Benutzerschnittstelle führt zu einem hochgradig interaktiven System, welches es dem Benutzer sehr einfach erlaubt, zwischen Papier und digitaler Information hin und her zu wechseln. Eine erweiterbare Eingabeschnittstelle ermöglicht es zusätzlich zu der von unseren Projektpartnern erarbeiteten Lösung neue Hardware Technolgien, welche sich zur Definition eines papierbasierten Links eignen, zu unterstützen.

Wir diskutieren verschiedene Ansätzte zur Entwicklung von Applikationen welche mehrere unterschiedliche Informationsmedien, wie zum Beispiel gedruckte Information, Video und Ton, umfassen. Die Information kann dabei entweder durch einen Verleger mit Expertise in einem spezifischen Fachgebiet verfasst werden, oder die Benutzer können ihre eigenen medienübergreifenden Informationsräume gestalten. Die vom Verleger verfasste Information kann jederzeit mit persönlich erstellter Information kombiniert werden. Eine verteilte Peer-to-Peer Version der iServer Platform ermöglicht zudem das kollaborative Erstellen und Austauschen von Information innerhalb einer Gruppe von Benutzern. Unsere erweiterbare Architektur für Informationsmanagement (XIMA), eine Publishing-Komponente, garantiert zudem universellen Zugriff auf die von iServer verwalteten Daten.

Unsere grundlegenden Konzepte für interaktives Papier und eine medienübergeifende Informationsverwaltung wurden unabhängig von spezifischen Hardwarelösungen und Interaktionsmodellen entwickelt. Dies ermöglicht eine einfache Adaption der iServer Platform, um neue Technologien und Applikationen zu unterstützen. Zu guter Letzt weist die von uns entwickelte Informationsinfrastruktur ein grosses Potential auf, um als experimentelle Platform für die Evaluation neu aufkommender Technologien für interaktives Papier sowie anderer Medien eingesetzt zu werden.

Acknowledgements

I would like to thank all those who either directly or indirectly contributed to the outcome of this thesis.

First of all I would like to thank Prof. Moira C. Norrie for her unlimited support and interest in my work. She provided me with the necessary freedom to realise my vision and, at the same time, always guided me in the right direction. Many of the results presented have been critically analysed and refined in very inspiring and fruitful discussions with her in which she also introduced me to the secrets and beauty of metamodelling.

A special thank you goes to my co-supervisor Prof. Christian Heath who opened my mind to many new ideas and interests. He showed me how the smallest detail recorded in field observations can have a tremendous impact on the requirements for technical problem solving.

I would also like to thank all the present and former members of the GlobIS group for their support and collegiality. It was always very stimulating to feel the great team spirit in many projects that we have realised together. I look back with pleasure to many of the adventures experienced in the "GlobIS family"!

Part of the research that has been undertaken in this thesis was embedded in the multinational European Research project Paper++ with participants from multiple disciplines. I therefore would like to thank all the other members of the Paper++ project including Guy Adams, Wolfgang Bock, Adam Drazin, David Frohlich, Peter Herdman, Sandy Johnstone, Paul Luff, Rachel Murphy, Nadja Linketscher, Abigail Sellen, Ella Tallyn and Emil Zeller.

The artists Axel Vogelsang and Curt Walter Tannhäuser further experimented with the possibilities of our cross-media platform and its application for interactive installations. I would like to thank them for challenging me in providing a variety of new requirements thereby pushing the development of our architecture. I would also like to thank my good friend Oscar Chinellato for many interesting discussions over lunch.

Further, I would like to thank him and my colleague Michael Grossniklaus for helping me to improve the quality of this thesis by proofreading the manuscript.

Last but not least I would like to thank my family who always supported me and showed interest in my work.

Table of Contents

There is only one thing that makes a dream impossible to achieve: the fear of failure.

Paulo Coelho

1 Introduction

> *Over the next decade, most corporate offices will still be geared to the movement of information on pieces of paper. Office automation will bring improvements in productivity through the use of automatic typewriters and other stand-alone equipment that crank out paper faster. Some interoffice information will move electronically, but what the manager reads and files will be printed on paper.*
>
> *But during this period, a relatively small but fast-growing group of companies will have moved into the office-of-the-future environment. The leap forward will be led by the "papermakers" — those companies that are involved primarily in generating, modifying, or moving paper. These pioneers will have hooked together word-processing equipment into office systems to transfer information electronically and to move it into and out of central electronic files. For them, it will be the start of the paperless office.*
>
> Business Week, 1975 [135]

These two paragraphs were published in 1975 as part of a Business Week article in which the future of the *paperless office* was predicted for the very first time. Some thirty years later, we are still far away from a realisation of that vision where paper becomes completely replaced by

digital technologies. The annual worldwide paper consumption is continuously increasing and has more than doubled since 1975. Paper not only refused to go away but is still prevalent in most offices. Even in research institutions with access to the latest hardware and information management technologies, paper still represents one of the main information sources. Figure 1.1 shows an example of an academic's office of the early 21st century where paper continues to play an important role in knowledge working. It is a fact that nowadays information is available on different types of digital and physical media. It is no longer sufficient that knowledge workers, people developing or using knowledge, have a tool for organising their information for a single type of media but we need possibilities to support the seamless transition across different types of physical and digital information sources. The integration of information available in a variety of formats results in *cross-media information spaces*, where users can associate information across physical and digital resources.

Figure 1.1: Example of an academic's office

Advocates of a paperless future commonly argued that there were just a few technical challenges to be solved and it would only be a matter of time until paper would become completely replaced by digital artefacts. The image quality of the computer screens on which documents had to be read was one such problem in the early 80s. Reading a document on

a computer screen used to be much more fatiguing to human eyes than reading the same document on printed paper. In addition, the reading of a digital document was restricted to specific locations since the heavy desktop computers could not be carried around. However, more recently these hardware problems have been solved and small portable electronic reading devices, called e-books, with a reasonable image quality are now predicted as the successor of paper documents. With this, the unfulfilled vision of the paperless office has even been extended to the vision of the *paperless home* where all our books and daily newspapers will be replaced by a single e-book which can hold multiple documents and dynamically download new content on demand over a wireless network connection.

To understand why even these most recent advances in technology will not lead to a paperless world — at least not in the near future — we have to revisit the long history of paper and the evolution of some of its properties over the last two thousand years. The portability and readability of paper have already been introduced as two of its important features but there are other affordances of paper that are equally important. For instance, paper is not only a simple carrier of information but often is also used as a medium for a set of collaborative activities. Various ethnographic workplace studies have analysed the usage of paper documents and revealed further properties of paper in terms of its embedded usage in work processes. The observations made in these studies influence the design of new systems and tools aiming to support and improve some of these paper-based interactions. In the next section we present the developments in the history of paper and highlight some properties of paper which are difficult to mimic in a digital world.

1.1 Affordances of Paper

Paper was invented in China in 105 A.D. It was only in the twelfth century that Arab traders brought it over to Europe. At that time mainly animal skins were used as writing surfaces. The cheaper but more fragile paper medium did not immediately replace animal skins but it took another three hundred years before paper became the major writing material. In 1440 Johannes Gutenberg developed the printing press and the demand for paper increased enormously since it was much better suited to printing than skins. Cotton or linen cloth waste, also known as rags, were used as the source material in the papermaking process. The invention of the printing process led to a shortage of rags in the sixteenth

century and people started looking for other materials as a substitute. However, it took almost another three hundred years before wood pulp became commercially used for paper production in the middle of the nineteenth century.

The history of paper covers nearly two thousand years during which paper has seen many major technical improvements to finally become today's highly optimised writing material. We can also see that adaptation from one medium to another, for example from animal skin to paper, did not take place immediately, but rather was a lengthy process. Often it takes a considerable amount of time before people get used to a new kind of medium. Further, a true evolutionary progress in terms of functionality of the new type of medium should manifest itself, which means that the new medium should not simply try to imitate the properties of its predecessor.

While there have been dramatic increases in the use of digital technologies for the storage, processing and delivery of information over the last two decades, the affordances of paper have ensured its retention as a key information medium. Paper has many advantages over digital media in terms of how people can work with it, both individually and in groups. It is portable, light, cheap, flexible and robust. The physical properties of paper support many forms of human actions such as grasping, folding, marking on etc. Furthermore, various forms of paper-based collaboration and interaction are nearly impossible to support in digital environments [98].

Sellen and Harper point out four reading-related key affordances of paper in the *The Myth of the Paperless Office* [156]. First, paper allows for quick and flexible navigation through a document with the size of a document being a rough indicator for the amount of information stored in it and the readers always knowing where they are while flicking through the pages. A second affordance of paper for reading is the possibility of marking up a document while reading. Free-form annotations help readers to mark important text passages for easier re-reading and to structure their thoughts. Further, Sellen and Harper mention an affordance which is based on the fact that the information on paper is fixed but still the paper documents remain mobile. It is possible to read across more than one document at the same time by placing multiple documents next to each other and thereby defining a spatial order on a work space. Finally, paper supports the seamless interweaving of hybrid activities such as reading and writing. For instance, by placing a document next to a notebook we can take notes while reading the document.

As an example of how we use different types of paper documents in combination with digital writing tools let us have a look at the process of writing this thesis. While writing this thesis various affordances of paper mentioned earlier were used. In addition to the portable computer that represents the primary writing tool, there are plenty of paper documents covering the author's desktop as shown in Figure 1.2.

Figure 1.2: The author's writing space

Behind the laptop there are four large folders of research papers about related work that have been read and annotated by the author over the past three years. On the right hand side there are three large piles of books classified by topics: books about information systems, paper, hypermedia and some philosophical and historical books. However, these piles are more than just a classification of different books. The order of the books within the different piles together with other contextual information represents the active process of thinking involved in writing the thesis. The paper notebooks that have been used for taking notes during the last three years and also some new research articles which needed to be read, so-called active or "hot" articles, lie in front of the book piles. Last, but not least, to the left hand side of the laptop there are two English dictionaries and some manuals for the LaTeX typesetting system that has been used to write the thesis. While most of these documents would also be available in a digital form it is important to note

that the task of writing the thesis would become much harder without the use of paper documents. The spatial layout of documents on the desk allows the author to easily switch from one document to another or use multiple documents next to each other. Further, all the open books and note pages together define the current state and context of the work. This awareness of spatial context is just one of the features that are very difficult to emulate in a digital world.

While typesetting systems on a computer provide great support in writing a manuscript, desks are often covered with paper documents that are consulted while writing articles. As mentioned earlier, it is no longer the case that the resolution on computer screens is not suited to reading documents digitally but other factors, such as the quick navigation among different documents and the flexibility of spatial layout, are critical in the writing process [134]. In addition, knowledge workers often cannot immediately classify a piece of information [81]. They use the physical space, for example their desks, as a temporary spatial state of their ideas and other information to be processed and classified. After a piece of information has been analysed and annotated by the knowledge worker, it may not be important that the handwritten marks are digitised in a way that computer tools can "understand" their meaning.

The marking up and annotation of content is supported in many different forms by paper. We can use a highlighter pen to mark important sections while reading a paper or use a pen to write comments in the margins of a document. There is a smooth transition from completely informal to more formal annotations. Free text or sketches are just two examples of simple informal annotations. The introduction of annotation guidelines leads to more formal annotations which are, for example, used in typographic markup. It is not easy to support the same richness of annotations in digital applications [109, 110].

Various workplace studies have been carried out to investigate the role of paper in working environments equipped with modern technology. For example, the use of paper flight strips in Air Traffic Control rooms was analysed in an ethnographic study by Mackay et al. [103]. A result of their studies is that paper flight strips support safe and effective work practices and that they offer a flexibility hard to achieve with digital systems [104].

In addition to these more evident features, paper provides other less apparent or "hidden" properties. Many processes in working environments involve paper as a medium for collaborative activities without having been specially designed for these tasks. Quite often new digital

systems which should support our work activities miss some of these properties of paper for collaborative activities and therefore are less accepted by a specific target community. The detailed analysis of working environments may help in designing effective computer supported cooperative work (CSCW) environments paying attention to exactly these "hidden" properties. Heath and Luff undertook several field studies in different organisational environments [67, 101]. They found out that paper and screen-based media provide rather distinctive support for cooperation and that the use of paper persists not only due to its intrinsic properties or constraints of screens but also because paper affords interactional flexibility based on its mobility. For example, they describe how physicians benefit from the mobility of paper-based medical records. An outcome of their observations is that paper affords interactional flexibility for asynchronous as well as synchronous cooperation that allows individuals a range of ways of participating in tasks-in-interaction.

On the other hand, digital technologies have many advantages in terms of storing and accessing large amounts of information, displaying multimedia and fast full-text searching. Digital information is less static than printed information and can be dynamically updated with ease. Brown and Duguid emphasise the complementary character of paper and digital information when they talk about the digital office in *The Social Life of Information* [26].

> *Time has only confirmed this early indication of paper's importance in the digital office. While other print technologies have come to compete with it, laser printer sales have increased twelve-fold in the past decade. If the digital office from PARC to the present is anything to go by, bits and atoms, the digital and the material, don't seem so much in opposition as in tandem. Despite confident claims that their only relationship is one of replacement and dismissal, the two look much more like complementary resources.*
>
> Brown and Duguid, 2002

The dream of the paperless office has failed — at least until this day. However, we cannot foresee if the move from paper to digital material is something that will never entirely happen or a rather complex task that will take a long time to be fulfilled; maybe another three hundred years such as the change from skin to wood pulp for paper production. If paper cannot be replaced by digital media, we can still try to carry on

its evolutionary process by integrating it more closely into digital information environments. We aim for a solution where the highly optimised properties of paper media which have evolved over the last two thousand years are augmented and enhanced with new digital features by tightly integrating paper and digital technologies, thereby gaining the best of the physical and the digital world.

1.2 Integrating Paper and Digital Media

The digital era has already influenced and changed the traditional publishing industry. There has been a major shift in the use of paper. More and more publishers not only produce, but also distribute, their documents electronically. While electronic documents are easier to distribute and update, most users still prefer to read and annotate documents on paper. Some of the traditional publisher's printing process has therefore been outsourced to the consumers.

As highlighted earlier, in the near future paper documents will not be replaced by digital information. However, we should at least try to reduce the gap between printed and digital information. If we find a way to address parts of paper documents, we can build new forms of *interactive paper*, where printed documents are augmented with supplementary digital information or services.

Over the last decade, there has been a major rethinking of how computers could help us in managing information and supporting our daily work. In the early nineties, Mark Weiser coined the term *Ubiquitous Computing*. Instead of trying to digitise our physical world and using computers mainly as storage components, small "computers" could be embedded in everyday objects to make them more powerful in terms of connectivity and information exchange with surrounding objects and the environment. In his article *The Computer for the 21st Century* [181], published in the Scientific American, Weiser describes a scenario of how intelligent paper could be integrated into future working environments.

> *At breakfast Sal reads the news. She still prefers the paper form, as do most people. She spots an interesting quote from a columnist in the business section. She wipes her pen over the newspaper's name, date, section, and page number and then circles the quote. The pen sends a message to the paper, which transmits the quote to her office.*
>
> Weiser, 1991

It is not only the case that physical artefacts may somehow be connected to, or enhanced by, digital information but they may also be used as input and output devices. The resulting tangible user interfaces (TUIs) are a new form of interacting with computers in a more natural way by electronically coupling physical and digital objects. The user no longer has to work with a digital representation of a physical concept. They can work directly with the physical object that has either some embedded computing power or is connected to a server performing the requested operations. Durrell Bishop's Marble Answering Machine [76] represents an early project in the area of tangible user interfaces. Incoming telephone messages are "physically instantiated" as marbles by the answering machine. The user can get access to the messages associated with single marbles by grasping a marble and dropping it into the designated slot on the answering machine. A benefit of tangible and ubiquitous computing in terms of work activities is that it can help to reduce the interruption that results from switching from physical to digital information and vice versa. The idea of interactive paper fits well in the scenario of ubiquitous and tangible computing where paper becomes a physical interface to interact with digital information.

The issue of architectures and technologies for the integration of existing data sources has also been a topic of interest for many years within the database community. A number of solutions have been proposed with the approach taken dependent on the degree of heterogeneity, the level of local autonomy and, of course, the required functionality of the target system. In terms of heterogeneity, many forms of data sources have been covered including different categories of database management systems, knowledge base systems, file systems and web documents. But one major form of information source has so far been neglected, namely paper documents.

The integration of paper and digital media involves two main steps. First, a method to *link paper* to digital actions is required. A lot of effort has been put into the development of new technologies to encode information on paper and capture information from physical paper. We will present many of these technical solutions when discussing related work in Chapter 2.

In a second step, the information acquired by a specific hardware solution is used to access the appropriate digital information or service. This implies the availability of a software infrastructure for the actual *link and information management*. Too often, existing solutions focus on the development of the hardware and tend to neglect the underlying

information infrastructure. In contrast, we turned our attention to the development of a powerful and flexible cross-media information management architecture. The focus on basic cross-media information management concepts naturally leads to generic and powerful concepts for interactive paper as well as the integration of other multimedia resources. Different hardware solutions can then be integrated into the general cross-media information management architecture. This data-centric approach for interactive paper facilitates adaptation to evolving technologies and reduces the risk of a dead end by choosing the wrong hardware solution.

The question remains as to the potential uses and models of interaction afforded by interactive paper. How can we fully exploit the potential of paper as a client device and integrate it seamlessly into semantically rich digital information spaces? While a number of research projects have proposed different variations of interactive paper and presented their visions for its use, none have really addressed the issue of making paper a *first-class medium* in the context of an interactive information system. This means that it should be possible to link, not only from paper to digital resources, but also from digital resources to paper and even from paper to paper.

The hypertext visionary Vannevar Bush discussed the problem of information overload already sixty years ago in his article *As We May Think* [28] published in the Atlantic Monthly. In this article, Bush described a mechanical device, the Memex, in which knowledge workers could store their books, records and communications in microfilm and retrieve them later with appropriate speed and flexibility. The important thing about the Memex was that information could, not only be stored, but also associative links could be defined between different pieces of information. He described the job of a trailblazer who was responsible for linking pieces of related information together. However, Bush never proposed using the Memex primarily as a writing tool. His initial idea was that people would still write linear texts but could later link them to specific parts of related texts as part of a hypertext authoring process. The Memex was intended to support the definition of edges in a graph where the nodes were built from existing text documents, but not to support the actual authoring of a document's content. The knowledge worker could build their personal associative information space which would support them in later knowledge retrieval. In addition to the definition of links, represented by edges between existing nodes, hypertext writing systems such as Eastgate Systems' *Storyspace* [169] provide an integrated solution for authoring textual content as well as connections (links) between

text fragments. Nevertheless, in hypertext literature there is a clear distinction between the writer of a hypertext and its reader. The hypertext narrative writer clearly defines the paths that can be taken between different fragments of text and the reader can only choose among a set of preauthored links.

Today's information infrastructures often restrict the authoring process to specific users. For example, if we take the traditional Web, only a person who has write access to the original HTML document can change the web page and add new content and links to other digital resources. We aim at a more open authoring process where each reader may also become an author and therefore dynamically change the hypertext structure. Such a collaborative authoring process can always be combined with information provided by a content publisher who is an expert in their specific domain. Of course, an open and *democratic authoring* process introduces new problems concerning the quality of information which may be corrupted by users publishing incorrect information. However, in our opinion, the positive effects of collaborative authoring processes in a strong community outweigh the negative consequences of potentially false information published by a few malicious individuals. A positive example of such a new form of an open web publishing platform is the free Wikipedia encyclopaedia [187], a fast-growing knowledge resource, where everybody can add new, or edit existing, information.

Our approach for interactive paper and more general cross-media information spaces is somehow similar to Vannevar Bush's idea of the Memex which should help in organising information by aggregating meta information in the form of links about related information entities. We do not want to replace paper and accept that the linear writing of original texts is preferable in many situations. However, our architecture for interactive paper and cross-media information management should help to organise and manage information more efficiently by annotating linear paper documents with more dynamic digital media or linking them to specific parts of related paper documents. Our goal was to develop a general platform for cross-media information management, called *iServer*, providing *fundamental concepts* for linking different information entities. These link concepts are general enough to support the full range of existing hypermedia solutions including, for example, spatial hypermedia, temporal hypermedia and physical hypermedia. Application developers can use the iServer platform to manage information across different types of media and write their own extensions in the form of resource plug-ins to support new types of media.

1.3 Contribution of this Thesis

Over the last twenty years we have seen a dramatic increase in the use of digital technologies for information storage, processing and delivery. However, despite technical advances in digital information management, paper is still the key information medium for many work activities. It is only recently that a new research area dealing with the integration of paper and digital information or services has formed. However, most of the interactive paper projects that have been realised so far, focus on specific hardware solutions for linking paper documents and digital information thereby often neglecting the information management part. This leads to isolated solutions which work only for very specific application domains and clearly defined hardware setup.

In this thesis, we show that by approaching the problem of implementing interactive paper from an information point of view, we can take the integration of the physical and digital information spaces one step further and realise highly-connected information environments that can link together elements of arbitrary resources and enable users to *move freely back and forth between physical and digital information sources.* As will be shown in this thesis, the potential functionality and flexibility of a cross-media information management system is determined by the underlying information model and architecture.

We identify fundamental concepts for cross-media information spaces, including interactive paper documents, and propose a general link model for cross-media information management bringing together many existing hypermedia concepts. Future extensibility is granted by a resource plug-in mechanism ensuring that new digital as well as physical resources can be easily integrated and, more importantly, cross-linked to the growing set of supported multimedia resources. Furthermore, we have integrated a layer and a user management component into the very core of the link model to manage link access rights and therefore enable applications to control the visibility of links in both a user- and context-dependent manner. Associative inter-application and cross-media linking as supported by the model enables effective information management across isolated application domains.

The general cross-media link model has been implemented by extending an object-oriented database management system with the required functionality. The resulting iServer platform introduces fundamental link concepts at an abstract level. Only a small media-specific part has to be implemented to support new resource types whereas the iServer's core

link management functionality is available across different multimedia resources. In addition to the associative linking of information, our solution allows for the integration of semantic metadata and the multiple classification of information units. Further, iServer can, not only link between various static information entities, but also link to active components, which are small pieces of program logic executed on link activation. The active component paradigm has proven to be very effective in enabling more complex interaction design.

A modern information infrastructure has to be able to cater for all forms of mobile devices, user preferences and context-dependent delivery. It is a matter of delivering the right information, to the right person, at the right time. The eXtensible Information Management Architecture (XIMA) is a web publishing framework ensuring that all the link metadata stored in the iServer framework can be delivered on different output channels. Based on a clear separation of content and presentation, it enables documents to be generated dynamically to suit client device, user and context.

An iServer plug-in for interactive paper (iPaper) has been developed to fully integrate paper and digital media. The interactive paper platform supports *enhanced reading*, where the active reading process [2] — a combination of reading with critical thinking and note taking — is supported with supplementary digital information. Furthermore, it enables the active, paper-based capture of new information, known as *enhanced writing*. Based on a set of abstract interfaces, the interactive paper platform is very flexible in supporting various forms of input devices.

To demonstrate the applicability of iServer and the iPaper plug-in to different application scenarios, various prototypes have been implemented interweaving paper and digital information. The flexibility of the iServer resource plug-in mechanism has been verified by implementing three additional resource plug-ins to support HTML content, movies and Radio Frequency Identification (RFID) tags as new specialised resource types.

The interactive paper authoring tool which is based on iServer authoring functionality allows active areas on paper to be defined and linked to digital or physical objects. *Link authoring* and *content authoring* are introduced as two possibilities for integrating paper and digital information and solutions for preauthored, personal and dynamic links are presented. Since the authoring of cross-media information spaces by a single publisher has proven to be a time-consuming and expensive process, we have investigated alternative authoring paradigms where much more power

is given to the user and each user becomes a potential author of new link knowledge. To support this collaborative form of link authoring and sharing, we have implemented a distributed version of the iServer platform based on decentralised peer-to-peer (P2P) technologies. In this collaborative authoring process, each user has their personal link information space which can be dynamically enriched with link information from other users. By applying such an adaptive community authoring process, each user can contribute information according to their domain of expertise and, on the other hand, profit by the knowledge of experts in other domains.

1.4 Paper++ Project

The iServer platform and its iPaper plug-in for interactive paper were developed within the European Paper++ research project (IST-2000-26130), which was part of the Disappearing Computer initiative. The goal of the Paper++ project was to integrate paper and digital information and thereby bridge the paper-digital divide.

Paper++ was a multinational project with collaboration from King's College London, HP Laboratories Bristol, ETH Zurich, Anitra Medienprojekte GmbH, and Arjo Wiggins. Our project partners worked on position encoding patterns and the corresponding hardware to read these patterns, new printing technologies, interaction design and user studies concerned with the general use of paper as well as new interactive paper applications. In this multidisciplinary team of professionals, it was the task of the author of this thesis to design and implement the necessary information management infrastructure for integrating paper and digital media.

As part of the Paper++ project, we implemented multiple demonstrator applications. These applications included an interactive nature encyclopaedia, an interactive city map of Zurich and an interactive worksheet for the Natural History Museum in London. Our project partners helped in the design of these applications and evaluated them in user trials to verify the feasibility of our approach. After the completion of the Paper++ project, we developed various other applications based on the interactive paper software platform. These applications are no longer based on the hardware prototypes that were developed within the Paper++ project but on commercially available digital pen and paper technology.

1.5 Structure of this Thesis

In Chapter 2 we discuss different approaches for the integration of paper and digital information and cross-media information management. We introduce the basic actions required for integrating paper and digital information such as document identification and the capture of handwritten information. Various technical solutions for solving these tasks are presented and advantages and disadvantages of specific technologies are discussed. Existing interactive paper projects using one or more of these technologies are presented and classified by the type of actions they support. We give a critical analysis of existing work in the area of interactive paper and present our hypothesis proposing a set of fundamental concepts for a general cross-media information management platform.

Based on the analysis of existing work in Chapter 2, we present a general model for cross-media information management in Chapter 3. The new cross-media information model introduces associations between different physical or digital resources at an abstract level without making any assumptions about a specific type of supported media. The information model builds the theoretical foundation for the cross-media information platform and the interactive paper framework presented in subsequent chapters. In addition to the new cross-media model, we introduce some basic terminology which is necessary to compare our approach to existing solutions.

Chapter 4 introduces the Java implementation of our cross-media information platform (iServer) and the corresponding Application Programming Interface (API) which is available to applications built on iServer. We show what is involved in integrating new media types into the iServer architecture using a resource plug-in mechanism and, in addition to the interactive paper extension, present three plug-ins for HTML documents, RFID tags, and QuickTime media types including movies, sound files and still images. In addition, the application of the iServer framework for cross-media annotations is discussed. We show that annotations are just a special type of links where the creation of a link often also involves active capture of information and point out that the iServer platform is well suited for this task.

In Chapter 5 we outline the requirements for tightly integrating paper and digital information and introduce the main components of our architecture for interactive paper. After presenting the client and server components, we describe the address space mapping component which is used to decode the positional information encoded in the paper documents.

The reader interface which makes it possible to integrate different kinds of input devices is presented before describing the OMS Java database system used for information storage. Finally, we present the XIMA framework which is applied for visualisation and supports universal access to the digital information managed by the interactive paper framework from a variety of client devices.

Various applications that have been developed to evaluate specific aspects of the interactive paper framework are presented in Chapter 6. This includes applications for enhanced reading as well as enhanced writing. Furthermore, the implementation of multiple applications helped in validating the flexibility of our cross-media information management platform to support a variety of paper-based interactions as well as different hardware settings.

Different kinds of cross-media authoring are introduced in Chapter 7. In addition to the screen-based generation of interactive paper documents, links can also be created directly on paper documents using writing capture. After presenting our authoring tool for link authoring we discuss the potential of semantic content authoring based on content management functionality. We then introduce collaborative link and content authoring based on a decentralised peer-to-peer version of the iServer platform.

We outline how issues of information semantics and granularity have an impact on the richness of the resulting interactive information environment. However, everything comes at a price, and we discuss the consequences for the content provider in terms of complexity and cost. These richer information environments could require a major shift away from the traditional practices of publishers towards the content management solutions that have already been adopted in the web publishing world.

Finally, in Chapter 8, we summarise the outcomes of this thesis and provide a critical analysis of the results achieved. Last but not least, we point out directions of future research in the area of interactive paper and cross-media information management.

A room without books is like a body without a soul.

Cicero

2 Background

In this chapter we give an overview of existing technologies dealing with the integration of paper and digital information and present various interactive paper research projects addressing different tasks and application domains. Before discussing related work, we provide some basic link terminology and outline minimal requirements for integrating paper and digital information.

Rather than presenting a historical review of related work, we classify the different projects according to their functionality and the working tasks they support. We start by discussing projects dealing with enhanced reading, where the focus is on reading digitally augmented paper documents. We then shift to projects where the writing process is enhanced by digital services. In addition to enhanced reading and writing, we present projects supporting annotation activities, which can be characterised in terms of the way that they combine reading and writing.

We then discuss projects where paper can not only be linked to additional digital information but also applied to trigger specific actions of a digital application workflow. In this case, paper becomes a tangible user interface for specific digital applications and may even replace traditional input devices such as the keyboard or the mouse. Finally, we analyse the presented projects, provide some concluding remarks about the flexibility and extensibility of existing interactive paper solutions and state our hypothesis for new forms of interactive paper based on a general cross-media link model.

2.1 Basic Terminology

Before we start presenting related work in the area of interactive paper and discussing how the integration of physical and digital information has been realised by different projects, we have to introduce some basic link terminology. A link L simply defines an association between a source entity S and a target entity T as shown in Figure 2.1. Each link L is associated with a source and a target entity, or more formally, $source(L) = S$ and $target(L) = T$. The arrow indicates that a link is always *directed*, pointing from the source S to the target T. By selecting and activating a link bound to a source S we get access to the link target T.

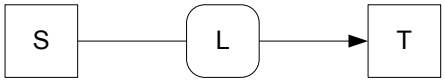


Figure 2.1: Link from source to target entity

In cross-media linking, the source S as well as the target T of a link can be of different media. This implies that there are basically the four types of links shown in Table 2.1 that can be defined: *digital-to-digital*, *digital-to-physical*, *physical-to-digital* and *physical-to-physical* links. The best known of these four link types is the digital-to-digital link, since we use it every time we browse the Web. The HTML links used on the Web are special in that the link and its source are stored in the same document. To give an example, let us have a look at the single HTML link `<a href="http://www.globis.ethz.ch">GlobIS</a>`. In this example, the link source S is defined as `GlobIS`, the text surrounded by the anchor tag `<a>`. The link target T is another web page whose address is given by the anchor tag's `href` attribute. Finally, the link L is represented by the anchor tag and its metadata and is stored in the same document as the link source S.

Most of the projects presented in this chapter implement some form of physical-to-digital links, where physical artefacts, in particular paper documents, become augmented with digital information. Physically augmented digital objects, i.e. links from digital information to physical objects, as well as physical-to-physical links are not supported by most of the interactive paper research projects. As we will show later, our general cross-media link model supports all four link types.

Some basic actions have to be provided by an interactive paper technology. First, we need a means of *identifying documents*. The different technologies for identifying paper documents and linking them to digital

Source S	**Target** T
digital	digital
digital	physical
physical	digital
physical	physical

Table 2.1: Cross-media link types

information differ in the flexibility and granularity that they support in defining link sources for a specific document.

A first solution to identify parts of a document is based on attaching *unique identifiers*, known as tags, to paper documents which are then associated with the corresponding digital or physical resources. However, links that have been defined in this way do not refer to the document structure. An alternative approach does not link directly from single unique tags, but defines *position dependant link anchors* based on specific elements of a document. In the case that a user has selected a position dependant link anchor, the associated link target will be processed. This position dependant link anchor approach is much more flexible than simple document tagging since arbitrary elements of a paper document can be used to trigger interactions rather than only specific parts of the document which have been tagged with unique identifiers. Note that in contrast to the simple mapping approach, a system based on position dependant link anchors has to apply some further processing on a user's input to check whether a link should be followed.

Furthermore, we can distinguish between technical solutions supporting the activation of predefined links for enhanced reading and approaches enabling the active, paper-based capture of new information, which we previously referred to as enhanced writing. In some enhanced reading systems, users can not only activate predefined links, but also add their own links to various kinds of resources, thereby supporting a form of authoring.

After the physical link sources have been defined, they have to be associated with the appropriate digital or physical information. Once again, there are various solutions supporting different forms of linking and levels of flexibility in terms of extensibility. Some projects apply simple resource mapping tables with two entries for each link: a unique identifier, which is normally encoded in a tag for the link source S, and

the link target T represented by a Uniform Resource Locator (URL) addressing a web resource. Such a mapping table approach is simple but has clear disadvantages regarding the quantity and granularity of links that can be defined in a document. Another class of projects therefore applies sophisticated link concepts which have been introduced by the hypertext community, for example spatial hypermedia infrastructures, to build more powerful and flexible systems in terms of link resolution.

Finally, in addition to the link meta information required to augment paper with additional information, an information management component models the digital as well as the physical information spaces. Some of the systems presented do not have an information management component at all and only support linking to static files or applications. Other projects implement application-specific databases and link the documents or parts of a page to information stored in these databases. This implies that we can not only skim paper documents, but also browse further digital media after following a paper-to-digital link.

2.2 Technologies

There exist various technologies for integrating paper and digital information. The most basic technology deals with *document identification*, enabling supplementary information to be provided to the user based on the document they are currently working on. However, if it should be possible to, not only associate entire physical documents with digital information, but augment various parts of a page with different information, we have to track a user's position within the document. The available *position tracking* technologies differ in terms of mobility and the resolution that they offer. While solutions with a low resolution may be used to address parts of a document, those with a high resolution can also be used for *writing capture*.

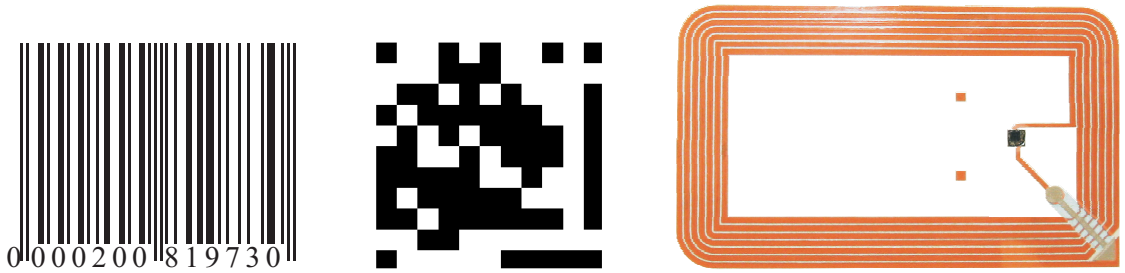
In addition to the position-based capturing of information, there exist other technologies for digitising printed or handwritten information from paper documents based on scanner hardware. Further, paper documents can not only be linked with digital information, but also provide a container for digital *information storage*, where digital data gets directly encoded within printed documents. However, all of these solutions require some devices in addition to the paper documents for accessing the supplementary digital information. The *electronic paper* technology tries to avoid this separation of paper and hardware components by

completely replacing paper with paper-like digital devices. The different technologies for document identification, position tracking, writing capture, information storage and electronic paper form the basis of all applications discussed later in Section 2.3 and we now examine each of these in turn.

2.2.1 Object Identification

In this first section about technologies for linking paper and digital information we focus on object or document identification. In general, the solutions for object identification are all based on the idea of attaching a machine-readable tag carrying a unique identifier to paper objects. The various approaches differ in terms of the tags and hardware devices for reading the identifiers stored on these tags.

A well-known technology for object identification makes use of the *linear barcode*, such as the one shown in Figure 2.2(a). Linear barcodes have been commercially used for more than 30 years and are well established in sales product identification in the form of the European Article Numbering (EAN) codes and Universal Product Codes (UPC). For example, the International Standard Book Number (ISBN) is a unique identifier that is encoded in EAN barcode format and applied to identify books and other forms of physical media. By printing unique barcode identifiers on paper documents and applying them as link sources, we can get easy access to digital information by decoding the unique object identifiers using off-the-shelf light-emitting diode (LED) or laser barcode readers.



(a) Barcode, EAN-13 (b) 2D barcode (c) RFID tag

Figure 2.2: Object identifiers

An advantage of using linear barcodes for document identification is the relatively low price for the barcode reader and, what is more important, the possibility to generate the barcodes almost free of cost with standard laser or inkjet printers. Furthermore, due to the long presence

in the market, linear barcode readers are very reliable. On the negative side, the visual barcodes may interfere with content on the printed document, especially if, not only entire documents, but also multiple areas within a page, should be tagged by barcodes.

Two-dimensional barcodes (2D barcodes), such as the one shown in Figure 2.2(b), are a special form of barcodes with information encoded in two dimensions. The 2D barcodes can no longer be read with a laser barcode reader but have to be captured with a camera device. However, low-cost camera devices which are suitable for reading 2D barcodes are nowadays integrated in most mobile phones. Two-dimensional barcodes allow for higher density of information encoding than linear barcodes. It is further possible to determine the three dimensional position of an object tagged with a 2D barcode based on the distortion of the camera image. However, the 2D barcodes interfere with the printed information in the same way as linear barcodes. Other forms of visual tags, for example small pictures with a thick black border, can be used as tracking marks by using vision-based tracking software such as the *ARToolKit* [9].

A newer technology for identifying objects is based on *Radio Frequency Identification* (RFID). Passive RFID tags [178, 179] are small objects encoding a Globally Unique Identifier (GUID) which can be read out inductively by a special reading device in combination with a wired antenna. An example of an RFID tag is shown in Figure 2.2(c). In comparison to linear or 2D barcodes, the RFID tags have the advantage that they can be applied without interfering with the printed artwork, since they can be read out even if they are hidden from sight. On the negative side, it is impossible to detect the exact position of a single RFID tag within the scanning range of an antenna. This implies that we cannot address specific regions of a page by using different RFID tags.

Other technologies used for object tracking based on an encoding of a unique identifier, in a similar way to RFID technology, are *Infrared Beacons* and *iButtons* [73].

2.2.2 Position Tracking

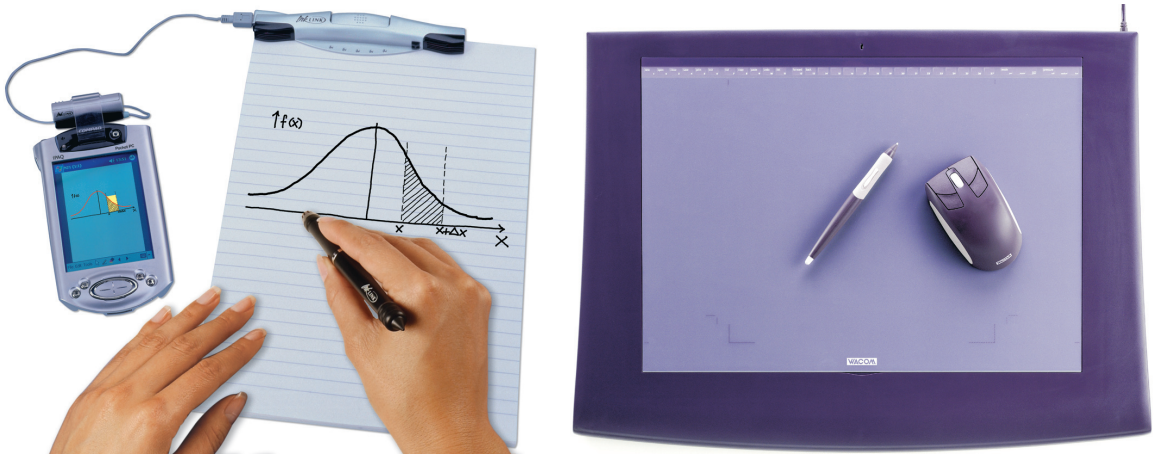
By tracking a user's activity within a document page it becomes possible to augment, not only whole documents, but also parts of document pages, with digital information. Existing technologies for position tracking can be classified in two dimensions. First, there are two classes of devices which track either *relative* or *absolute positions*. Then we can classify the devices based on their *support for mobility*. Some position tracking

solutions require fixed installations within a room and are not mobile at all while other technologies can be freely carried around with the paper documents on which they have to track the position. As stated earlier, the position tracking systems are applied to address specific parts of a document, but if the resolution is high enough, they can also provide handwriting capture functionality.

A relative position tracking device that most of us use in our daily work with computers is the mouse. The mouse is mainly used to select and click but sometimes, for example in presentation software such as PowerPoint, it is also used to capture writing. The position tracking mechanism of a mouse can easily be integrated in a writing pen which enables us to write comments on paper documents and at the same time capture all positions. This solution can be applied to track a pen's relative movements on a paper document but it is not possible to detect the pen's absolute position without the use of any additional calibration mechanism.

The *FieldMouse* [164], a special reading device based on mouse tracking technology, solves this problem by measuring a user's pointing position relative to a reference location within the paper document. To work on a specific document page, a user first has to calibrate the FieldMouse by scanning a barcode in the upper left page corner. By moving the FieldMouse over different objects on a page and clicking the mouse button, the associated web resources can be accessed. The system tracks the FieldMouse's position relative to the fixed barcode in the upper left corner and uses this positional information to access the digital information. Thus the FieldMouse combines the functionality of a barcode reader and a regular mouse. The barcode reader scans the unique identifiers associated with the printed artwork whereas the mouse records the FieldMouse's movement relative to the barcode position. Note that the concept of the FieldMouse can be generalised to any combination of a device that can read unique IDs such as a barcode reader or an RFID tag reader and a device that can measure relative motion, for example a mouse or an accelerometer.

Another form of position tracking system, the so-called *clip-on* solution, is based on the paper being attached to a special device with two or more reference points. These reference points continuously measure the distance to a special sound-emitting pen based on high-resolution ultrasonic position detection. Depending on the pen distance measured from the different reference points, the paper clip-on device can calculate the pen's position on the paper document.



(a) Seiko InkLink (b) Wacom graphics tablet

Figure 2.3: Position tracking devices

The clip-on solutions are well suited to writing capture. An advantage of this approach is that the position detection process is independent of the actual medium on which the pen is used. Therefore, the clip-on technology is a candidate to augment media other than paper, such as transparencies or whiteboards. However, it gets more complicated if we want to address specific parts of a document page using this technology, since the documents always have to be calibrated with the clip-on device. The clip-on technology is a single page and single document solution, which means that the position detection device has to be changed to work with different documents and page switches have to be initiated explicitly.

The *PC Notes Taker* [142] by Pegasus Technologies and the *Seiko InkLink* handwriting system [74] shown in Figure 2.3(a) (courtesy of Seiko Instruments) are two pen-based capture systems for documents up to A4 size which are based on the described clip-on technology. The *mimio Xi* [118] is a commercially available product for capturing handwritten notes from large whiteboards based on the same clip-on technology. The detection device is attached to a whiteboard, and special pens emitting an ultrasonic signal are used to capture information written on the whiteboard.

Another device for position tracking is the *graphics tablet*. A graphics tablet normally consists of a flat writing surface and a special pointing device in the form of a stylus as shown in Figure 2.3(b) (courtesy of Wacom Technology). A weak magnetic field is applied to the writing surface, thanks to which the position of the writing stylus can be detected. If we

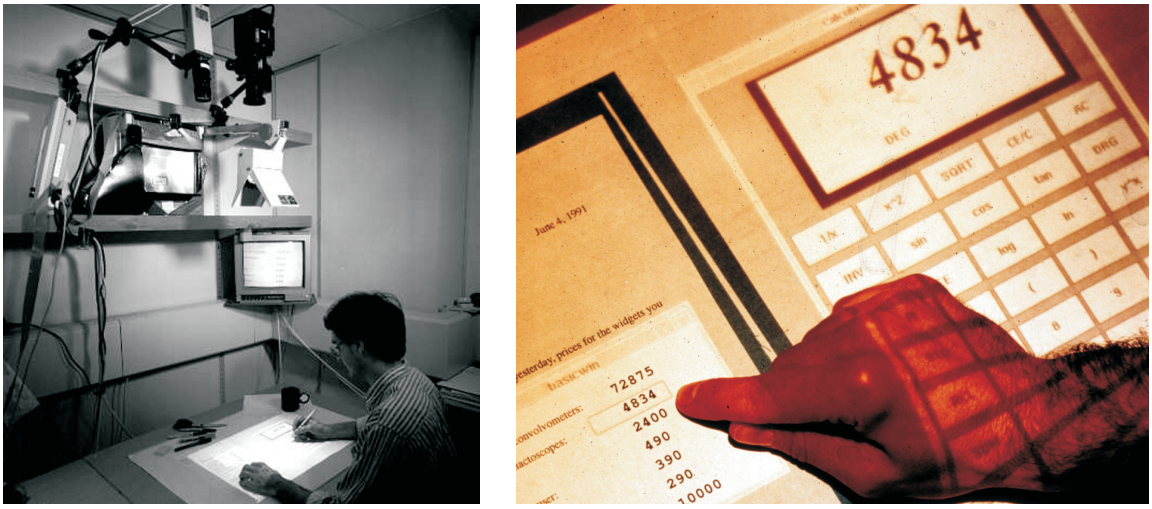
place a paper document on top of a graphics tablet, we get a position tracking solution for paper documents which is very similar to the clip-on technology, including the problem of missing page and document context.

For applications that require only rough positional information there exists another position detection technology based on low-frequency *electric field sensing* [166]. Instead of relying on a special pointing device, the electric field sensing solution measures the signal changes caused by a user's hand placed between a transmitter and the corresponding receiver hidden in the writing surface. By analysing these changes in signal strength, the system can roughly detect the current position of a user's hands.

A lot of daily office work is performed sitting in front of a desk writing documents on a desktop computer. In addition to the computer, many desks are covered with paper documents currently being processed by the knowledge worker. However, the paper documents and the digital information stored on the computer represent two isolated information environments and the worker always has to switch between physical and digital representations. *Augmented desk* systems try to close the gap between physical and digital information by bringing additional functionality to an ordinary physical desk based on camera-tracking technologies.

The first augmented desk system, the *DigitalDesk* [122, 183], was built in the early 90s by Pierre Wellner at Xerox EuroPARC. A camera mounted above the desk surface, so-called over-desk video, is used to track a user's interaction with documents lying on the desk as shown in Figure 2.4(a). Additional information can either be presented on a separate computer screen or, in a more advanced version of the DigitalDesk, a projector can be used to directly display information on the table in the form of a physical paper overlay. Figure 2.4(b) shows the *DigitalDesk Calculator* [182], a specific application implemented on top of the DigitalDesk architecture. A calculator is projected onto the desk surface and the user can work with it as with a regular electronic calculator. An over-desk camera tracks the user's finger and a number can be entered by touching the projected number buttons. However, there is an alternative way to enter the numbers. If the number to be entered is already printed on a piece of paper lying on the desk, the person sitting on the DigitalDesk can use it as input for the calculator by simply pointing at it with their finger. The camera then captures the printed number and applies optical character recognition (OCR).

Over the last ten years, the concept of the digitally augmented desk has been addressed by various research projects and different forms of



(a) DigitalDesk (b) DigitalDesk Calculator

Figure 2.4: DigitalDesk with calculator application

augmented desk prototypes have been designed. The DigitalDesk has been extended in the *CamWorks* project [172] which investigated different kinds of over-desk video based information capture ranging from greyscale image, black and white image to OCR. The *EnhancedDesk* [92] is an augmented desk system similar to the DigitalDesk with a strong focus on real-time hand tracking and gesture recognition. In contrast to other augmented desk solutions, the EnhancedDesk uses an infrared camera to track a user's hands on the desk surface. Note that the idea of using a desk surface for position tracking and, on the other hand, projecting digital information on the very same desk can be applied to any flat surface, including large wall-mounted boards.

A drawback of the augmented desk and augmented wall solutions is that we lose one of the main benefits that paper affords in terms of its mobility. To get access to supplementary functionality or information, the paper documents can be used only in special places which are equipped with a digital desk system. However, for augmenting single rooms such as offices or installations in museum exhibitions, the digital desk and wall approaches have great potential.

So far we have presented position tracking technologies which augment the paper documents with a coordinate space. We now discuss solutions that directly encode positional information on the paper documents. In the Paper++ project [138] a grid of almost invisible linear barcodes is printed on the paper documents with conductive ink. Figure 2.5(a) shows first prototype prints of such a barcode grid where the conductive ink is not yet totally invisible.

(a) Conductive barcode grid

(b) Inductive pen

Figure 2.5: Paper++ hardware

A specially developed *inductive pen* shown in Figure 2.5(b) detects information encoded in the barcodes by measuring the inductivity and decodes it to the corresponding (x,y) positions. Cost was a major consideration in the development of the conductive Paper++ pen technology as the project aimed at finding solutions that could be widely deployed in schools, homes etc. Therefore, the cost of reading devices should be so low that they would almost be at the level of disposable technologies, i.e. only a few Euros. Consequently, the project devised a solution that avoided expensive optical components such as the camera-based tracking used in other approaches. Due to the size of the barcodes encoding the positional information, the resolution of the current inductive pen is effective for enhanced reading only and does not support pen-based writing capture.

The *Digital Pen and Paper* technology [47] for high-resolution paper-based position tracking has been developed by the Swedish company Anoto [6]. Again, the idea is to detect a pen's (x,y) position on a paper document. The position information is directly encoded on each piece of paper, in this case using a special pattern of tiny visual dots as shown in Figure 2.6 (courtesy of Anoto Group AB). One can assume that there is a virtual grid over a page and the dots are printed with a small displacement relative to the intersections of the horizontal and vertical lines of the grid fitted to the dot matrix. Each dot then encodes a two bit sequence which is defined by its horizontal and vertical displacement from the corresponding intersection point. Several dots together form a unique sequence of zeros and ones which defines a position in a large virtual document space. The dot pattern results in a slightly grey page background with minimal interference with the document's content.

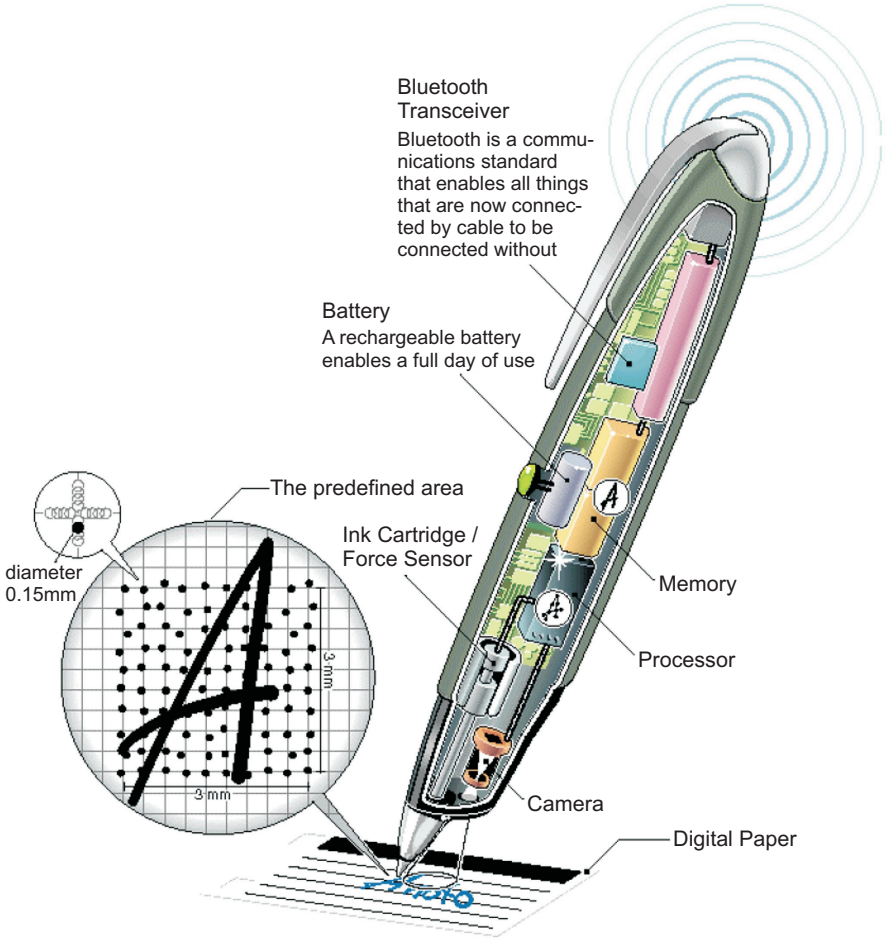


Figure 2.6: Anoto technology

A special digital pen for the Anoto dot pattern has to be equipped with a camera in addition to the writing stylus to track the pen's movement relative to the paper surface. A record of the pen's movement can then be used to recreate what a user has written within the digital world. The capture of additional information such as the pressure of the pen nib, orientation and tilt can be used to enhance the rendering process and create images that realistically recreate the look and feel of hand-written text and sketches. These additional parameters together with the timing information further can be used in applications dealing with signature authentication. Note that information is not simply stored as static images but the writing can be replayed using the corresponding software tools.

Several Anoto pens from Sony Ericsson (*Chatpen*), Logitech (*io Personal Digital Pen*) and more recently from Nokia (*Digital Pen*) are now available on the market. Currently, there is a clear focus on writing capture since a user's actions are stored as positional information within

the pen and only transmitted to another device on demand. However, this technology has the potential to also be used for enhanced reading and direct interaction: It only requires that the position information be transmitted continuously rather than in batch mode.

The *Universal Pen* (uPen) [72], based on similar concepts as implemented by the Anoto pens, is currently being developed at Microsoft's Beijing lab. The uPen contains a camera, some memory and a Bluetooth wireless connection. As in the case of the Anoto solution, the paper has to be covered with a special background pattern which is then used for position detection by the pen.

2.2.3 Writing Capture

Many of the technologies for position tracking presented in the previous section, for example pens based on Anoto technology, can also be applied for capturing handwritten information. We now present solutions which are suited for writing capture but not for position tracking.

A well established technology for writing capture from paper documents are *desktop scanners* and their smaller counterparts, i.e. *handheld scanners*. In desktop as well as handheld scanners, a row of charge-coupled device (CCD) image sensors is moved over the paper documents thereby detecting the printed information. Scanners are mainly used for digitising paper documents and less for augmenting paper documents with digital functionality. Drawbacks of these scanning technologies are that they are not portable, information is not captured in real-time and the documents have to be processed by the scanner in an additional step.

Smaller parts of printed documents can be captured by pen-like devices with an integrated camera. These camera-equipped pens are mainly used for capturing single printed words. For example, the *MyPen* by C-Channel AG [32] can read the Anoto pattern but also works as a capturing device for printed text such as account numbers printed on payment orders. It is tethered to a computer and immediately transmits new pen input. The pen is sold together with a mouse mat with predefined "active areas" as shown in Figure 2.7(a). If the user points to an active area covered with the Anoto pattern, a specific action will be triggered. In the case of the mouse mat deployed with the MyPen, users can define physical bookmarks for favourite websites which will be loaded as soon as they point to the corresponding active area. Further, there is a larger area in the centre of the mouse mat that can be used to move the mouse cursor on the computer screen.

(a) MyPen (b) Quicktionary

Figure 2.7: Camera-equipped pens

Often the captured words do not have to be transmitted to a computer but can be processed directly on the pen. For example, a captured word can be translated to another language which is then shown on a small display integrated into the pen. Figure 2.7(b) shows the *Quicktionary*, an example of such a camera-based translator pen manufactured by WizCom Technologies [33]. Of course, such a camera-based capture approach can be used, not only for digitising printed information but also to capture handwritten information, if combined with a ballpoint pen as done in the *Memo-Pen* [119].

2.2.4 Information Storage

Instead of storing unique identifiers or positional information on paper documents, which can then be used to look up corresponding digital information, it is also possible to encode and store digital information directly on the paper documents. The *PaperDisk* [137] and the *Strip-Reader* [37] are two such technologies for storing digital information on paper documents.

The initial goal of the Strip-Reader developed by Datasound was to augment photographs with additional audio commentary which could later be accessed by a special reader device. The current version not only supports augmentation with audio files but with any type of digital media. Data is encoded within rectangular strips with a size of 55 by 18 millimetres. Each strip contains a two-dimensional code defined by thousands of tiny dots and therefore is perceived by the user as a slightly grey coloured area. After scanning a single strip, the digital

information is stored on the reader device. While audio files can be directly played by the Strip-Reader, all other types of media have to be transmitted over a wired connection to a personal computer for further processing. Since no computing power is required in addition to the reader device for playing back audio files, the Strip-Reader system represents a very portable solution for augmenting paper documents with any form of acoustic information. The PaperDisk technology is very similar to Datasound's information storage solution and also encodes digital information using two dimensional dot patterns called DataTiles. The major difference regarding the encoding of information is that there is no fixed size for the DataTiles. Furthermore, PaperDisk provides no special reading devices, but off-the-shelf desktop scanners are used for input processing.

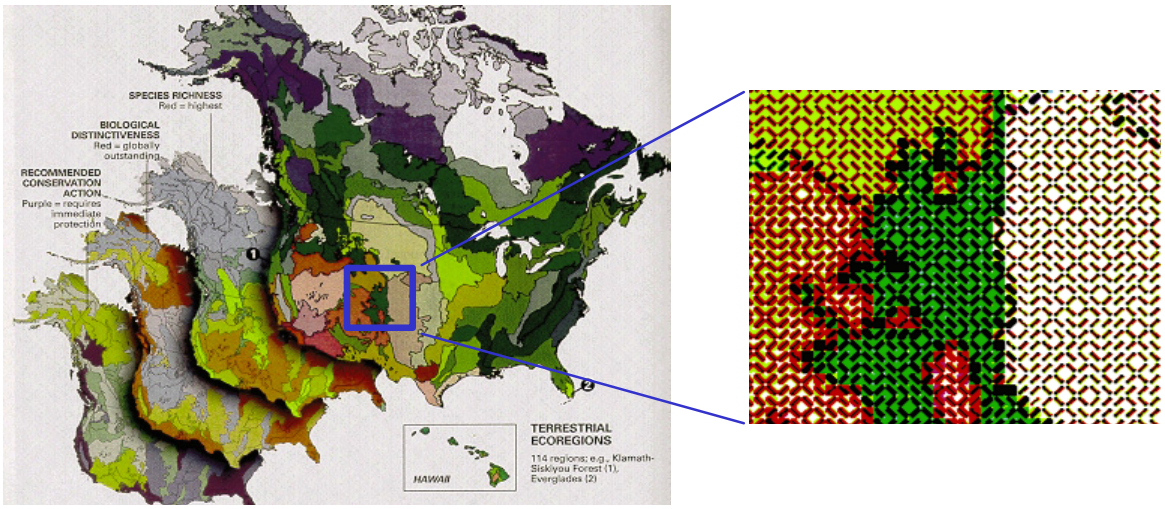


Figure 2.8: Xerox DataGlyph technology

DataGlyphs [68, 80], which were developed by Xerox, can also be used to store digital information on paper using a special printed pattern. The DataGlyph pattern consists of forward and backward slashes representing zeros or ones as shown in Figure 2.8 (courtesy of Palo Alto Research Centre). If the pattern is small enough, it is not intrusive to our eyes and can be integrated into the printed artwork. The encoded information can be extracted by a *GlyphPen* scanning device and further processed by a computer.

Essentially, any type of high-resolution two-dimensional barcode can be used for encoding larger amounts of digital information directly on paper documents. We only have to be careful that the printed codes do not interfere too much with the rest of the document content.

2.2.5 Electronic Paper

In addition to the technologies presented for integrating paper and digital information in a way that preserves the best properties of both, some electronic paper technologies aim at replacing traditional paper and print. The development of new display technologies for thin, paper-like output devices is a major research activity in developing electronic paper. New forms of electronic inks and electronic paper are discussed in [173].

E Ink [77], a form of electronic ink, has been developed at the Massachusetts Institute of Technology. Millions of tiny microcapsules contain positively charged white particles and negatively charged black particles. By applying an electric field pointing from the top electrode to the bottom one, the black particles move to the top of the microcapsule, which makes the surface appear dark at that spot as shown in Figure 2.9 (courtesy of E Ink Corp.).

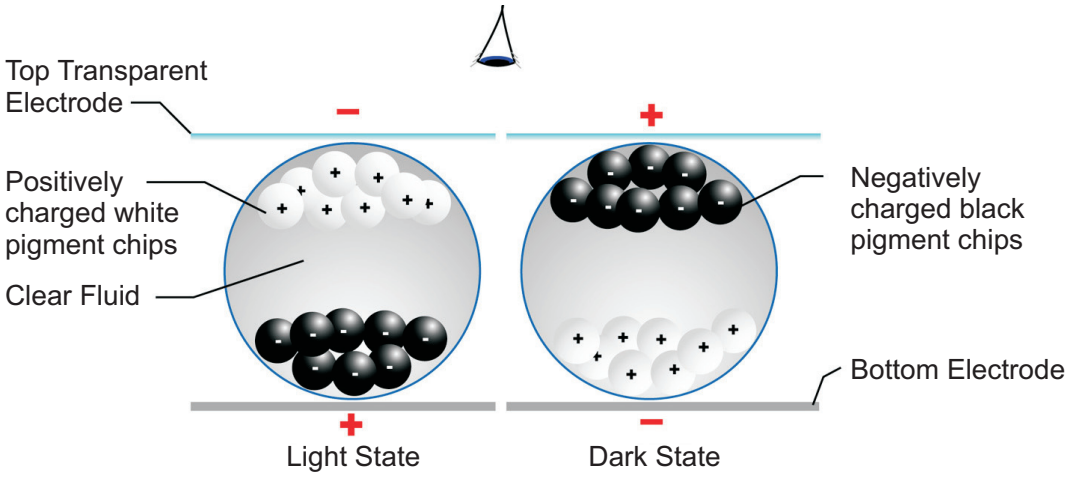


Figure 2.9: E Ink

A similar approach is addressed by *SmartPaper*, a dynamic paper developed by Gyricon Media, a Xerox spin-off company. Millions of tiny beads are placed in oil-filled cavities between two sheets of thin plastic. The beads have hemispheres of two contrasting colours and are electrically charged, forming an electrical dipole. By applying a voltage to the surface, the beads rotate and present one side, which is either black or white, to the viewer. After removing the voltage, the image stays without consuming any power until a new voltage pattern is applied to rearrange the previously generated image.

Electrowetting [66], developed at Philips Research Laboratories, is a novel principle for building reflective displays. By applying an electric

field to a coloured oil droplet that has been positioned between two glass plates, the shape of the droplet can be manipulated and an optical switch with a high reflectivity and contrast ratio can be realised. Flexible electronic displays with fast response times, even suitable for video-playback, can be constructed by combining thousands of these optical switches. Numerous other research labs such as Acreo AB and Plastic Logic are also developing flexible display technologies.

These technologies for electronic paper imitate at least some of the properties of paper documents. However, there are still many advantages of paper over electronic paper solutions. Normally, it is assumed a user has one electronic book on which different electronic documents can be read. As we have seen earlier, one of the key affordances of paper documents is that one can easily work with several documents simultaneously reading across more than one document at the same time by juxtaposing multiple documents next to each other. The physical space can be used to define a spatial order on the documents representing a knowledge worker's current activities. All this is hard to achieve with electronic paper without the use of multiple electronic books.

Even if we had multiple of these electronic books available, it is, for example, not possible to tear electronic paper to distribute parts of a document as easily as with paper documents. Furthermore, the power consumption of electronic paper, which is crucial in mobile applications, is a problem still to be solved. Therefore, we believe that electronic paper will have its specific application domains but will never replace paper documents entirely. The best is to use paper and digital media, deciding when to use which based on appropriateness to the task at hand. In this way, the gap between paper and digital information spaces can be bridged and the strengths of each combined.

2.3 Applications

In this section we present various interactive paper applications which are based on one or more of the technologies described. We have classified the projects by the types of activities they support and start by presenting applications where the focus is on the reading activity before going on to discuss projects dealing with writing activities. Notice that quite often, reading and writing activities cannot be separated but rather form part of a combined annotation activity and therefore we devote a separate section to the description of such projects.

We then go on to introduce projects where the focus is neither on the reading nor on the writing activity, but instead on an integration of paper with digital services and applications. The resulting paper-based interfaces to digital applications often provide interaction with computer systems in a very natural way. Finally, we describe the evolution of hypermedia systems as we know them, for example, from the Web. A generalisation of the hypermedia concept leads to *physical hypermedia* where paper documents and other physical objects become "physical bookmarks" for digital web pages.

Note that while discussing different applications, we reference technologies introduced earlier in this chapter. We further give comments about specific advantages and disadvantages of the chosen approaches and compare different applications.

2.3.1 Reading

The goal of the projects presented in this section is to enhance the reading experience by augmenting paper documents with supplementary or related digital information. While the printed information is static, the digital information can change dynamically and updates can be propagated, i.e. the paper documents can be used as a summary or overview which is bound to dynamic digital information. In addition to printed text, the digital media offers functionality not available on paper documents, such as the playback of sounds or movies.

Many of the projects presented in this section support the definition of paper-based bookmarks to digital web resources in a manner similar to digital bookmarks used in web browsers. All of the projects represent rather "simple" applications in terms of the functionality supported by the paper interface. The invocation of a single web resource is the only task that most of these "physical bookmark" solutions support. The linking from physical to digital information is straightforward and most of the presented systems, not including those based on position dependant link source anchors, use a simple mapping of unique identifiers, such as barcodes, to the corresponding URLs as shown in Table 2.2.

A commercial solution for such paper-based bookmarks to digital information is offered by *PaperClick* [121]. Printed shopping catalogues and other physical objects become linked to the corresponding information in an online-store or other digital information available on the Web. PaperClick uses linear barcodes which are placed next to the printed object which has to be linked. By sweeping the barcodes with a barcode

Resource ID	Resource
01237685121	http://www.cnn.com
01426782492	http://www.globis.ethz.ch/research/paper/
04632534266	http://news.bbc.co.uk
18735676434	http://www.anoto.com
...	...

Table 2.2: Simple mapping of identifiers to resources

reader, users get direct access to related information on the Web. If a barcode reader is temporarily unavailable, the unique number represented by the barcode, which is also printed in human-readable form next to the barcode, can be manually typed into the user interface to get direct access to the linked web page. In addition to using special barcode readers, the barcodes can also be processed by the built-in cameras available in newer mobile phones and the information associated with a physical object can be accessed directly from the mobile phone as shown in Figure 2.10(a) (courtesy of NeoMedia Technologies).

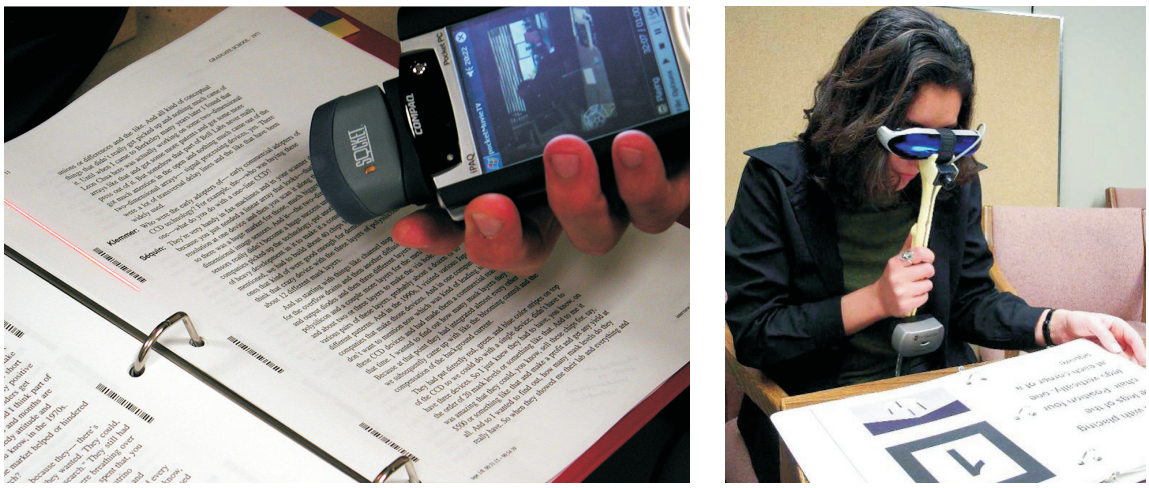
(a) PaperClick (b) Wiziway

Figure 2.10: Paper-based bookmarks

Paper bookmarks which can be linked to arbitrary web pages providing digital information are also supported by *Wiziway* [189]. The Wiziway system shown in Figure 2.10(b) (courtesy of Wiziway) differs from the PaperClick project mainly in how the physical link source anchors are defined on paper documents. Instead of using barcodes, the Wiziway approach is based on small symbols, so-called pictograms, which

are printed on the paper documents. These pictograms are decoded by the Clicker, a special reading device, and the information encoded in the pictogram is processed to fetch the appropriate digital information as described in the previous project.

PaperClick as well as Wiziway use visible barcodes or pictograms to link paper and digital information. While this solution may be appropriate for shopping catalogues and other brochures, it is not suitable for augmenting most reading books. The visible marks cannot be placed within a piece of text and even if there is place for the tags, they become visually intrusive as soon as we want to link various parts of a single document page.



(a) Books with Voices (b) MagicBook

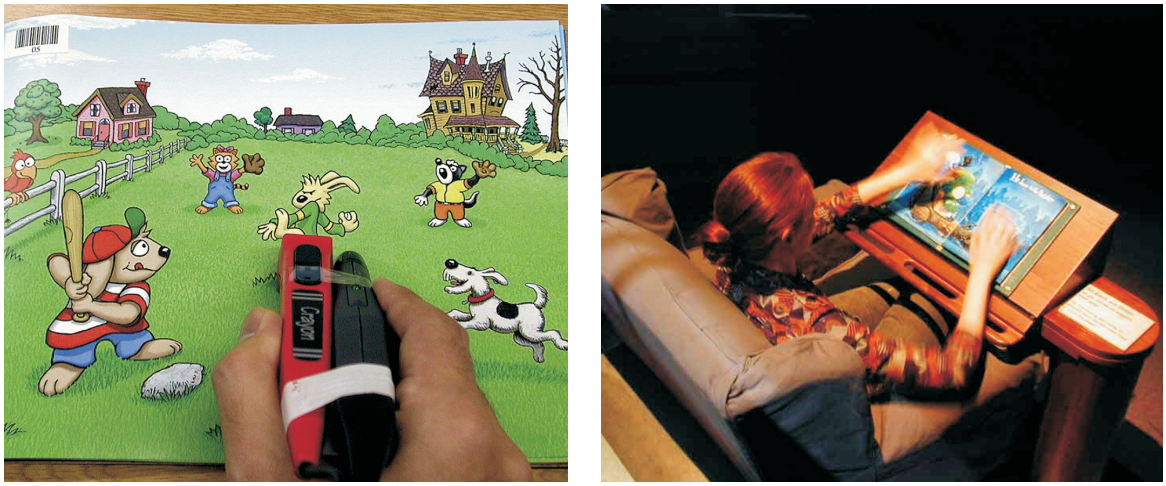
Figure 2.11: Interactive books

A paper interface for fast random access to digital video interviews has been realised in *Books with Voices* [85]. While reading a paper transcript, a user can easily get fast, random access to digital video interviews augmenting the paper document. The enhanced transcripts have been tagged with linear barcodes. A Pocket Computer equipped with a display, audio output and an integrated barcode scanning device is then applied to get fast random access to the supplementary digital information as shown in Figure 2.11(a) (courtesy of Scott R. Klemmer). The same hardware setup has also been used in *Video Paper* [55], a paper-based application for skimming and watching video programmes. An annotated transcript of a TV programme provides links to the corresponding positions in the video recording stored in an archive of TV programmes. The videos can be played either directly on the Pocket Computer or on a video monitor.

The *MagicBook* [19, 20] user interface applies three dimensional models for augmenting printed books and other paper documents. One or multiple users can work with a book in three different ways: They can access the printed information only, overlay and augment the physical content with three dimensional digital information, or switch from the physical information to a completely virtual, computer generated environment. A MagicBook is a regular book with printed texts and pictures on each page as shown in Figure 2.11(b) (courtesy of Mark Billinghurst). Some of the pictures have a thick black border and are used as tracking marks for a computer vision-based head tracking system. A head-mounted semi-transparent display is used for the presentation of supplementary digital information. It is combined with a small camera for detecting the tracking marks. Based on the ARToolKit software, which was introduced in Section 2.2.1, the system computes the position and orientation of a user's head relative to a tagged document and generates the appropriate three dimensional images which are overlaid to the printed artwork. By pressing a special button, the user can switch the head-mounted display non-transparent and immerse into a completely virtual world. Different books for various application domains, ranging from architecture and scientific visualisation to general entertainment, have been designed for the MagicBook framework.

The *Active Book* [164] is a point-and-click application, which allows a user to access digital information such as sound files or movies by selecting objects in a storybook with the FieldMouse input device as shown in Figure 2.12(a) (courtesy of Itiro Siio). The reader of an active book can point at any position within the booklet and is no longer limited to select only specific locations which have been tagged with a barcode. The system uses clickable maps which are provided by the Hypertext Markup Language (HTML). For each physical page of the Active Book there exists an HTML document containing a clickable image map with links to different digital resources. By scanning the barcode that is printed on each page of the Active Book, the corresponding HTML page is selected. The FieldMouse's position then directly provides the input for the clickable map which in return gives back the linked web resource.

The *Listen Reader* [11, 12] presented in Figure 2.12(b) (courtesy of Maribeth Back) uses audio output in the form of ambient sounds to augment the reading of storybooks. In the setup of the Listen Reader, a chair is used in combination with a table placed over the armrests. On top of the table there is a fixed book. A reader can access and play ambient sounds by touching parts of the specially designed book



(a) Active Book (b) Listen Reader

Figure 2.12: Interactive books

containing large images and pieces of text. The multimodal interaction design interweaves the modality of sound into the book's storyline. In contrast to other solutions, the Listen Reader tries to avoid a so-called *control button methodology* where the user has to choose a specific element within a page to play the appropriate sound. The four sounds associated with each page — one for each quadrant — are played in a continuous loop and a reader's task is to mix them in a satisfying way with respect to the story by placing their hand in one of the four quadrants. After turning a page, the ambient sound is faded to a new set of looping sounds but the continuous flow is never interrupted. A reader can further control the volume of the sounds by the distance of their hands from the book and fade between different sounds.

The implementation of the Listen Reader uses two different technologies for detecting the page which is currently open and for tracking the position of a user's hand within a single page. Each page contains an embedded passive RFID tag. An RFID tag reader is attached to the book's binding in such a way that it can detect the tags of the pages that form the right hand side of the opened book. By turning a page from the right to the left in a regular page turn, the tag of that page leaves the range of the tag reader which thereby can automatically detect any page turns. In addition to this page detection mechanism, electric field sensing is used to track a reader's hands. There is an electrode in each of the four corners of the book establishing an electromagnetic field in each quadrant of the book. A sensor then detects and tracks a user's hand and assigns it to one of the four quadrants.

The *Interactive Textbook* [93], an application of the EnhancedDesk, is an augmented physics textbook. While a user is reading the textbook, digital data and animations of experiments are projected next to the textbook. The user can manipulate the digital data with their hand and change parameters of the experiments. The Interactive Textbook links single pages to digital applications by tagging them with a two-dimensional barcode.

Figure 2.13(a) shows the *LeapPad* [97], an educational toy which supports learning through interactive books that play sound files in response to a child pointing to areas within a book. The book is fixed on a special tablet and it is the absolute position of the "magic pen" over this tablet that determines the position pointed to on a page. For every pair of open pages there is a special 'Go' button which has to be selected before working on these pages. The position of this button varies which enables the LeapPad to detect the pair of pages on which a user is currently working. More recently, a modified version of the LeapPad has been used in developing countries for delivering healthcare information to illiterates.

(a) LeapPad (b) Sony LIBRIé

Figure 2.13: Interactive books

Electronic books (e-books), another technology for enhanced reading, aim to replace paper by paper-like electronic devices, thereby profiting from new functionality offered by digital books such as the possibility to hyperlink parts of a document. Various PDA-like e-book reader devices, such as the *Rocket eBook*, came on the market a few years ago. The idea is that a customer can download a digital version of a book and read it on the screen of an electronic device. However, e-books have not been very successful so far and this is not only due to the limited amount of

content currently available and incompatibilities across different reading devices. Real paper still has many advantages since it is light and flexible and no battery or networking facility is required. In addition, everybody knows how to read a book whereas e-books provide a slightly different user interface including, for example, electronic page turns.

The *Sony LIBRIé*, a new generation of electronic books based on E Ink technology, came on the market two years ago. The LIBRIé, which is shown in Figure 2.13(b) (courtesy of Sony Corp.), stores up to five hundred downloaded books and can even be read in bright sunlight. Unfortunately, Sony defined its own Broad Band eBook (BBeB) format which makes it impossible to display content that has been developed for other e-book readers.

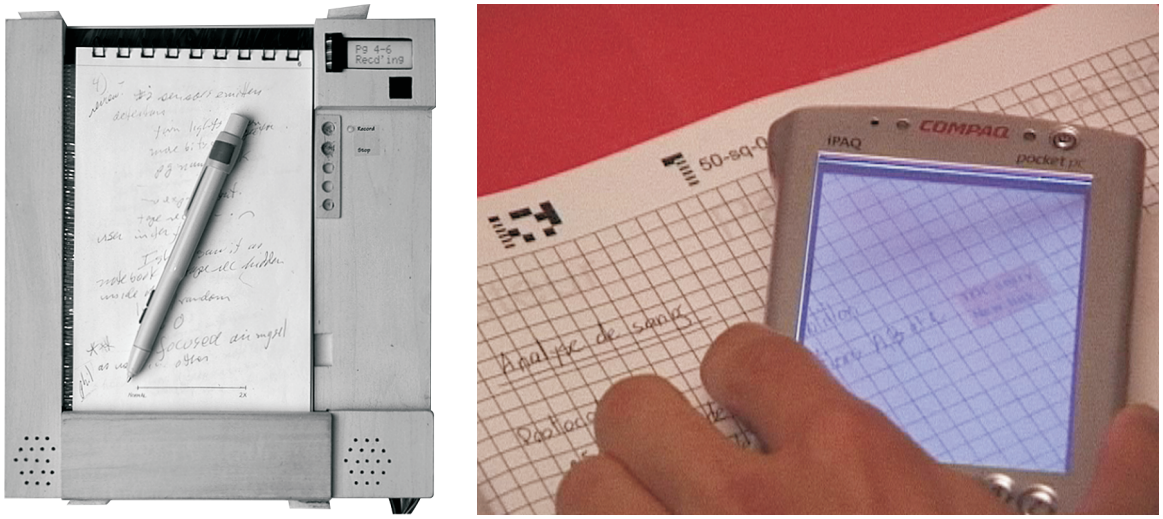
SyncroSigns are message boards and display signs for retail applications based on Gyricon Media's SmartPaper technology. The SyncroSign displays are updated over a wireless interface. Electronic price tags which can be updated automatically are just one of the potential application domains of SyncroSigns. A SyncroSign consumes power only while the content is updated which results in longer battery life time. One of the reasons why the SyncroSigns are not yet very widespread is that they are relatively expensive and require a special server architecture for content updates.

In this section we have presented projects dealing with enhanced reading where additional information can be retrieved or actions can be triggered by a paper-based user interface or by interacting with an electronic paper document. In the next section we discuss applications where not only can digital information be retrieved, but also enhanced writing is supported by means of paper-based writing capture.

2.3.2 Writing

The process of taking written notes has always been a method of recording and reflecting on knowledge. Students normally take notes during lectures for later review or annotate the lecturer's handouts with supplementary information. Often less attention is paid to audio recordings of whole lectures since it is hard to later retrieve specific parts from a long audio stream on the one hand and to have an overview on the other hand. The capture of notes during lectures has been addressed by the *Audio Notebook* [167, 168], a combination of a digital audio recorder and a paper notebook. The Audio Notebook, which is shown in Figure 2.14(a) (courtesy of MIT Media Laboratory), has been designed for taking notes

and automatically linking them to speech recording. Audio files recorded during a lecture are synchronised with the user's handwritten notes. The notes later serve as fast indexes for retrieving parts of the recorded audio stream.



(a) Audio Notebook (b) A-Book

Figure 2.14: Enhanced writing

A document identifier and page number are encoded using a binary code where each bit is represented by a black or white square. This encoded information is printed at the bottom of each notebook page and is automatically detected by optical sensors attached to the underneath of the Audio Notebook's hand rest. A digitising tablet then tracks the pen's position within a single page.

The indexing of handwritten notes has also been addressed in Hewlett-Packard's *Filochat* prototype [185]. However, in contrast to the Audio Notebook, the notes are taken on a graphics tablet rather than a paper notebook. *Marquee* [180] uses a similar approach to apply pen recorded notes for video indexing whereas in *FAME* [29] handwritten paper notes are used to index arbitrary multimedia data captured during meetings.

Dynomite [188] is an electronic notebook for capturing and retrieving handwritten notes and audio recordings similar to the Audio Notebook. It replaces the paper notebook which was used as an input device in the Audio Notebook by a digital pen-based input device. The notes written on a digital screen can be linked and augmented with audio recordings. In contrast to a paper-based solution, the digital notes taken by the Dynomite system can be altered dynamically. The notes can be

annotated with special properties such as names, URLs etc. for classifying captured handwritten information. Information can later be retrieved based on the assigned properties, timestamps or specific keywords.

In many research laboratories, notes have to be written down to document the setup and outcome of an experiment as well as observations made during an experiment. The *A-book* [107] is an augmented laboratory notebook supporting research biologists in their daily work. It consists of a paper notebook which has to be placed on a WACOM graphics tablet together with a Pocket Computer as shown in Figure 2.14(b) (courtesy of Wendy Mackay). The Pocket Computer is equipped with a 4D mouse and acts as an "interaction lens" or window between the physical and the digital world. The graphics tablet detects and captures a user's pen strokes. In addition to the (x,y) position, the 4D mouse delivers information about the fingerwheel position and the rotation of the mouse which helps tracking the position of the interaction lens relative to the graphics tablet. By placing the physical interaction lens on the paper notebook, the biologists get access to additional digital information which overlays the information written on the paper documents and is shown together with the captured handwriting on the display of the interaction lens.

The user can create new content by writing on the paper notebook with an inking pen which is tracked by the graphics tablet. The information written in the A-book can be read as in any paper notebook without special technology being required. In addition, information written in the paper notebook can be highlighted, linked or annotated with digital information by using a non-inking pen in combination with the interaction lens. While reading the paper notebook, supplementary digital information can be accessed on the interaction lens by placing it over the corresponding parts of the A-book as shown in Figure 2.14(b). The interaction lens displays a digital version of the part of the page over which it is placed, enriched by links to supplementary digital information.

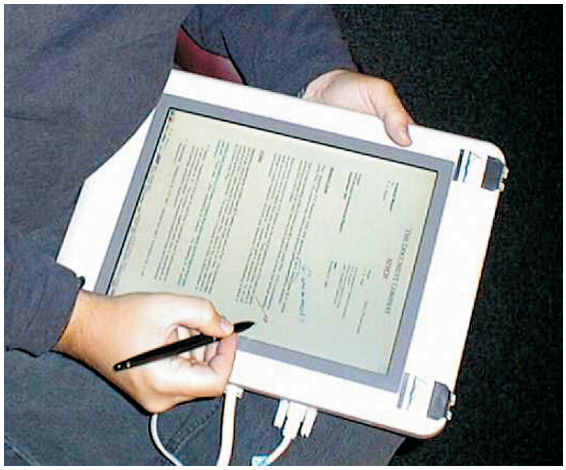
Hewlett-Packard is selling a *Forms Automation System* (FAS) [31] based on Anoto functionality which integrates data filled in a form into a company's business database. *ExpeData*, another commercial forms processing solution based on Anoto technology, is offered by Standard Register [146]. A business solution for transportation companies based on Anoto's digital pen and paper technology is provided by Fruits [49], a Danish company. The *Fruits Transportation* solution is already used by the DHL international transportation company in multiple countries. Each DHL agent is equipped with a Nokia Digital Pen which is used by

the recipient to sign off on a shipment. The signature is immediately transmitted to the DHL head office via a Nokia mobile business device resulting in up-to-date delivery status information.

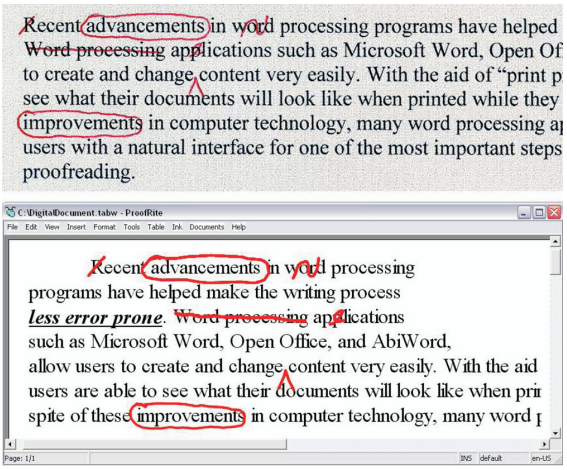
In *Nostos* [10], Anoto functionality is applied to build an experimental ubiquitous computing healthcare environment. Physicians fill in Anoto-enabled paper forms to enter new data and interact with the information system. They get feedback via numerous displays in the physical environment and a radio-enabled headset.

2.3.3 Annotation

The *XLibris* system [144, 154] is an *active reading machine* which is shown in Figure 2.15(a) (courtesy of FX Palo Alto Laboratory). XLibris has a pen-based interface and runs on a tablet computer to make it as paper-like as possible. The main goal of XLibris is to enable deep reading of electronic documents by supporting free-form annotations. The system always shows a single page at a time and supports different forms of document annotation such as writing comments or underlining and highlighting words. The annotations are made directly in the digital document but later they can be reviewed separately in the *Reader's Notebook*, a special view showing annotated text snippets. Further, a user's annotations are processed and used to dynamically generate links to material for further reading based on common information retrieval (IR) techniques. Finally, the system provides a skimming mode where characteristic phrases and sentences are automatically highlighted.



(a) XLibris



(b) ProofRite word processor

Figure 2.15: Annotation

Electronic notebooks for improving the learning experience in classrooms have been investigated in *Classroom 2000* [1]. The system distinguishes three different phases: In a pre-production activity all information leading up to the classroom interaction is prepared. During the lecture, different streams of information are captured by live recording. The students annotate existing lecture material and share their comments. Multimedia-enhanced web pages are created to summarise the classroom activity as part of a post-production step. Numerous other projects, for example the *MATE* [63] project, also dealt with collaborative markups and comments on digital documents. The *intelligent pen* metaphor [54] is a more recent approach to uniformly treat markings in electronic documents. Thereby a variety of markings such as correction marks, annotations, marking by underlining etc. are supported. All document markings are stored together with the original electronic document and can be shared with other users.

Paper Augmented Digital Documents (PADD) [60] are documents that can be manipulated either digitally or on paper based on Anoto functionality. After printing out a document on Anoto paper, it can be annotated with a digital pen and the strokes can later be synchronised with the digital version of the document. The upper part of Figure 2.15(b) (courtesy of Kevin Conroy, Dave Levin and François Guimbretière) shows an annotated paper document whereas the lower part of the figure presents the updated digital version of the same document. Note that any annotations are embedded as images in the original document but not directly integrated with the textual content by applying optical character recognition. This edit cycle can be repeated multiple times and subsequent printouts of a document will always include changes made to a previous paper version.

2.3.4 Paper-Based Interfaces

In this subsection we discuss paper-driven applications and related projects which extend the enhanced reading concept by integrating paper with digital application user interfaces.

Paper-driven access to a document storage and retrieval system has been realised in Xerox's *PaperWorks* product [78] based on DataGlyph technology. Specially designed forms, containing information encoded as DataGlyphs, are attached as cover sheets to paper documents. The documents are processed in full-page batch mode by a scanner or fax machine which is connected to a document services system, the *XAX* server,

and the glyph encoded information is used to invoke appropriate actions on the subsequent paper document. For example, by marking the intended recipients on such an interactive cover sheet, a document can be distributed automatically to several people. While the XAX server provides some basic functionality for processing documents containing DataGlyph encodings, specific applications can be implemented on top of the XAX architecture.

The *Protofoil* system [145] extends the XAX server to cope with various paper filing and distribution needs resulting in an electronic filing system for paper documents. Based on the same idea of specially designed cover sheets with embedded DataGlyphs, the system provides functionality for storing, retrieving and distributing documents. Further, remote services such as printing, mailing and faxing can be invoked based on the paper user interface. *FlowPort*, a commercial version of the Protofoil system is currently manufactured by Xerox. Various applications based on the interactive cover sheet approach have been investigated in *Printertainment* [71]. They range from simple games, such as a Tamagotchi or an application generating stories based on single words entered by the users, to *Cover Notes*, a lightweight group communication application which can be accessed from printed cover sheets while waiting in the printer room for jobs being printed out. Each cover sheet contains four boxes on the right page margin. Three of the boxes are filled with comments from other users selected randomly from a pool of available notes. The empty box can be filled with a new comment and scanned in by the user while waiting for their printout.

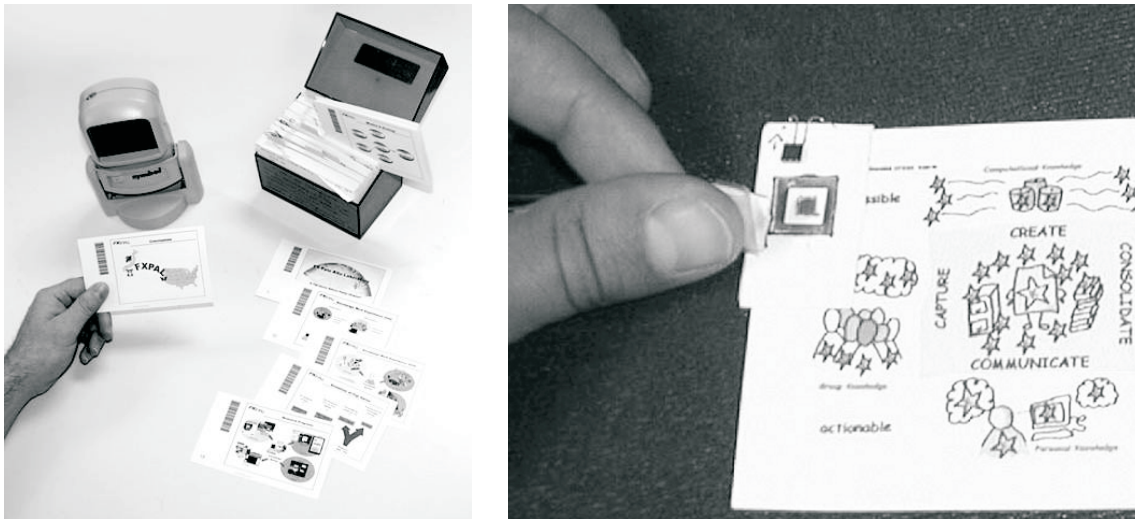
An integration of physical documents and digital activities based on an enhanced pen in combination with special paper is proposed by *Intelligent Paper* [43]. In the presented proposal, standard sheets of paper are entirely covered with printed marks invisible to the human eye but readable by a special reader device. DataGlyphs are suggested as a possible solution for the printed pattern encoding a page number and the (x,y) position within a specific page. Further, Intelligent Paper proposes the use of the Adobe Acrobat suite of products to build an electronic counterpart of the paper document with embedded links for every physical page. Pointing actions on the paper document are then mapped to virtual mouse actions on the display and by pressing a button on the pen, mouse clicks can be accomplished for selecting objects on a page. Every action that the user could have performed by clicking with the mouse somewhere on the screen can also be performed by using the special pen on the paper document. These synchronised paper/digital actions are

proposed because they simplify the processing of pen-based input. Further, a possible extension of the pen for capturing handwritten notes is described.

In the lifecycle of a digital document, a printout always represents a snapshot of the document at a specific point in time. How can we be sure that our paper version of a document is still up-to-date if we return to a document that was printed out some time ago? The *InteractiveDESK* [8], an augmented desk system, has addressed this issue of document management by using paper documents as bookmarks for retrieving up-to-date digital versions of documents. The digital version of a paper document can be retrieved by placing the documents or folders tagged with a visual marker on their cover page at a specific location on the desk. An over-desk camera that is mounted on the ceiling monitors the desk surface and detects any tagged documents which are in its range. The InteractiveDESK can only associate entire objects with their digital counterparts and it is not possible to link parts of a document such as a document page or even a single paragraph. The visual detection of object identifiers further implies that documents on the desk should not overlap for correct detection by the camera.

As pointed out in Chapter 1, paper provides several affordances which make it superior to digital media for many tasks. One problem of digital media is that it is often hard to see various information components at a glance. A good example for this lack of overview is the task of giving a presentation using a digital presentation tool such as Microsoft's PowerPoint application. While giving a presentation with PowerPoint, the focus is always on a single slide. To keep an overview, presenters often print handouts containing small versions of several slides together with keywords supporting the presentation on each page. During a presentation, it is no longer possible to easily change the order of the slides or to switch to a specific slide. A paper-based user interface addressing the issue of giving flexible presentations has been realised with the *Palette* system [36, 120] shown in Figure 2.16(a) (courtesy of Lia Adams).

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



(a) Palette (b) PaperButtons

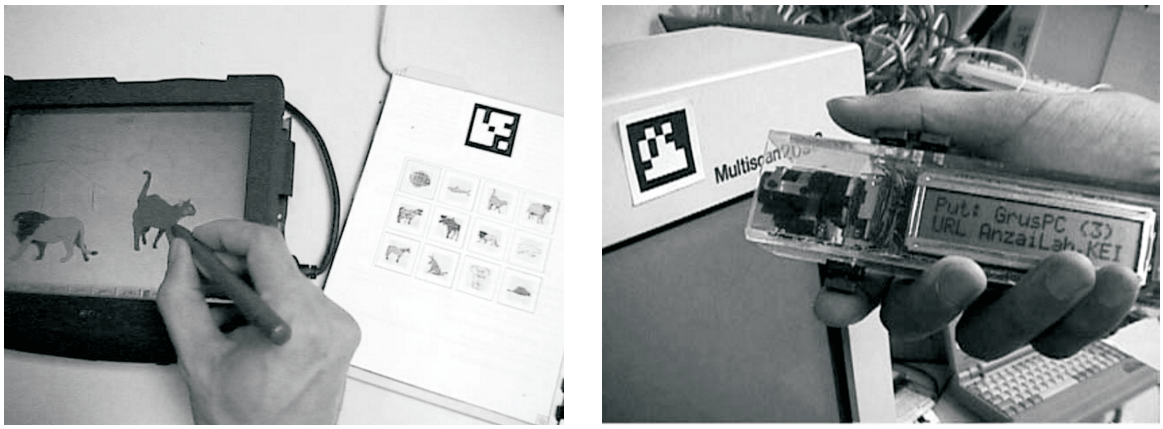
Figure 2.16: Paper-based presentation tools

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 Palette user interface supports new forms of giving presentations, for example collaborative presentations, and enables a spatial layout of the slides on a desk in preparation for nonlinear presentations. 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 also creates problems in that it could sometimes be difficult to find the "next" slide.

Another 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* [141] 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 as shown in Figure 2.16(b) (courtesy of Tomas Sokoler). 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 which transfers the information to the Palette application manager showing the appropriate slide. The wireless solution solves the problem of being restricted to a specific location given by the barcode reader. Multiple

PaperButtons can be attached to a single Palette card providing control functionality such as starting and stopping embedded media clips, controlling animated slides etc.

The same idea of having fast random access to digital information based on a paper user interface is addressed by the *PaperIcons* [148] project. In PaperIcons, the use of a printed clip art book for fast random access to a digital clip art gallery has been automated. While browsing a paper document, a user can pick printed icons and drop them on a pen-sensitive computer screen. This pick-and-drop operation is quite suitable for applications where a user can, for example, select clip arts from a physical book and integrate them in a digital presentation. The PaperIcons project uses CyberCodes [149, 150], a form of 2D barcodes, to uniquely identify different document pages as shown in Figure 2.17(a) (courtesy of Jun Rekimoto). 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. A camera is mounted to the ceiling over the table to read the CyberCode attached to each document page encoding its page number. 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.



(a) PaperIcons (b) InfoPoint handheld device

Figure 2.17: Pick-and-drop interfaces

The idea of using CyberCodes to connect physical and digital information has been generalised in the *InfoPoint* handheld device [91] which is shown in Figure 2.17(b) (courtesy of Jun Rekimoto). The InfoPoint is a CyberCode reader combined with a small display. It enables physical real world objects to work together in terms of data transfer based on drag and drop. For example, the address of a web page can be read from a

paper document by scanning the corresponding CyberCode and the web page is loaded in a web browser by scanning the CyberCode attached to the computer screen on which the web page has to be shown. The same approach is also applied for transferring files between computers or other devices such as digital cameras and printers.

So far we have seen how paper documents can be linked with digital information or functionality available on a computer and possibly be augmented with additional information. They can also be used to interact with and control other physical objects in a working environment. *Real-world graphical user interfaces* (RWGUIs) were introduced by Masui and Siio [112] as paper-based interfaces for arbitrary physical devices. Customised paper-based remote controls can be defined and accessed via the FieldMouse. An example of a paper-based TV volume control is presented, where a special barcode has to be scanned first 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 easily be customised.

The mobility of paper documents plays an important role in the *Campiello* project [56, 90] where paper artefacts containing tourist information about different cities have been augmented with digital information. Four different paper artefacts have been designed: a tourist guide, maps, flyers and newspapers. The tourist map becomes personalised when printed out based on information stored in a user's profile. The tourist guide is also a folder for other Campiello paper forms requested or collected in town. The customised Campiello maps contain a set of suggestions based on a user's profile. New maps can be generated during a visit based on what a user has already done and also the location in which the new map is requested. Flyers are one-page documents containing information mixed with special areas providing means of interaction with the system that are spread around the city. Finally, customised newspapers can be printed based on a user's preferences. Users can write down comments and rate specific events. These annotations are processed by a computer, using DataGlyph technology and made available for other visitors. However, in Campiello, the DataGlyphs are only used for document identification whereas standard OCR is applied to capture new information written in a document. Note that users of the Campiello system do not have to carry any special devices with them, since the additional digital information can only be entered and accessed at fixed access points distributed over the city.

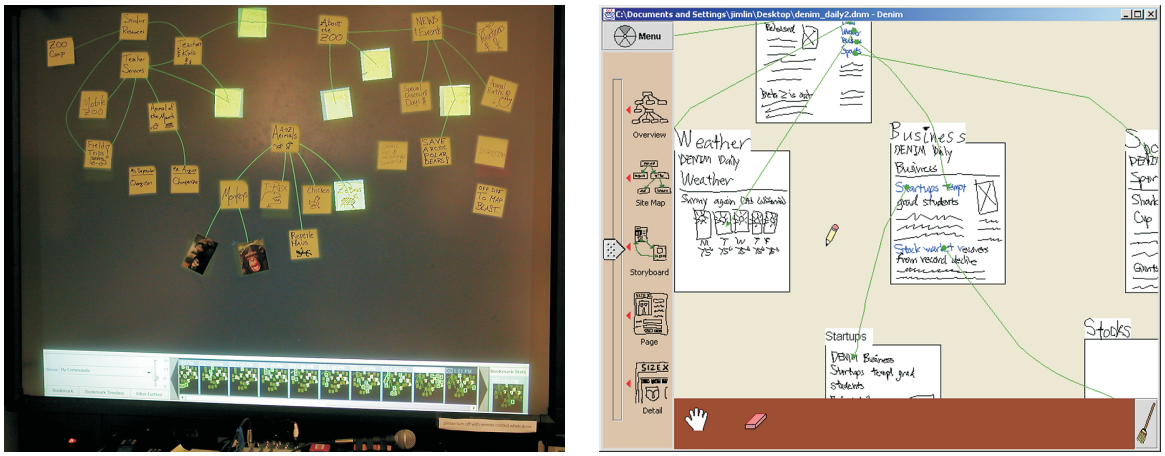
Video Mosaic [105], an application and extension of Pierre Wellner's DigitalDesk, has been designed to support video producers in managing their film material. They often use storyboards consisting of a set of elements, one for each shot, to arrange their scenes on paper. Each element of the storyboard contains a sketch or a key image of the associated video material, some notes that describe the action etc. The paper storyboards support the spatial representation of a fundamentally linear video by placing single elements of the story on separate cards. The designer can lay out the cards on a desk to get an overview of a lengthy part of the story and easily rearrange parts of the film by reordering the cards and also write comments on the cards. However, to actually view the video material, the producers have to switch to a separate video-editing system for accessing the film material. Video Mosaic bridges the gap between the paper storyboards and the video material by digitally augmenting the entries of the storyboard and linking them with the video-editing equipment. Handwritten annotations are captured by a video camera mounted above the desk and integrated into the digital version of the storyboard. In addition to the LCD video projector mounted above the desk to project information from the storyboard as well as menu commands, a video TV monitor for playing the videos is installed directly in the desk. Finally, there is an ink-jet printer on the desk for quickly generating new paper storyboard elements.

The augmentation of paper blueprints with digital information and an integration of this into a *Computer Aided Design* (CAD) system has been investigated in *Ariel* [106]. Given the tight schedules when designing a building, small changes are often only recorded on an engineer's personal paper drawing while the update of the digital version happens much later. A major issue of this work practice, the mismatch between the paper and the digital CAD version, is addressed by Ariel. The problems outlined are related to consistency management issues between digital and printed document versions addressed by PADD. Each paper drawing has a barcode for its unique identification. The different versions of Ariel use different technologies such as digitising tablets and video cameras for writing capture. An LCD projection panel is applied to superimpose the digital information directly on the paper documents. By using the projected digital menus, an engineer has direct access to the digital CAD system and can perform updates from the paper-based user interface.

Early stages of the architectural design and sketching process are supported by an application based on another augmented desk platform, the *Visual Interaction Platform* (VIP–3) [3, 4]. The paper is placed on

a digital drawing board and an electronic pen which does not leave any physical traces on the paper is used for the sketching. The pen's information is digitised and the information is projected onto the physical paper by a ceiling-mounted projector. The project further investigated 3D interaction based on physical handles for analysing volumetric scientific datasets.

In an early phase of website development, designers often collect ideas about what should be on a website on Post-it notes and arrange them into categories on a large, shared physical wall. This collaborative design process of websites has been investigated in the *Designer's Outpost* project [44, 86]. The Designer's Outpost integrates paper and digital information by linking the Post-it notes to their automatically created digital counterparts as shown in Figure 2.18(a) (courtesy of Scott R. Klemmer). The designer writes their ideas on Post-it notes and places them on a touch-sensitive smart board. A video camera mounted in the rear of the smart board interactively locates the Post-it notes based on computer vision algorithms, while a high resolution digital camera mounted in the front of the smart board captures and digitises the information written on the Post-it notes.



(a) Designer's Outpost (b) DENIM

Figure 2.18: Paper-based website development

By placing a Post-it note on the smart board, a digital version of the note, containing the same information as written on the paper note, is generated. The physical note works as a handle for the digital object and moving it around causes the digital counterpart to be moved too. The designer can define connections between different objects by drawing lines with a special pen on the smart board. By tapping a paper note, an

electronic context menu is invoked from which the designer can choose different commands and, for example, delete the digital object. Finally, the smart board can be used as an electronic whiteboard where the designers can place their free-form annotations. The digital information space can be shared remotely among different users and can be further processed by digital tools such as *DENIM* [99], an informal tool for early stage website and user interface design, which is shown in Figure 2.18(b) (courtesy of James Lin).

The same idea of using paper Post-it notes to physically organise information is applied in *Rasa* [113, 114], a tangible and multimodal system supporting military command post work. When an officer receives information from a new unit, they write the information together with a predefined symbol representing the unit on a Post-it note. All information written on the Post-it note is captured by a digitiser tablet. Finally, the officer puts the Post-it note on a paper map which has been taped to a large touch-sensitive digitising board with front projection. By placing the Post-it note on the paper map, its associated digital counterpart is also positioned accordingly in the digital system. In addition to the paper-based user interface, commands can be invoked by voice recognition and voice feedback is generated by a text-to-speech engine.

A team environment linking paper, displays and data has been realised in *Insight Lab* [96]. Barcodes are used to enrich paper and whiteboard printouts with multimedia data presented on the numerous display devices positioned within a lab. The paper documents are linked to information managed by the Insight Multimedia Analysis Software Tool (*Insight MAST*). A barcode command slate — a paper sheet filled with barcodes and associated commands — can be used to trigger different operations. Insight MAST further supports the production of customised annotated printouts for information stored in the database. For example, customised reports can be generated using various criteria for filtering or sorting. Each item of such a report contains a barcode identifier linking to additional digital information in the form of video or audio clips.

Various other projects, for example the *mediaBlocks* project [176], use RFID tags for identifying arbitrary physical objects and augment them with digital information or operations. The use of RFID tagged paper documents in museum settings has been investigated in *SHAPE* [48]. Museum visitors write their comments on paper documents which at the same time can be used for interactions with multiple and diverse displays placed within the museum. By placing an RFID tagged piece of paper next to a display, a visitor gets access to information bound to the paper

document. Further, the system can provide context-dependent information based on previous user activities and displays visited in advance. In the SHAPE project, as well as in most other RFID tag based solutions, the tags identify entire documents and it is not possible to address parts of a document which can then be linked to different information or actions.

2.3.5 Physical Hypermedia

After presenting various projects linking from physical objects to digital information it is worth thinking about an extension of the hypermedia concept. While a first evolutionary step was the switch from simple hypertext to hypermedia, where arbitrary digital resources can be linked, the next step leads us from hypermedia to *physical hypermedia* where the digital space is extended with physical objects. In physical hypermedia, real world objects become physical bookmarks that can be linked to related digital hypermedia objects.

(a) WebStickers (b) IconStickers with label printer

Figure 2.19: Physical bookmarks

The *WebStickers* project [70, 100] allows paper and other physical objects to act as bookmarks to digital information available on the Web. Instead of worrying about clever solutions for creating and maintaining digital bookmark lists, the project proposes the use of the physical environment to manage such lists of bookmarks as shown in Figure 2.19(a) (courtesy of Lars Erik Holmquist). Therefore, the primary application of WebStickers is the handling of personal bookmarks and the sharing of bookmarks with other users. The project further raises the issues of ageing bookmarks. Empirical studies about how people use paper and

electronic documents have identified three types of information based on how long they are being used: ephemeral, working and archived information [14]. Post-it notes with inherent glue aging are proposed as a representation for ephemeral bookmarks.

WebStickers uses visible barcodes for uniquely identifying paper documents and other physical objects. New barcodes can either be printed and attached to physical objects or existing barcodes, already available on many commercial products, can be used. A barcode can be linked to a single web page (URL) or a collection of URLs. When scanning a barcode with multiple associated target URLs, an intermediary web page with a list of the associated web pages is generated. In this way, web pages become coupled with physical representations or tokens. The project introduces the term *token-based access to digital information* and provides the corresponding definition:

> *We will define token-based access to digital information as: A system where a physical object (token) is used to access some digital information that is stored outside the object, and where the physical representation in some way reflects the nature of the digital information it is associated with.*
>
> Holmquist et al. [70]

By reflecting the properties of the associated digital information, the physical object may support later recovery of the appropriate digital information. For example, an online phone book can be bookmarked and the corresponding WebSticker is attached to the phone. By scanning the WebSticker pasted to the phone with a barcode reader, a user can get easy access to the digital online phone book.

The *IconStickers* project [165] proposes the use of real world objects as an extension to our digital working environment by introducing paper icons which are associated with the corresponding digital desktop icons. The main idea is that a user should be able to easily create physical representations of computer icons, which can then be used as proxies for the digital application and spatially arranged in the real world. Therefore, a special icon linking to the IconStickers manager program is placed somewhere on the computer desktop and serves as an "exit to the real world". If a user drag and drops any digital icon to this special exit icon, an IconSticker for the dropped icon is printed on the label printer shown next to the display in Figure 2.19(b) (courtesy of Itiro Siio). In addition

to a unique barcode, the IconSticker contains an image of the associated digital icon and its name. Whenever a paper IconSticker is scanned with a barcode reader, the program or document associated with the computer icon is invoked in the same way as if the computer icon would have been double-clicked. Different scenarios for augmenting real-world objects are suggested by IconStickers: a hard copy of a document can be linked to the original digital version, a printer can be associated with a printer queue monitoring program or a product can be connected with a URL providing detailed product information. IconStickers represents an extension of physical bookmarks presented in the previous section by supporting, not only links to web resources, but also to arbitrary program invocations.

A solution related to IconStickers which, in addition to the invocation of a specific application, also supports the handling of running applications by controlling the mouse cursor is presented by Siio [163]. *InfoBinders* are small devices with a push button and an LED that can either be attached to paper documents and other physical objects or used as stand-alone devices. When a user pushes an InfoBinder's button, the LED emits light which is detected by a ceiling-mounted video camera. The LED's light is modulated so that each InfoBinder's LED represents a unique identifier.

An InfoBinder works in two modes: It can be used like a conventional pointing device, such as a mouse or a trackball, to manipulate digital objects. Note that an InfoBinder's pointing functionality is limited when it is attached to a physical object. After an application has been closed, the associated InfoBinder changes from pointing mode to information binding mode where it represents a physical handle for the associated application. By double-clicking an InfoBinder, the linked application is started and the InfoBinder immediately switches to pointing mode for controlling the application.

In the *PaperLink* prototype [7] a VideoPen consisting of a highlighter pen combined with a small camera is used to augment paper documents with electronic features. PaperLink uses computer vision and pattern recognition techniques to detect specific printed words on a page and to link them to some digital content. The main focus is on supporting the definition of links from paper to digital information or paper-based commands. Optical character recognition further can be applied to use printed information as input for digital services. For example, a recognised word can build the input for a language translation service. However, PaperLink focuses on supporting the definition of links from paper

to existing digital content rather than pen-based capture of written information. The associated digital content has to be chosen in an application-specific way, for example by selecting a file from a file chooser menu.

Physical bookmarks have also been investigated in Hewlett-Packard's *Cooltown* project [82, 143]. The main idea is that any physical object, including paper documents, has a web presence. By reading a tag attached to a physical object, a user gets access to the object's digital web presence. Infrared beacons, barcodes and iButtons are used to uniquely identify people, places and things. Similar to Hewlett-Packard's Cooltown project, virtual counterparts are associated with printed material in the *Entry Points* project [153]. Barcodes encoding a unique identifier are attached to paper documents and other physical objects located in a university campus. The unique identifiers can be read by Pocket Computers or mobile phones equipped with a barcode reader and provide an entry point for accessing available digital information related to the university campus.

The augmentation of photographs with supplementary digital information is the main goal of the *Pulp Computing* project [83] carried out at HP Labs. There, a user can define hot-spots on a printed photograph and associate them with arbitrary web resources. The information bound to these hot-spots can be recalled in a later session by pointing to the corresponding parts of the photo with a Seiko InkLink pen as shown in Figure 2.20(a) (courtesy of Tim Kindberg). The pen's current position within a photograph is used to retrieve any digital information associated with a specific part of the photo. To interact with a photograph, it first has to be identified by scanning a unique barcode attached to its reverse side. The photo then has to be placed on the working surface and its upper left and lower right corners have to be selected with the pen for calibration.

Real-world objects become augmented in *WorkSPACE* [57] to enhance the working experience in architectural settings. While earlier hypermedia systems mainly focussed on linking various forms of digital information, the goal of WorkSPACE was to organise information in mixed-media environments where physical information coexists with digital data. The physical objects are linked to digital entities by marking them with RFID tags. The digital representation of various real-world objects, including paper documents such as architectural drawings, is organised in three dimensional space based on the *Topos* spatial hypermedia infrastructure [58]. Based on the observation that people often classify and group physical objects in collections, a set of abstract actions

(a) Active photos (b) metaDESK

Figure 2.20: Physical bookmarks

based on collectional artefacts and metadata actions are proposed. Filing cabinets, shelves, paper clips or poster tubes are introduced as examples of collectional artefacts. The collectional actions which are no longer associated with a single physical object but rather with a group of related objects extend the notions of digital hypermedia and enable better mixed-media integration.

The *metaDESK* [175] is an augmented desk system that focuses on the integration of arbitrary physical objects into a digital information space. Physical objects and instruments are used as tangible interfaces by linking them to digital information as shown in Figure 2.20(b) (courtesy of Tangible Media Group, MIT Media Laboratory). *Tangible Geospace* is an application of the metaDESK for interacting with a geographical space. By moving small physical models of real world buildings on the metaDESK, the user can control the orientation and scaling of a city map projected onto the desk.

The *Origami* project [151, 152] applied and further extended Xerox EuroPARC's DigitalDesk. As part of the project, a paper-based user interface to the Web based on interactive paper documents has been implemented. The system provides some import functionality for processing conventional web pages. The 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 digital desk and detected by the over-desk camera, links can be activated by pointing to the specific positions within the paper document. As a result, the corresponding digital web page is projected onto the desk next to the printed document. Many

people print out the content of web pages to read it on paper. By using this DigitalDesk application, web pages can be read on paper while still having easy access to the digital hyperlinks. As part of the Origami project other applications linking paper and digital information, for example *Active Alice* [25], a digitally augmented version of Lewis Carroll's *Alice's Adventures in Wonderland*, have been realised.

2.4 Analysis and Hypothesis

The integration of printed and digital materials has become achievable through a variety of technologies for encoding active links on paper in such a way that special input devices, for example digital pens, can detect and activate these links. The solutions for interactive paper differ not only in terms of functionality (browsability, information capturing etc.), but also mobility and flexibility. While most of the augmented desk solutions offer rich functionality, they are not suitable for mobile applications. On the other hand, technologies appropriate for mobile solutions are often limited in terms of digital output capability. By this we mean that it is, for example, no longer possible to overlay digital and physical information as in the case of a fixed augmented desk solution that projects the output directly onto the desk surface.

While the presented projects have proposed different variants of interactive paper as a means of linking printed and digital media, their focus has been on the input device, paper, printing and other hardware technologies rather than on the data integration and information management aspects. This leads to *isolated solutions* which work only for very specific application domains and hardware setups. The proprietary data format of many of these interactive paper systems makes it difficult or impossible to share information between different applications or run a single application with different hardware technologies.

As a result of the strong focus on specific hardware solutions, the linking and integration mechanisms of existing interactive paper solutions tend to be based on physical rather than logical concepts. The definition of links based on physical concepts makes it difficult to change an application's input device technology, for example if a more advanced technology becomes available, without having to perform new authoring of all link metadata. By choosing the right level of abstraction and defining a set of interfaces for input handling, the same interactive paper solution could be applied for a variety of input devices. Such a general interactive

paper framework would represent an ideal platform to experiment with new hardware technologies avoiding the need to reimplement the whole system for each new device by only providing a specific device driver for each interface.

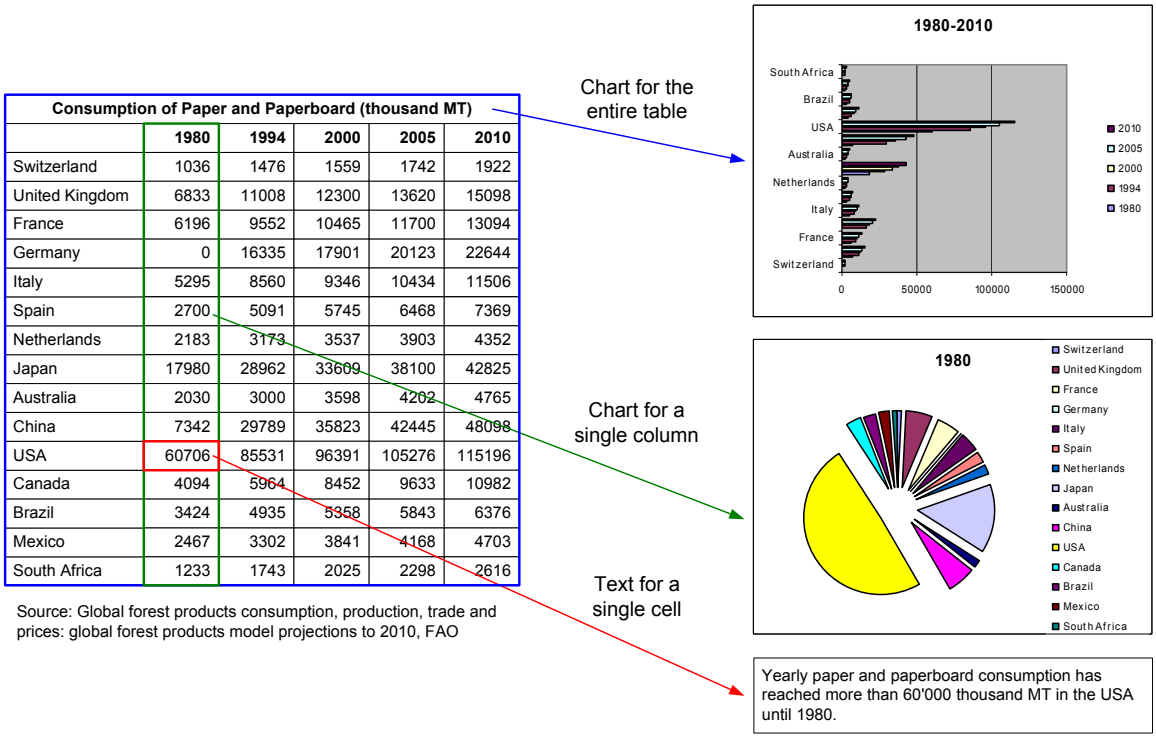
The information management component of the analysed interactive paper systems is often limited to a simple mapping table solution where unique identifiers become associated with specific digital resources. Such a simple mapping table approach may support an enhanced reading process at a certain level. However, based on single identifiers it is not possible to support enhanced writing.

Another drawback of solutions that are based on a simple mapping of identifiers is that they are limited in terms of granularity, i.e. the number and size of objects that can be addressed on a single document page. In the case of the visible barcode approaches, for instance, their use is limited by their visual interference with the printed artwork. The same constraints exist for solutions based on passive RFID tagging since the RFID antenna cannot resolve the exact position of different RFID tags.

To explain what we mean by controlling the granularity of links we introduce the example of a printed table shown in Figure 2.21 where different parts of the table have been annotated with digital content. The bar chart augments the whole table, whereas the pie chart is bound to a specific table column. Last but not least, there is a text note providing detailed information for one specific table entry.

This degree of flexibility in defining links at different granularity levels becomes available only if we no longer associate information directly with single physical tags, for example in the form of barcodes or other technologies for document identification. In fact, a link should be defined in a position dependant manner and the applied hardware technology is only a tool to activate these logically defined links. By clearly separating the link definition from the underlying hardware layer, it becomes possible to use the same link metadata with different hardware solutions.

The foregoing analysis of related work revealed that many interactive paper solutions solely focus on linking paper documents to simple media types such as web pages, images or movies. These links to simple resource types can for example be applied to print interactive URLs in advertisements and other paper brochures which then can be activated by pointing at them with a special input device. However, a powerful architecture for interactive paper must support the integration of paper with semantic content in addition to these links. Instead of only linking resources such as video, text etc. together, resources could consist



	1980	1994	2000	2005	2010
Switzerland	1036	1476	1559	1742	1922
United Kingdom	6833	11008	12300	13620	15098
France	6196	9552	10465	11700	13094
Germany	0	16335	17901	20123	22644
Italy	5295	8560	9346	10434	11506
Spain	2700	5091	5745	6468	7369
Netherlands	2183	3173	3537	3903	4352
Japan	17980	28962	33609	38100	42825
Australia	2030	3000	3598	4202	4765
China	7342	29789	35823	42445	48098
USA	60706	85531	96391	105276	115196
Canada	4094	5964	8452	9633	10982
Brazil	3424	4935	5358	5843	6376
Mexico	2467	3302	3841	4168	4703
South Africa	1233	1743	2025	2298	2616

Figure 2.21: Augmentation of printed tables

of data or metadata objects of a database that describe a particular entity or concept of the application domain. A much richer information model can lead to much richer and more flexible interactive information environments.

A major restriction of most existing interactive paper solutions is that they only deal with links from paper documents to digital information, whereas linking from digital information to paper is not supported in most cases. We believe that the power of interactive paper strongly depends on the functionality of the information management component handling the links between paper and digital media and less on a specific hardware solution. Users should be given the freedom to browse back and forth between the paper and digital worlds, dynamically creating not only annotations, but also links between paper and digital resources, or even between different paper resources. In this way, paper no longer remains a simple input device for accessing supplementary digital information and services only, but becomes a first-class citizen within cross-media information spaces.

We have clear ideas and a vision on how we can bring the field of interactive paper and cross-media information spaces one step further. It is our hypothesis that *the key to a highly integrated interactive paper*

solution lies in the introduction of a platform for general cross-media information management introducing fundamental link concepts in combination with other database functionality such as, for example, querying. The underlying link model should deal with bidirectional links and provide a mechanism to control the discussed granularity of links. For handling data ownership and access rights, a user concept should be introduced in the very core of the link model. We argue that approaching the problem from an information point of view by introducing such a general cross-media information model can take us a step further in realising a next generation interactive paper framework. Such a cross-media information platform could support the seamless integration of paper by supporting all possible types of links between paper and digital content including *paper-to-digital*, *digital-to-paper*, *digital-to-digital* and *paper-to-paper*.

The proposed cross-media information platform can then be successively extended by new types of digital or physical media and therefore leads to a true integration across multiple types of different resources. Such a general cross-media platform should be flexible enough to support any future applications where different kinds of media, including paper documents, have to be handled. The important point of the proposed information-centric approach for cross-media information management is that we do not make any assumptions about specific hardware solutions or resource types to be supported. It is our goal to define the cross-media information model with the appropriate level of abstraction and to show that an implementation of this model is flexible enough to support emerging hardware technologies for interactive paper as well as new media types based on a resource plug-in mechanism.

In addition to integrating semantically rich database objects into the new interactive paper framework, we propose the introduction of *active content* in the form of *active components* as a solution for simplifying the interaction design of interactive paper applications. An active component is a piece of program code that can be linked in the same way as any other resource and that gets executed after the corresponding link has been activated. We believe that the introduction of active content may help to enhance the rapid prototyping of paper-based applications since the programmer can focus on providing very specific application logic by implementing new active components and does not have to care about the more general paper-related issues which are handled by the interactive paper framework. The concept of active content further simplifies the implementation of complex paper-based interaction scenarios. Each modular

active component managed by the interactive paper framework provides a well-defined functionality. Even without any linked multimedia content, tangible user interfaces, integrating paper and digital functionality, can be realised based on such an active component concept.

In the next chapter, we introduce our fundamental concepts for cross-media information spaces. The resulting general cross-media link model provides a resource plug-in mechanism for integrating arbitrary physical or digital resources, including interactive paper.

Simplicity is the ultimate sophistication.

Leonardo da Vinci

3 Cross-Media Link Model

In the previous chapter we analysed existing solutions for interactive paper and argued that the key to achieving a true integration of paper and digital media lies in providing powerful concepts for cross-media information management. In this chapter we will first introduce some more link related terminology by extending the basic link concepts introduced so far and then continue by presenting the general information concepts of our cross-media link model. The link concepts presented in this chapter define the very core of our link model. Therefore, they provide the foundation for any specific implementation of cross-media links presented in succeeding chapters.

It is crucial to have a general and flexible information management component for supporting a variety of interactive paper applications that provide different forms of interaction and may depend on different hardware solutions for integrating paper and digital media. We show that this can be achieved by focussing on fundamental concepts for cross-media information management. General issues such as data ownership, access rights to shared information and link classification are handled by such a cross-media information model and only resource or hardware specific details have to be implemented to support a particular type of media such as interactive paper, movies, web pages etc. By applying a common cross-media information model, data can, not only be shared among different applications, but also be accessed from different hardware technologies.

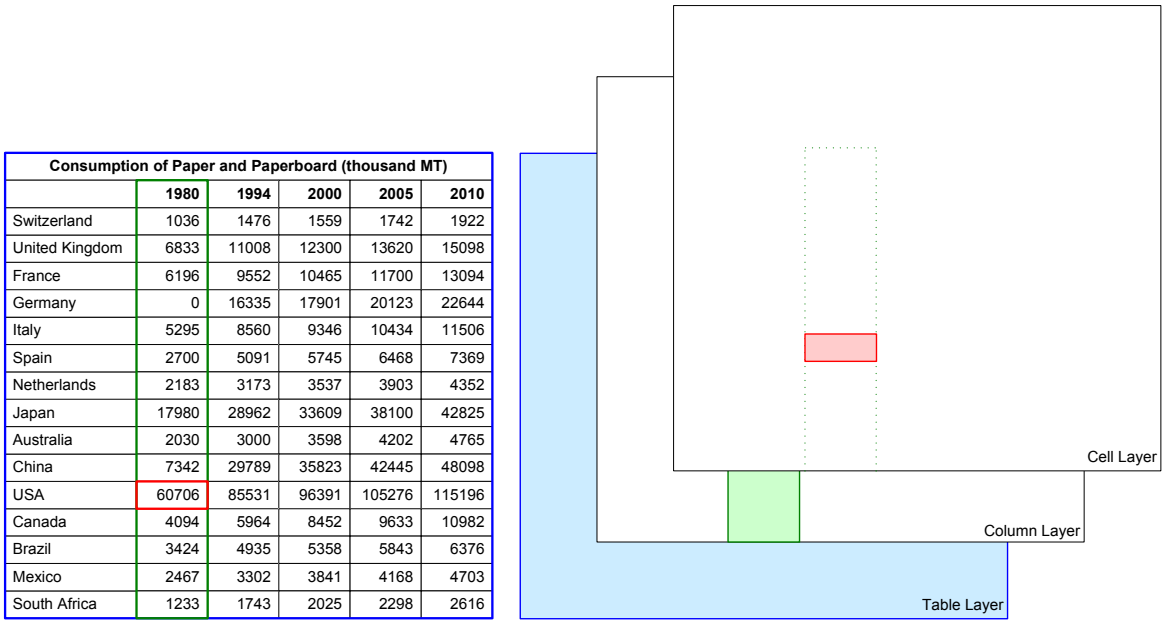
3.1 Terminology

Different possibilities for defining paper-based link source anchors have been presented. We propose a solution where so-called *active areas* can be defined on a sheet of paper and then chosen as link sources or targets. An active area is any arbitrary shape and might correspond to a paragraph, a text box, a single word, an image or only a part of it etc. The selection of a link source is no longer taking place on a physical level, for example by scanning a single barcode, but rather on a logical level. This requires the availability of a mechanism which maps physical locations, expressed in terms of (x,y) coordinates, to logical areas that correspond to semantic document units. Indeed, for selecting a link source at a logical level, the input device delivers information which has to be further processed rather than unique identifiers as in the case of simple mapping approaches. For our approach, where active areas are defined in terms of logical shapes, the only assumption made is that a potential input device delivers positional information about its location on a document page. This information can then be processed to find the relevant active areas that are selected by the input device's current position.

A major advantage of such a logical link resolution is that the links are no longer directly associated with single identifiers printed on the artwork. In fact, if we want to change the definition of an active area, this can be done on the logical level without affecting the artwork of the printed document at all. In addition, the position dependant resolution may provide some contextual information about where a link's source or target is located within a document represented by its position, the page number and a document identifier. Such contextual information is essential if a richer and more flexible interactive information environment has to be designed.

To show other advantages of our position dependant link resolution approach based on active areas of paper let us go back to the example of the annotated table printout presented in Figure 2.21 of the previous chapter. Figure 3.1(a) shows the annotated table whereas Figure 3.1(b) highlights the logical definition of the corresponding link source anchors. As described earlier, link sources can be defined by arbitrary shapes. What happens now if we would like to link the entire table with some digital information and furthermore add an association from a single column to supplementary information? We could define two rectangular shapes, one framing the entire table and the other one surrounding a single column. The problem with such overlapping active regions is that

it is no longer clear what should happen if a user points to a position lying within the intersection of multiple overlapping shapes. On the other hand, if we do not allow for any overlapping shapes, we are somehow limited in the granularity of link sources or targets that can be defined. Therefore, to control the granularity of links we introduce the concept of *layers* imposing an order on the shapes. In the case of overlapping shapes residing on different layers, the layer ordering is applied to determine which link source has to be selected.



Consumption of Paper and Paperboard (thousand MT)					
	1980	1994	2000	2005	2010
Switzerland	1036	1476	1559	1742	1922
United Kingdom	6833	11008	12300	13620	15098
France	6196	9552	10465	11700	13094
Germany	0	16335	17901	20123	22644
Italy	5295	8560	9346	10434	11506
Spain	2700	5091	5745	6468	7369
Netherlands	2183	3173	3537	3903	4352
Japan	17980	28962	33609	38100	42825
Australia	2030	3000	3598	4202	4765
China	7342	29789	35823	42445	48098
USA	60706	85531	96391	105276	115196
Canada	4094	5964	8452	9633	10982
Brazil	3424	4935	5358	5843	6376
Mexico	2467	3302	3841	4168	4703
South Africa	1233	1743	2025	2298	2616

(a) Annotated table (b) Virtual page layers

Figure 3.1: Multiple layers

In our annotated table example, a rectangular active area has been defined on the `Table Layer` linking the entire table with some general information (bar chart). On a second layer, the `Column Layer`, an overlapping shape has been defined annotating a single column with more specific information (pie chart). Finally, a third link source has been defined on the `Cell Layer` annotating a single cell with supplementary textual information in digital form. The concept of multi-layered link sources and targets is very helpful in controlling the link granularity and the level of detail. The presented table is just a simple example but overlapping active areas are quite common in the augmentation of printed documents. Other examples, where a piece of more specific information overlays more general information, are a specific word within a paragraph, text overlapping an image or part of an image and the whole image.

All solutions for interactive paper use some kind of link metadata to define associations between paper and digital information. To discuss further limitations of existing approaches we therefore have to elaborate a bit more on the link concept which was introduced in Figure 2.1 in the background chapter. There, we defined a link L as an association between a source entity S and a target entity T and pointed out that most interactive paper projects are based on such a simple link concept. The same link definition associating a single source element with a single target object is also used on the Web.

We aim for a more flexible link concept where more than two resources can be associated by a single semantic link. For example, a concept described on a printed page could be linked to its digital definition. The same semantic link could then be used by other instantiations of the concept in the same or any other resource. Furthermore, different forms of explaining the concept can be handled by adding multiple resources to a semantic link's set of target entities. Therefore, for effective cross-media information management, handling digital as well as physical resources, we propose a more general link definition where a link L can have multiple source entities $S_1, \ldots, S_m$ (*multi-source*) as well as multiple target entities $T_1, \ldots, T_n$ (*multi-target*) as shown in Figure 3.2.

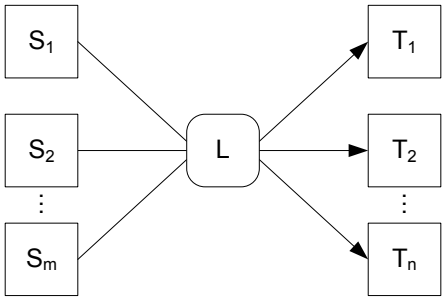


Figure 3.2: Linking multiple source to multiple target entities

In some of the presented interactive paper projects, a user can, not only browse existing links, but also define new associations between related resources. In such a dynamic link authoring approach individual users can link the same source to different resources. Therefore, a source entity S_i or a target entity T_j is not restricted to be associated with one link only. For example, the same source S_i can be used by a link L_j as well as another link L_k. It is even possible that an entity which is a source S of one link at the same time represents the target T of another link. Figure 3.3 provides an example of four links $L_1, \ldots, L_4$ all making use of the multi-source or multi-target features.

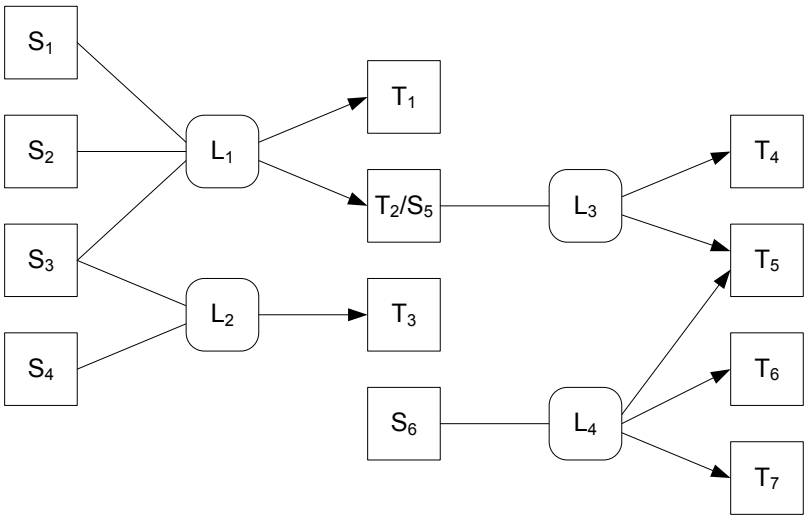


Figure 3.3: Different forms of multi-source and multi-target links

The first link L_1 has multiple sources (S_1, S_2 and S_3) and multiple targets (T_1 and T_2). The selection of S_1 or S_2 will activate link L_1. Since L_1 is a multi-target link both targets can be accessed. Note that it is up to a specific application to decide how multi-target links are visualised. For example, a menu could be presented to the user for selecting a single link target or the application could automatically display multiple link targets in different areas or even on separate output channels. The entity S_3 is not only the source of L_1 but also of L_2. Therefore, a selection of S_3 will activate L_1 as well as L_2. As we mentioned before, an entity can be the source of one link and the target of another at the same time. This is shown with entity T_2 which is the target of link L_1 and at the same time the source S_5 of link L_3. Link L_3 and link L_4 further share a common target T_5.

The introduction of multi-source and multi-target link features enables the definition of more complex link structures not only associating two resources but one or more source entities with one or more target entities. More generally, the application of our new link concept implies that a link can have a combination of digital and physical sources as well as a combination of digital and physical targets. If we come back to the classification of links presented earlier in Table 2.1, the new multi-source and multi-target link concept supports five additional link types which are shown combined with the previously defined ones in Table 3.1.

For the special case of simple links with a single source and a single target, which are used by most existing projects, the link can either be embedded in the document containing the link's source entity or it can

Source S	**Target T**
digital	digital
digital	physical
digital	digital and physical
physical	digital
physical	physical
physical	digital and physical
digital and physical	digital
digital and physical	physical
digital and physical	digital and physical

Table 3.1: Extended cross-media link types

be defined as separate metadata in an external link database. A well-known example of embedded links is the HTML anchor tag which is used to define links in web documents. A first drawback of embedded links is that only persons who have write access to the resource containing the embedded links can also add new link sources or targets. This makes it impossible to build open link systems, where write access to original documents is still restricted but anyone can create new links by adding the corresponding metadata for associating arbitrary resources.

If we stay with HTML documents for a moment, we can spot another link restriction that is also present in most previous interactive paper projects. By selecting an HTML hyperlink, we can access another web resource which has been defined as the hyperlink's target. However, as soon as we follow a link to its target resource it is no longer possible to get information about its link source. This means that, for a given web page, it is impossible to get a list of all web resources linking to that page. However, two resources are often associated by linking them together because the information stored in both resources is somehow related. The over-simplistic *unidirectional link model* introduced by HTML enables only one-way access to this valuable link information. More formally, a unidirectional link is a link where for a given source S_i we can resolve the corresponding targets $T_j, \ldots, T_k$, but for a given link target T_l it is impossible to find the associated sources $S_m, \ldots, S_n$. This means that for a user it is only possible to navigate from a link's source entity to its associated target entities.

We think that one should not restrict a link model to support unidirectional links only. A link should be stored as a *bidirectional link* which, potentially, can be traversed in both directions. It is then up to a specific application of the link model to define if this functionality should be provided at the user interface level or not. For a given bidirectional link, it is not only possible to get the target resources $T_i, \ldots, T_j$ for a source S_k but we can also access source entities $S_l, \ldots, S_m$ for a given target T_n. How can bidirectional links improve the functionality of a framework for interactive paper? By applying such an extended, bidirectional link model it becomes possible to link from digital resources back to the paper documents that they augment. For a given link target in the form of a digital resource, the system can give us any digital or physical resource that is linked to the given target. This can significantly improve the interactivity of a system integrating paper and digital resources, since a user can not only browse from physical documents to digital content, but also back from digital content to physical resources. For example, a piece of digital information that is accessed within a database may provide information about different paper documents it is linked from together with the exact occurrence of the link source anchors within these documents.

The bidirectional link concept alone is not sufficient to enable digital-to-physical links from digital media to paper. To give an example why, in addition to the bidirectional links, we need something more, let us inspect the simple mapping based on unique identifiers once more. Even if we can obtain the physical link source for a given digital link target, this information is useless, since the only information we get is the unique identifier without any information about where it occurs within a document, let alone what semantic concept it represents. Based on a single unique identifier, a user cannot find the relevant part of a document. What is missing is the *context* of a link source, as introduced by our approach, which links on a logical rather than on a physical level. The name of a document together with the page number and the active area defining the link source provides enough information to find the relevant link source in the physical space. Only the combination of bidirectional links and link context information enables links from digital information to paper documents, thereby supporting the physical augmentation of digital information.

While it is simple to include the additional page and document information in our information model, we have to think about what this means in terms of the underlying technologies. The encoding system must be amended to include information about the document and page

number. With current encoding schemes, it would be difficult to include globally unique document identifiers within a grid-based page position encoding scheme. As we show in Chapter 5, one solution consists of including page and coordinate information within the page encoding scheme whereas document identifiers are handled separately by printing special document encodings on the cover page. In addition, other technologies, such as Radio Frequency Identification, could be used for document identification.

While most existing systems focus on the digital annotation of paper documents, a system that aims for tightly integrating paper and digital information should also be dealing with the physical annotation of media by providing links from digital information to paper documents. With the additional contextual information of document and page provided by our approach, we are able to map digital resources to active areas within documents. For example, we could easily find all the places where a given video or term is referenced by a printed word or phrase within a document set. This gives us a basic bidirectional mapping back and forth between printed and digital media. We have to stress that back references to printed documents are not just another piece of text within the database giving, for example, a book title and a page number. All links back to physical paper are generated dynamically based on information about other link sources in printed documents that are bound to a given link target. This meta information is already managed by the information server in the form of bidirectional links and therefore no additional authoring is required to enable links back to physical paper.

Note that the possibility to browse back and forth between printed and digital information and also between different sources of printed information may contribute to avoiding problems of so-called "mode-locking", where a user navigating from paper to digital information gets locked into the digital world and is distracted from the original document.

3.2 OM Data Model

The cross-media link model is defined using the semantic, object-oriented data model OM [124]. OM is a data model that integrates concepts from both the entity relationship and object-oriented data models. The OM model is intended as a basis for efficient data management as well as semantic expressiveness, and a family of object-oriented database platforms have been realised based on this model. It supports information

modelling through a two-level structure of classification and typing, dealing with these on separate layers. *Typing* deals with representation and entities are represented by objects with attributes, methods and triggers defined for the corresponding object types. *Classification* deals with semantic roles and a particular classification is represented by a named *collection* of objects with a specified member type. The member type of a collection not only constrains its membership, but also specifies a contextual view of objects accessed through that collection. Thus, we can have role-dependent object views and behaviours by associating different types with collections according to the roles that they represent. Note that in OM and its associated systems there is support for collections with different behaviours like sets, bags, rankings and sequences.

In addition, OM provides a high-level *association* construct which enables associations between entities to be classified and manipulated directly. The association construct further supports the decoupling of information objects since objects tend to be referenced through the association concept, rather than through object reference attributes. Last but not least, cardinality and other OM constraints can be defined over collections and associations resulting in enhanced semantic expressiveness — and this has proven to be a key factor in the design of our iServer model.

Multiple instantiation is supported by all OM implementations to enable *role modelling*. An OM database object can not only have multiple instances of different types, but it can even dynamically gain or lose specific types as part of object evolution.

The OM model differs from commonly used data models such as the ones provided by the Unified Modeling Language (UML) in that it is, not only intended for system design, but also as an operational model for data management. Thus the OM model defines a full operational model over objects, collections and associations as well as constructs for their definition. The expressive features of the OM model enable us to capture the semantics of application domains in terms of a simple, but powerful set of constructs. Its support for the direct representation and manipulation of associations is particularly useful in supporting link management in systems that provide hypermedia functionality such as our cross-media information platform.

A number of systems have been developed based on the model. These include the OMS Java data management system [87, 88] on which the presented cross-media link server, and therefore also the interactive paper framework, rely. In the development of the cross-media link model we

could, not only profit from the semantic expressiveness of the OM model, but also benefit from the seamless transition from the design of the link model to the corresponding implementation in OMS Java.

3.3 Information Concepts

After presenting the main features of OM, we now introduce the general concepts of our cross-media link model which represents the foundation of the cross-media platform and its applications. In this section, the focus is clearly on modelling issues whereas implementation-specific details of the cross-media information platform are discussed in the next chapter. Note that the information models presented in this thesis are based on the OM notation. We introduce some basic OM notation while presenting the link model and refer the interested reader to the work of Norrie [124] for further details about the model.

3.3.1 Links

The core link model is shown in Figure 3.4. The shaded rectangular shapes denote *collections of objects* (classification) where the name of the collection is given in the unshaded part and the name of the associated type in the shaded part. The type serves both as a constraint on membership in the collection and also as the default view of objects accessed through that collection. Thus, links are represented by objects of type `link` grouped into the `Links` collection. The shaded oval shapes represent *associations between entities* of two collections which can be restricted by cardinality constraints as described later.

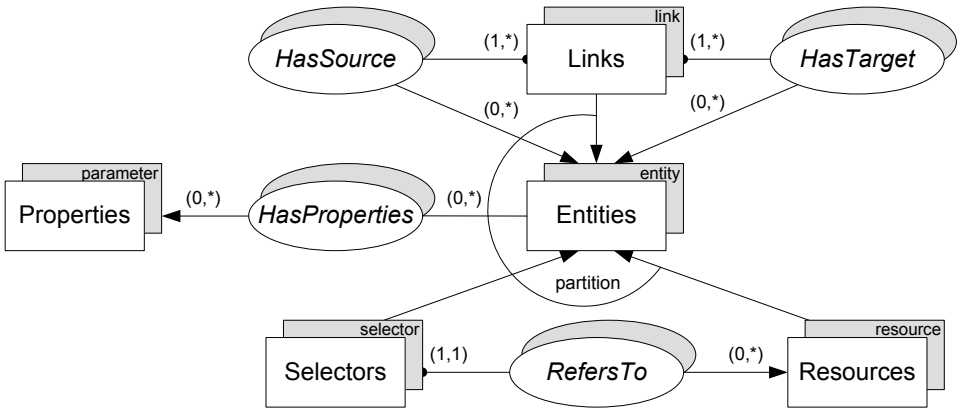


Figure 3.4: Links

Links within the iServer model are always directed and lead from one or more *sources* to one or more *targets*. A source may be either an entire document or a selected element within a document, commonly referred to as an anchor. An information entity can be used equally as a link source or as a link target. This is taken into account in the information model by introducing a generic notion of `entity` and making the collection `Entities` the target collection of both the `HasSource` and `HasTarget` associations. The cardinality constraints specified at the source and target points of the associations indicate the possible level of participation of individual objects. Thus `(1,*)` at the source point of both associations indicates that each link must have at least one and possibly many sources and targets. In this way, we achieve our objective of supporting not only multi-target links but also links with multiple sources. The `(0,*)` at the target point of the `HasSource` and `HasTarget` associations specifies that there is no limit on the number of links for which an entity may be the source or target.

The simplest type of entity is the `resource` type representing an entire information unit. A hypertext author normally wants to control the link granularity by being able to address specific parts of a document rather than the document in its entirety. Therefore, as a second subtype of the entity type we provide the concept of a `selector`, a construct enabling parts of the related resource to be addressed. These specialisations of `Entities` are reflected in the model by the subcollections `Selectors` and `Resources`. An association `RefersTo` represents the fact that a selector is always associated with exactly one resource, whereas each resource can have more than one referencing selector. The abstract `resource` and `selector` concepts have to be extended to support concrete types of media as described in Section 4.1.

Furthermore, by modelling `Links` as another subcollection of `Entities`, we gain the flexibility to create links whose sources or targets are defined by other links, thereby annotating any link with supplemental information. An important point to mention is that a link can not only be annotated with textual information but with any arbitrary entity. This means that we can use resources, parts of resources or even other links to annotate a link. For example, we could have a web page with links to additional information and these links could then be annotated by other users with textual comments or links to different web resources etc. A `partition` constraint is specified over these subcollections to denote that each entity belongs to exactly one of the three categories `Links`, `Selectors` and `Resources`.

Additionally, every entity can be associated with a set of properties which are stored as a set of string tuples in an entity's property attribute. These properties, represented by (key,value) pairs, are not predefined by the iServer framework. They can be defined individually to customise an entity's behaviour for specific application domains. For example one could define a link property `onActivate` which would represent the action to be taken when a link is activated. Possible values could be `openInline` to open the link target within the current resource or `openNew` to display the link target in a separate view. Similar concepts also exist in the XML Linking language (XLink) [35, 41] where the `actuate` attribute is used to define the traversal behaviour and the `show` attribute defines where a link should be shown, e.g. if the result should be shown in the same or in a new window. However, we try to be as flexible as possible by not predefining a set of properties but rather introducing an abstract property set which can be extended by specific applications.

A flexible typing of links can be achieved by introducing a property with the name `type` and assigning the appropriate values to it. For example, we could introduce a special type for links which represent annotations and then handle them in a specific way. Further, we could also introduce domain- or application-specific subcollections of `Links` as a means of classifying links. This combination of being able to associate properties to links and also classifying them provides a very flexible and powerful means of representing link taxonomies.

Note that since the underlying OM model provides bidirectional associations as a higher-level construct, all the associations used within the cross-media link model are also bidirectional. This enables us to, not only get all the link targets for a specific link source, but also to find the corresponding link sources given a specific target object.

3.3.2 Layers

Most existing link services either do not allow nested link anchors at all or, in cases where it is possible to define nested anchors, they lack the possibility of defining the behaviour of the nested links. Here is a simple example showing exactly what we mean by nested links. Assume that we have a link from a paragraph of a document to a target object which contains a user's critical comment about this paragraph. In addition, we want to define links from various individual words within the same paragraph to their textual explanations. What happens if we select a specific word with its associated link? In some hypermedia systems, we

would only get the link associated with the corresponding word, whereas other systems might return the link target of the word as well as the paragraph's link target. Some hypertext standards, such as for example the HTML format, do not support overlapping link sources at all. To become more flexible in defining the semantics of nested link source and target anchors, we introduce the concept of layers [159]. Figure 3.5 shows an annotated page containing multiple layers with active regions.

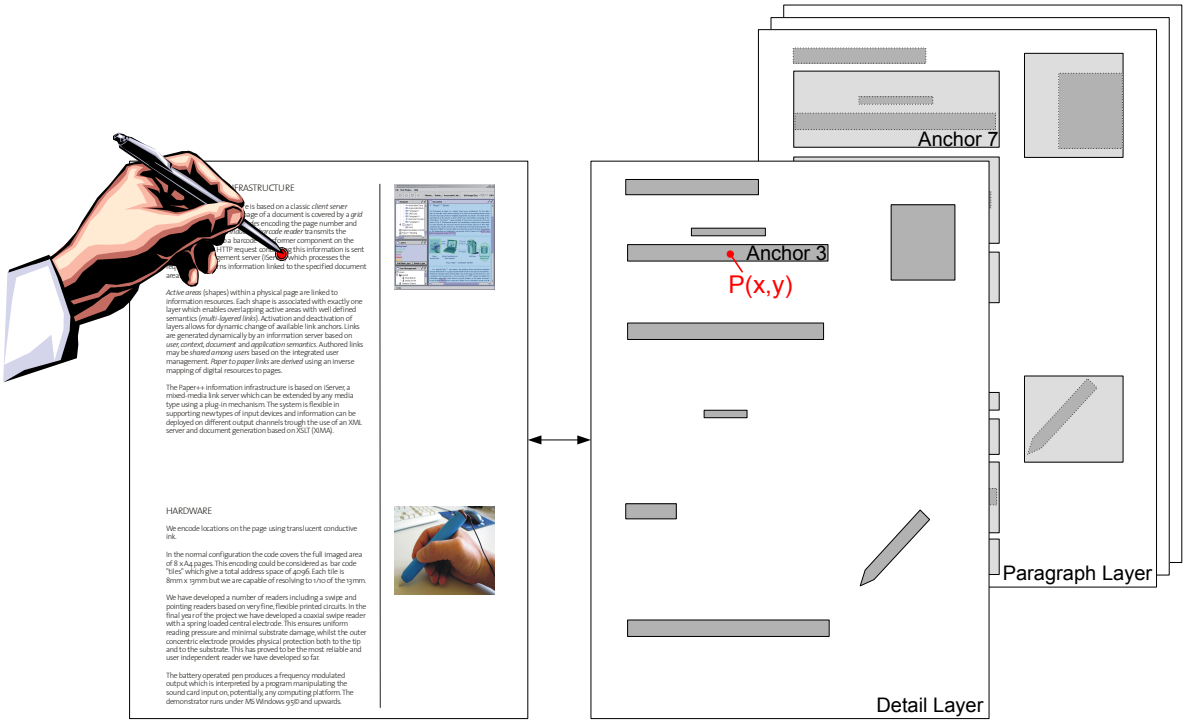


Figure 3.5: Virtual page layers

On the left hand side of Figure 3.5 we see the physical page whereas the right hand side shows the page's representation within the interaction database. The current position `P(x,y)` lies within the rectangular shape defined by `Anchor 3` on the `Detail Layer` as well as in the shape on the `Paragraph Layer` which is defined by `Anchor 7`. In the case of any ambiguity, the shape on the topmost layer always has the highest priority. Therefore, the link bound to `Anchor 3` will be chosen.

A representation of layers within our cross-media link model is provided in Figure 3.6. Each selector is associated with exactly one layer and we do not allow overlapping selectors on the same layer thereby forcing overlapping links to be defined on separate layers. In the case that a concrete selection would return several links by activating multiple overlapping selectors, by definition, the link bound to the selector on the uppermost layer will be selected. The notation shown in Figure 3.6

indicates that there is an explicit ordering of the layers, i.e. |`Layers`| is a totally ordered collection (ranking). In addition, specific layers may be activated, deactivated and dynamically reordered enabling us to generate context-dependent links by resolving a particular selection to different selectors depending on the current set of active layers.

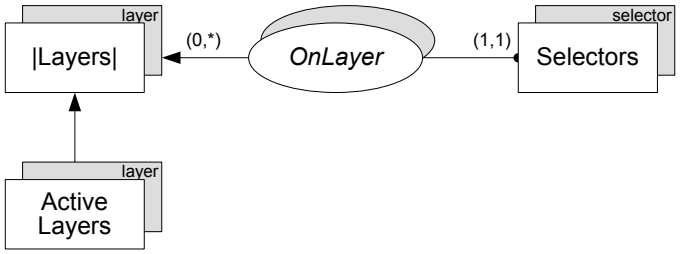


Figure 3.6: Layers

The association `OnLayer` relates each member of `Selectors` to exactly one member of collection |`Layers`|. More formally, assume that there are k layers numbered $1, 2, \ldots, k$, and that each selector S_i is associated with exactly one specific layer i.e. $\text{layer}(S_i) = j$ where $j \in \{1, \ldots, k\}$.

For a given selection q, we write $q \in S_i$ to denote that the selection q, which could for example be an (x,y) position within an active area, activates the selector S_i. If two selectors S_i and S_j are overlapping, i.e. there exists some selection q such that $q \in S_i$ and $q \in S_j$, then the selectors must be on different layers i.e. $\text{layer}(S_i) \neq \text{layer}(S_j)$.

At any point in time, the set of active layers is a subset of all layers. Let $\mathscr{A} \subseteq \{1, 2, \ldots, k\}$ denote the index set of all active layers.

For a set of selectors $\mathscr{S} = \{S_1, S_2, \ldots, S_n\}$, the set of potentially activated selectors associated with a given selection q is given by the set

$$\mathscr{S}(q) = \{S_i | q \in S_i\}.$$

The finally activated selector $S[q]$ is given by

$$S[q] = \arg \max_{\substack{S_i \in \mathscr{S}(q) \\ \text{layer}(S_i) \in \mathscr{A}}} \text{layer}(S_i).$$

OM is object-oriented and has a full algebra over objects, collections and associations. Without going into the details of the OM query language AQL, below we show a query that, given a selection q, will return the uppermost shape of the active layer set that is activated by q.

```
last(domain((OnLayer rr ActiveLayers)
       dr (all S in Selectors having S.activatedBy(q))
     )
)
```

First, two selections are applied to the association `OnLayer`: we select pairs where the second element belongs to `ActiveLayers` and, in a second step, we further select those pairs whose first element is a member of `Selectors` and whose method `activatedBy` returns `true` for the input parameter q. Having carried out this reduction of the `OnLayer` association, we then take the `domain` which gives the collection of selectors which are activated by the selection q and belong to an active layer. Since `OnLayer` is a total ordering that respects the ordering of `Layers`, the `domain` operation will maintain that ordering and the `last` operator will select the uppermost selector as required. Details of the OM algebra and query language are given in [190].

The ordering of the layers does not have to be static and may be changed dynamically, if appropriate, by the application. A potential application of the layering concept is to support a "zoom-in" functionality on any resource by switching the active layer set as a result of repeatedly providing the same selection q. A similar concept is described in [18], where the *Magic Lens* is used as some kind of information filter.

3.3.3 User Management

In order to support both personalisation and the sharing of link knowledge, we need a notion of *data ownership* combined with different levels of access rights. Since we consider user management to be a fundamental requirement for ubiquitous hypermedia, we provide the necessary functionality as one of the link model's fundamental components and implement it in the core layer, as presented in Figure 3.7. This contrasts with many systems that introduce user management on the application level only.

A `user` can either be an `individual` or a `group`. Users can be classified in different groups represented by the collection `Groups`, where a `group` itself can be part of other groups. Each `entity` is created by exactly one individual user, its owner. The owner has full control over an entity's content and can define access rights for other groups of users or individuals. The two associations `AccessibleTo` and `InaccessibleTo` are introduced to define access rights in a flexible way. The set of individuals having access to a specific entity is defined by the groups and

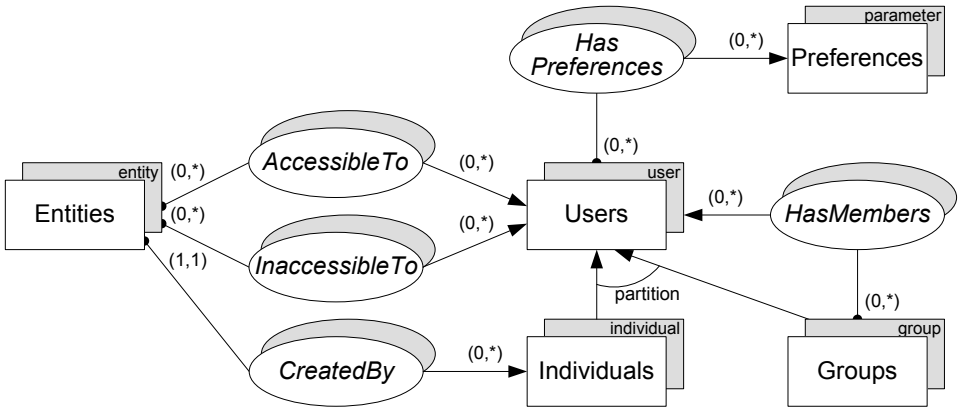


Figure 3.7: User management

individuals associated by `AccessibleTo` minus the groups and individuals defined through the `InaccessibleTo` association. In addition, there exists a constraint that access rights defined for an individual always have priority over access rights defined for a group. More formally, a group $\mathscr{G}$ is defined by some subgroups $\mathscr{G}_i$ and some individuals I_k that it contains, i.e. $\mathscr{G} = \{\mathscr{G}_1, \ldots, \mathscr{G}_m, I_1, \ldots, I_n\}$. The expanded set of individuals who are members of a group is given by

$$\mathscr{E}(\mathscr{G}) = \bigcup_{g \in \mathscr{G}} \mathscr{E}(g) \quad \text{with} \quad \mathscr{E}(I) \equiv \{I\}.$$

For a specific entity e, let $\mathscr{G}_a(e)$ denote the set of groups explicitly specified as having access to e and $\mathscr{G}_x(e)$ those explicitly denied access. Correspondingly, let $\mathscr{I}_a(e)$ denote the set of individuals explicitly specified as having access to e and $\mathscr{I}_x(e)$ those explicitly denied access. Then $\mathscr{A}(e)$, the set of individuals having access to entity e is defined as

$$\mathscr{A}(e) = \mathscr{I}_a(e) \cup (\mathscr{E}(\mathscr{G}_a(e)) \setminus \mathscr{E}(\mathscr{G}_x(e)) \setminus \mathscr{I}_x(e)).$$

This allows us to define complex access rights for an individual entity of the form "the entity should be visible to everybody except one specific group of users and two particular individuals". By defining the access rights at the entity level, we can define individual permissions for links, resources and selectors. We can not only define different link sources for different users, but also bind different content to the same link source anchor depending on a user's current role.

The activation of a link may depend on the user and even the user role. An author of a cross-media application based on our link model may, not only define different selectors for different users, but also link the same selector to different information resources based on the user profile. We

may want to present different materials to different users according to preference or role. For example, in the case of an encyclopaedia, we may want to present different links and content depending on the age of the user. Typically, children prefer less detailed textual information and more images and audio. For educational scenarios, we may want to present different information according to whether the user is an instructor or a student. In particular, links from an exercise sheet may link to hints in the case of a student and detailed solutions in the case of an instructor.

Note that, in the current implementation, an individual can only be given two kinds of access: no access or full access. The two levels of access have been sufficient for all the applications that we have implemented so far. However, we could always extend the model with an additional access rights component to define different levels of access if necessary.

While in this chapter we have introduced the different components of the cross-media link model, a detailed schema definition of the complete model is provided in Appendix A.

3.4 Summary

We have extended the simple link model that was introduced in Section 2.1 and presented a general cross-media link model where links can have multiple source entities as well as multiple target entities. Three different types of entities, the resources, selectors and links, have been introduced. Thereby the abstract selector construct can be used to address parts of a resource and define link sources or targets.

The concept of multi-layered links has been introduced as a tool for controlling link granularity. The layers further define the semantics of overlapping selectors by putting an order on them. Dynamic activation and deactivation of layers has been discussed as another mechanism for controlling link activation.

Finally, the issues of data ownership have been discussed and we presented the user concept which forms part of the very core of our link model. We have emphasised the importance of access right control in collaborative environments where links are shared within a community of users.

Our data-centric approach led naturally to a very general and flexible link model that not only supports various concepts of current hypermedia systems, but also caters for cross-media linking and extensibility for new resource types.

When data of any sort are placed in storage, they are filed alphabetically or numerically, and information is found (when it is) by tracing it down from subclass to subclass. It can be in only one place, unless duplicates are used; one has to have rules as to which path will locate it, and the rules are cumbersome. ... The human mind does not work that way. It operates by association.

Vannevar Bush

4 iServer Platform

The implementation of the general link concepts presented in the previous chapter, based on a semantic data model and object-oriented database technologies, led to iServer, a cross-media information server. The iServer platform is a very general cross-media information management component enabling links to a wide variety of resources [160].

In this chapter we describe resource plug-ins that have been implemented for the iServer framework, including the plug-in for interactive paper, and describe the Java programming API available for applications built on top of the iServer architecture. In addition to the Java interface, iServer provides an XML interface together with a Web Service in order to be independent of any particular programming language.

An information platform for cross-media information management should be flexible in providing *context dependent content delivery* based on the type of the output device and other factors. Therefore, it was necessary to build a multi-channel user interface which can easily be adapted to new requirements in terms of the desired output format. All information stored in iServer becomes accessible from web browsers, WAP-enabled mobile phones and even regular telephones by applying the eXtensible Information Management Architecture (XIMA).

Different types of semantically rich digital information, provided in the form of database objects and active content, are introduced as a tool to build applications with rich database content and complex interaction design. Furthermore, a distributed iServer version for collaborative

information sharing and filtering based on peer-to-peer (P2P) technologies is presented.

4.1 Resource Plug-ins

The iServer architecture is designed as an extensible Java-based cross-media information management platform which can support any type of digital or physical media. The iServer core component is an extension of the OMS Java data management system with some general cross-media concepts for linking, layer and user management. A resource plug-in mechanism based on the `resource` and `selector` concepts enables the integration of new resource types by providing a resource-specific Java implementation of a resource and its corresponding selector.

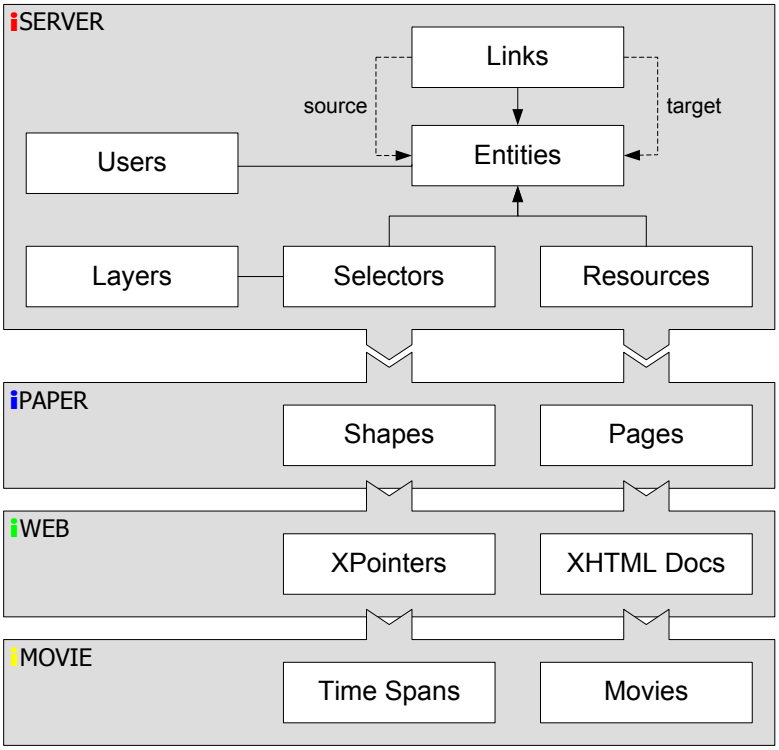


Figure 4.1: Different resource plug-ins

In Figure 4.1 we show a simplified version of the iServer link model containing only the main concepts such as links, selectors and resources. The `Selectors` representing elements within a resource in combination with the `Resources` which represent entire resources are the central components for the resource plug-in mechanism. For a particular media type, we can extend the iServer platform by introducing a component that

defines selectors and resources for that media type. Figure 4.1 shows a component for movies (iMovie) where a resource is a movie and a selector may be a time span. Another plug-in is responsible for dealing with web pages (iWeb) where a resource is an XHTML document and a selector is an XPointer expression.

A major advantage of the chosen plug-in mechanism is the tight integration over all types of media based on a common core link model rather than isolated applications for specific kinds of media. As soon as a plug-in for a new resource type has been implemented, entities of that medium can be cross-linked with instances of any existing media type. With each resource plug-in introduced, the iServer platform becomes more powerful and provides a richer cross-media information space. In Table 4.1, we provide a list of existing iServer resource plug-ins represented by the corresponding resource and selector types for each type of media. Note that the list of plug-ins is far from being complete since arbitrary digital or physical resources could be added.

Medium	**Resource**	**Selector**
paper	document page	shape
web page	XHTML document	XPointer
movie	mpeg file, avi file etc.	time span
movie	mpeg file, avi file etc.	shape
sound	mp3 file, wav file etc.	time span
image	gif file, jpeg file etc.	shape
database	database workspace	query
physical object	RFID space	RFID tag

Table 4.1: Potential iServer plug-ins

For a given resource type there may be varying types of selectors based on the requirements of specific applications. In Table 4.1, we suggest time spans to be used as a candidate for movie selectors. However, a specific application might need to link movies based on spatial information within the movie whereas another application might need to define links based on a combination of temporal and spatial information. The iServer architecture therefore supports the definition of different selectors for a single resource type.

A first iServer plug-in was implemented for interactive paper to integrate paper and digital content. More recently we developed three additional iServer plug-ins to fully integrate XHTML content, movies and arbitrary RFID-tagged physical objects. Of course, XHTML documents and movies already have some basic support in the core iServer implementation as it is possible to link for example from a web page to a movie. However, to *fully* support these resource types it should be possible to link to/from elements within a resource and not just associate entire resources. This implies that we have to implement a resource plug-in with the corresponding selector for addressing parts of a resource. We present the realised iServer plug-ins in more detail before describing various other implementation issues of the iServer architecture.

4.1.1 Interactive Paper

The base unit for linking paper documents is a single page. Therefore, in the case of the interactive paper plug-in (iPaper), a resource is represented by a single page and a selector is an active area defined by a shape within that page. The iPaper plug-in supports simple links between printed and digital materials as well as highly-interactive applications where users can easily move back and forth between the printed and digital worlds.

Since we are not interested in capturing and digitising physical objects, but rather in augmenting them with supplementary information, we do not have to store a digital version of a document's content within the iServer framework. We only need to store some metadata about a printed document, such as its dimension and the number of associated pages, in the form of a digital document model. The digital document model allows us to define selectors on documents which can then be used to link from physical documents to digital content, from any digital information to physical content or to build links between physical documents. For that purpose we need the concepts of documents and pages providing us with meta information about specific documents. The `document` type has been modelled as a container which is able to contain zero or more pages. Each document has a unique document identifier which is used for the mapping from a physical document to its digital document model instance. The only information provided by a specific page is its page number and dimension. By modelling documents and pages as two separate concepts, in a first step, we can use either a whole document or a specific document page as a source or target anchor of a specific link.

If one thinks of how we work with paper documents and organise them, often we do not annotate or link entire documents or pages but rather mark specific parts of a page with a highlighter pen or write marginal notes. Furthermore, section headings and document indices are often used as the basis for providing links within paper documents. Therefore, it is evident that we need to support a means of addressing specific parts of a page. The concept of the selectors provided by the iServer link model is exactly what is required to address specific parts of a resource.

The introduction of geometrical shapes as a specific form of selector for interactive paper enables us to define active areas on a page. Each time a user points to a position within an active region, its associated links will be activated. Figure 4.2 shows a UML model of the iPaper plug-in with the specification of the specialised resource and the corresponding selector. By supporting different forms of shapes such as rectangles, circles and polygons, we are flexible in defining arbitrary page selectors. Further, the concept of composite shapes allows for the grouping of multiple shapes. A composite shape has to be explicitly defined by the link author and is always handled as a single shape by the system. Therefore, if a user points to one of a composite shape's subshapes, the composite shape will always be selected.

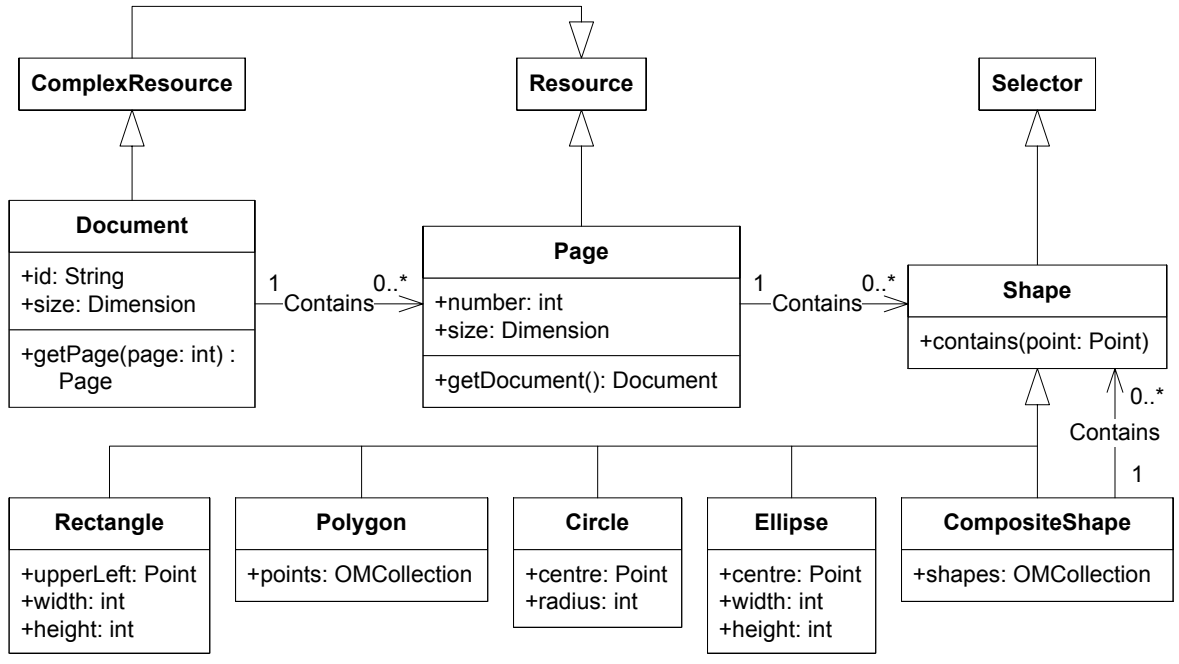


Figure 4.2: iPaper plug-in

In the case of interactive paper, the `page` type becomes a specific implementation (subtype) of the more general `resource` type whereas the `shape` and its subtypes are resource-specific implementations of the `selector` concept. The definition of a page selector implies that an input device for the interactive paper framework provides a document identifier and a page number as well as the absolute (x,y) position within a page. This input data is then used to send a request to the iPaper plug-in which has to resolve the selected shape.

After a request has been sent to the iPaper plug-in, we first fetch the relevant document based on the unique document identifier. Using the page number information delivered by the input device, we retrieve the corresponding page object. The only thing to be done now is to check which page selectors (shapes) are selected by the input device's current position. This means that, for each selector defined on the processed page, we have to check whether it contains the current (x,y) position. In the case of multi-layered active areas, the shape positioned on the topmost layer has to be selected.

The information model of a particular interactive paper application, as well as of any other iServer application based on different resource plug-ins, can be separated into a resource model and an application model. The resource model, the one for interactive paper shown in Figure 4.2, is the same for all applications. It manages the meta information which is necessary to return the corresponding source or target anchor for a given input provided by the digital pen. It is important that this be done as generally as possible so that future changes to the forms of paper and pen technologies used will have minimal effect on the database server. As we show later when discussing the remaining parts of the interactive paper framework, we achieved maximal flexibility to support new input devices as well as new information encodings by defining a set of abstract interfaces for input handling. The application model handles application-specific information as well as functionality and therefore has to be designed separately for each new application. It can either be implemented directly as an extension of the iServer database or as a separate application database that is referenced by individual link instances.

4.1.2 Web Pages

Nowadays, the Web is still mainly based on HTML documents and lacks sophisticated link functionality that was developed by the hypertext community a long time ago. A major drawback of the link mechanism used

in HTML documents is that the anchor of a link always has to be embedded in the HTML document. This implies that only the owner of a document can edit or add new links to a web page. The hypertext community has always been aware of this problem and provided special link servers for greater flexibility in associating web content. The link server stores the metadata about the links between different web pages which is then combined with the static HTML content at request time. The goal of the iServer plug-in for web pages (iWeb) was to use iServer as an external link repository for web pages in a similar way to existing link servers. However, in addition to the link servers enabling links between different digital web documents, an iServer-based approach results in a flexible cross-media solution, not only capable of integrating HTML pages, but also other digital or physical resources.

To simplify the definition of the web page plug-in, at the moment we only deal with HTML documents that conform to the W3C's XHTML standard. For HTML pages not conforming to the XHTML standard, we would have to supply alternative selectors based on standard HTML parsers. This notion of selector is similar to that of reference objects in the open hypermedia model FOHM [117].

The definition of a corresponding selector for XHTML documents is straightforward since we can build on work that has already been accomplished in the context of the XML Linking Language (XLink) [41]. XLink uses the XML Pointer Language (XPointer) to address a specific part of an XML document. By using XPointer expressions as XHTML selectors within the iServer framework, we can define any part of an XHTML document as a link source or link target. However, note that we obtain some additional features not available in the XLink language. Let us assume that we want to link, not only a full paragraph of an XHTML document, but also specific words within this paragraph. The layering concept will always provide us with the context in which the current selection has to be resolved. Thereby, we can define multi-layered links while keeping a well-defined semantics for link activation. Of course it is also possible to define overlapping link sources based on the XLink language. However, the XLink standard does not provide any functionality to define the semantics of overlapping link source anchors. It is therefore up to individual applications to define the behaviour and appropriate link activation in the case of multiple overlapping link sources.

Furthermore, the iServer-based web page support is not limited to linking within HTML documents, but enables us to link to any resource type already supported by iServer, such as, for example, interactive

paper. In addition, instead of having to define a separate component for data ownership, the iServer user management component handles cross-media link sharing issues for HTML documents.

In a first prototype, a proxy server approach as shown in Figure 4.3 was applied to integrate the link information managed by iServer into existing web pages [155]. The advantage of such a proxy server approach is that it is completely transparent to the client which means that no special software has to be installed on the client side. After the web browser has been configured to use the special proxy server, which was implemented together with the iWeb plug-in, all subsequent requests are redirected to the proxy server. The proxy fetches the original document from the Web. At the same time, the proxy connects to iServer to get additional metadata for the requested web page in the form of external link information. The original page and the external links are combined by the proxy server based on a Document Object Model (DOM) representation of the web page, before the resulting HTML page, augmented with iServer link information, is sent back to the web client.

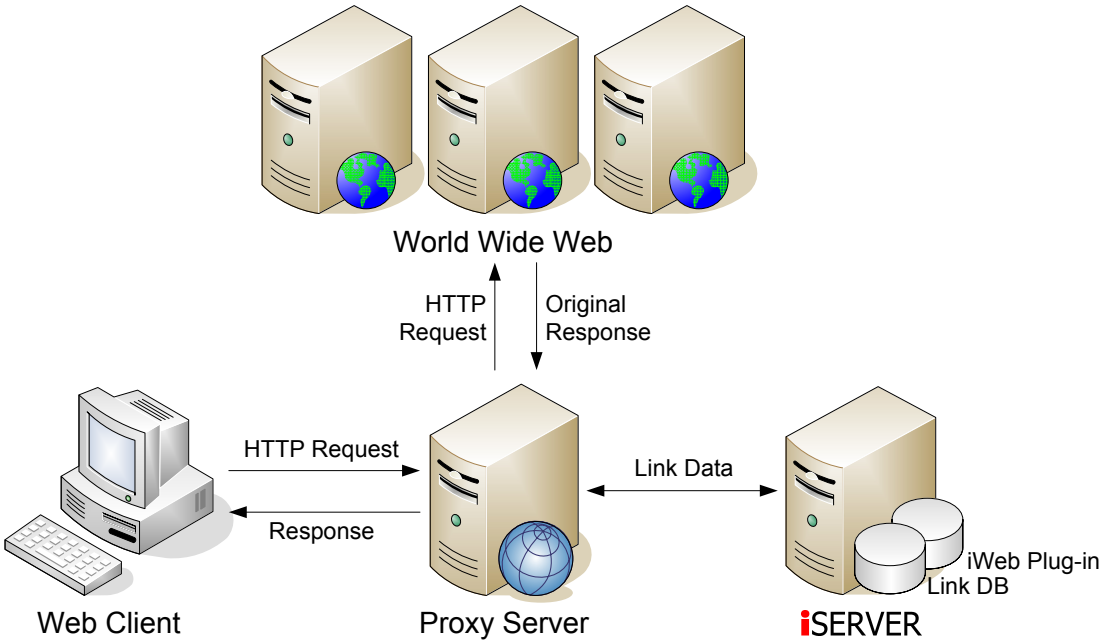


Figure 4.3: Proxy server approach

The authoring and creation of new links is based on two components. The first component is a signed Java applet running in a separate browser window which provides the main user interface and is responsible for all communication with the iServer database. The second component is based on JavaScript functionality that is added to each web page by

the proxy server. It manages the pop-up menus which are presented when a user clicks somewhere within a web page. In addition, JavaScript functionality is also used to detect which parts of a web document have been selected as the source or target anchor of a new link. The JavaScript functions contact the Java applet that communicates with the iServer platform to create a new link for the selected part of the web page.

The proxy-based approach for integrating iServer link metadata into HTML documents also has some drawbacks. First of all, the proxy is an additional component introducing some extra delay in the communication between the client browser and the web server. A page has to be completely restructured and augmented by the proxy server before it is handed over to the client for visualisation. A second disadvantage of this approach is the limited functionality for link visualisation and authoring within the web browser. We therefore implemented an alternative prototype for web pages, based on an extension for Mozilla's Firefox browser, which no longer requires a proxy between a client and a server [108]. A screenshot of the iServer extension for Firefox is shown in Figure 4.4.

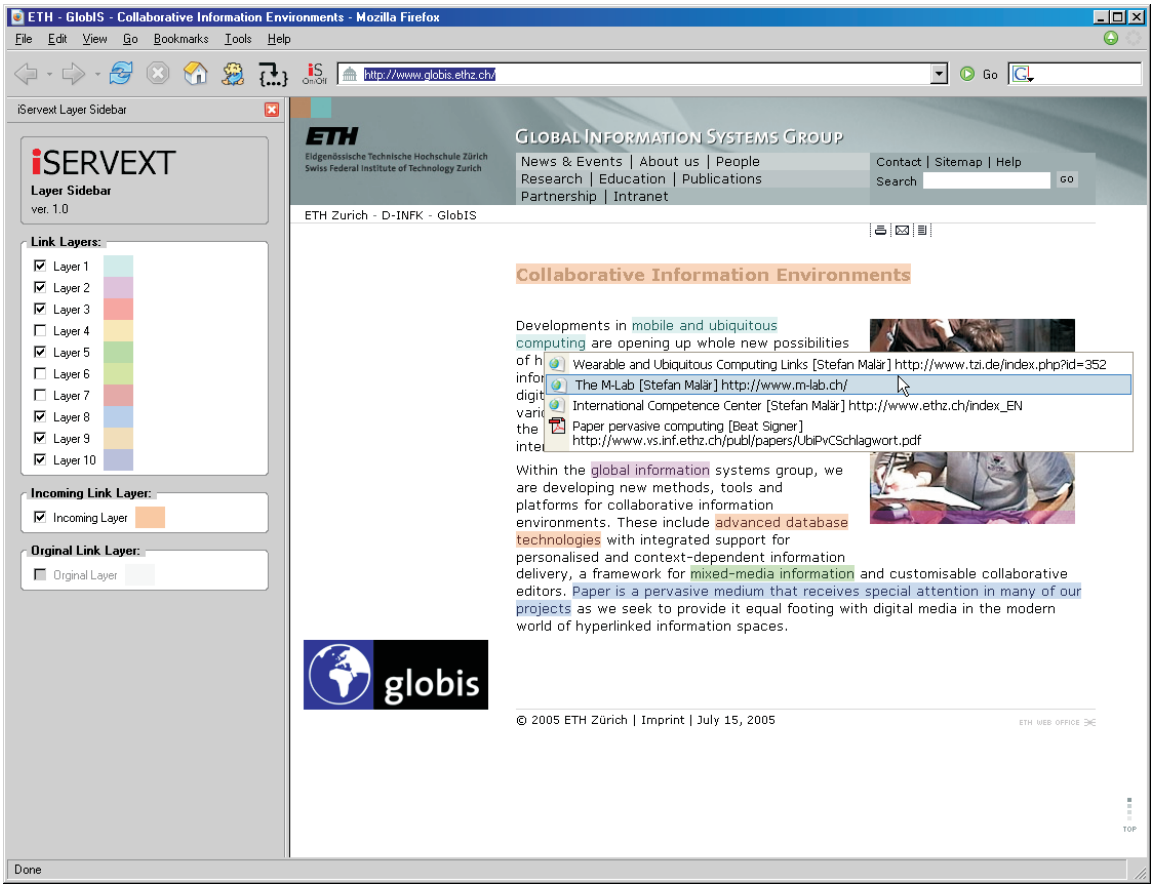


Figure 4.4: iServer extension for Firefox

A first advantage of this browser extension is that we can avoid any extra delays introduced by a proxy-based approach. When a new web page is requested by the browser, it is first downloaded from the server and immediately visualised by the browser. In a second step, the browser extension starts to parse the web page and augment it with supplemental link information that is acquired by contacting iServer. As soon as the final web page has been rendered by the browser extension, the page gets redisplayed in the browser's main window as shown in Figure 4.4.

Another advantage of the browser extension is the tight integration with the browser's visual user interface. There are many more possibilities of using browser-specific visual components such as sidebars etc. and to directly integrate iServer functionality into the browser's menu structure. A larger part of the user interface was implemented based on the XML User Interface Language (XUL). The communication with iServer was established based on contacting the iServer Web Service interface, which is described later in this chapter, based on JavaScript functionality.

Of course the tight browser integration obtained by implementing a Mozilla Firefox extension has not only advantages but also some drawbacks. The main problem is that the iWeb plug-in can no longer be accessed by any web browser. For each browser family, such as Mozilla's Firefox, Microsoft's Internet Explorer or Apple's Safari browser a customised iWeb extension has to be implemented.

4.1.3 Movies, Sounds and Still Images

An iServer plug-in for movies, sounds and still images has been implemented based on a QuickTime development kit provided by Apple [95]. The system level QuickTime software package provides the required functionality to control time-based data based on a set of Java wrapper classes.

Three different types of selectors have been defined for the three different resource types. *Spatial selectors* may be applied to address parts of an image as shown in Table 4.1. Portions of a sound file can be selected by *temporal selectors*. For a movie, which is a combination of moving images and a sound, we can use either spatial or temporal selectors. In addition, a combination of spatial and temporal selectors can be applied to define a link source or target for a movie. In Figure 4.5 we show a prototype of the QuickTime-based movie plug-in. In the upper left corner, there is a window for replaying a movie combined with the usual controls for play, pause, fast forward etc.

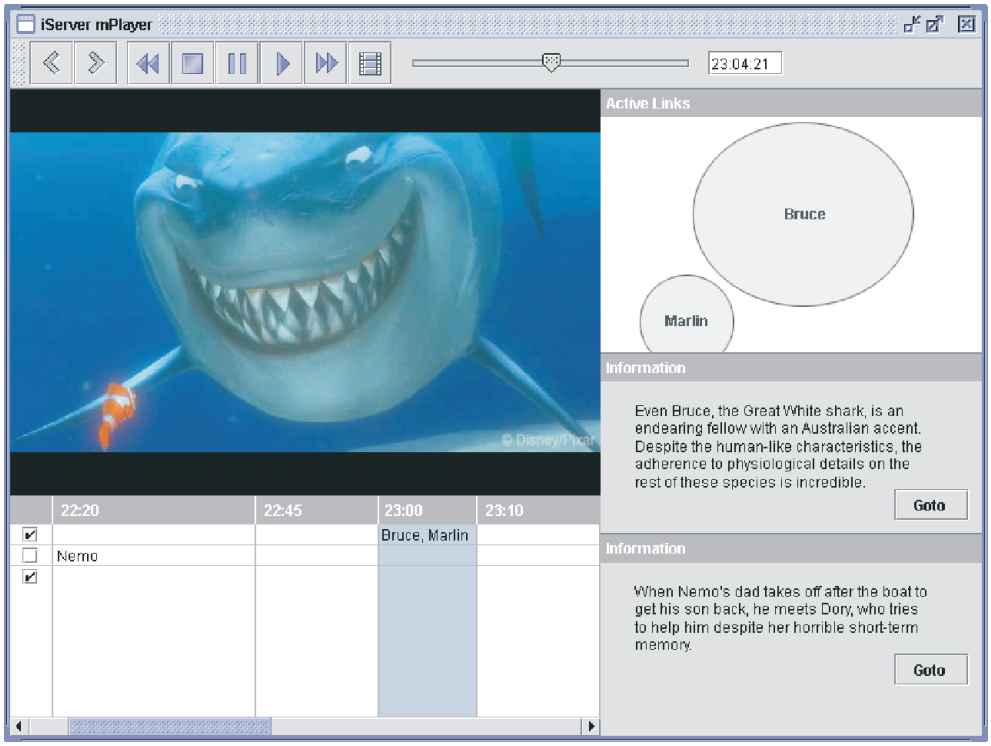


Figure 4.5: iMovie plug-in

In addition to the main replay window, information about link data is provided. Below the replay window there is a timeline for defining temporal links. In the example shown in Figure 4.5, the timeline is further separated into three different layers. On the first layer a temporal link named 'Bruce, Marlin' has been defined which starts at '23:00' and ends at '23:10'. On the right hand side of the replay window there is information about the currently active regions within the movie, i.e. the spatial selectors that are currently active. A link can be activated by clicking with the mouse at a specific position within the replay window. Furthermore, some textual meta information of active links is shown in the lower right corner. By clicking on a 'Goto' button, the corresponding link target is shown. Note that another possibility for defining link anchors would be the use of text-based selectors on a movie's subtitles.

4.1.4 RFID Tags

A main feature of the iServer link model is the distinction between a resource and the corresponding selector providing the flexibility to address a specific part of a resource. Nevertheless, it is also possible to support simpler applications where the entire resource is always selected. The

augmentation of arbitrary physical objects with digital information is one possible application of such a simplified link definition. For example, RFID tags are often used for non-visual object identification in ubiquitous computing.

We have extended the iServer platform with a resource plug-in for object identification based on RFID tags. This turned out to be a simple undertaking since we only had to model the RFID tags as a specialisation of the selector which enables us to address parts of an RFID address space. This is illustrated in Figure 4.6 where the only two classes that had to be implemented are `RFIDSpace` and `RFIDTag`. Each `RFIDTag` stores a unique identifier. At request time, iServer checks which RFIDTag is selected by a given input identifier and activates all associated links. Note that there exist various RFID standards, all with different address space ranges.

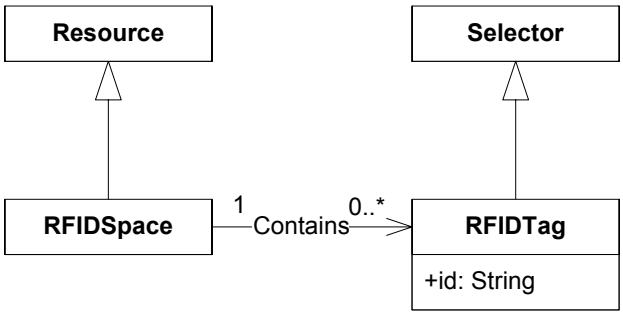


Figure 4.6: RFID plug-in

Our approach for identifying objects based on unique identifiers has many advantages compared to the simple mapping table approach described in the background chapter. Of course, a simple mapping table could be applied for resolving the unique identifiers to a Uniform Resource Identifier (URI). However, by implementing an RFID plug-in for the iServer architecture, we get access to all other iServer features such as, for example, user management for access control and are not limited in linking physical objects to URIs only, but can link them to any other iServer resource plug-in.

4.2 Application Programming Interface

Before we describe more advanced iServer features for integrating semantic resources, active content and information sharing, we would like to discuss some iServer implementation issues and present the application

programming interface provided for iServer applications. The iServer cross-media link model has been directly implemented on top of the persistent OMS Java object management framework [88]. Instead of defining a complex hypermedia architecture and then storing all the link information in a separate database, or even a file system, we directly empowered our database objects with the required hypermedia functionality.

The different types of objects forming part of the iServer model are bound to their corresponding persistent Java classes. Each Java class can be made persistent by implementing the `OMInstance` interface defined by the OMS Java framework. OMS Java then provides a fully operational model over objects represented by Java instances, collections and associations. In contrast to many other systems, there is no mapping required from the high-level modelling language used to define the cross-media link model to the actual model of the implementation platform, e.g. the relational model.

Figure 4.7 shows some of the methods defined on the abstract Java class `Entity` providing access to an `entity`'s data. This includes various methods for user management control such as `addAuthorisedUser` or `isAccessibleTo` and the handling of user defined entity properties such as `addProperty` and `getProperty`.

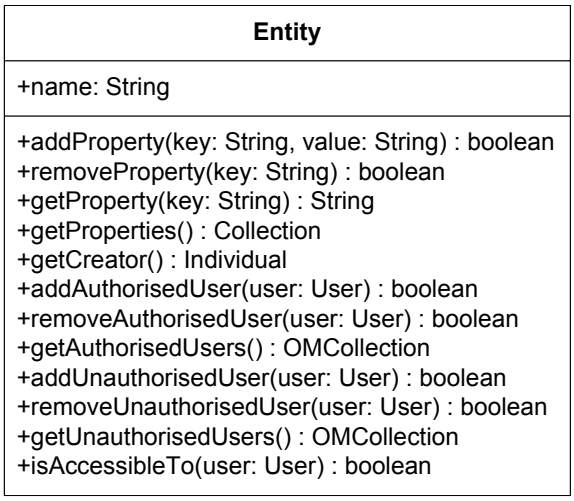


Figure 4.7: Entity API

The methods themselves are implemented based on meta information managed by the iServer link model. For example, the method `getProperties` uses the `HasProperties` association to find all properties defined on a specific entity. A slightly more complex task is to check if a specific user should be granted access to a specific entity. The corresponding `isAccessible` method works on the `AccessibleTo` and

`InaccessibleTo` associations and further has to check for a user's membership in groups etc. as described in Section 3.3.3.

Links, resources and selectors are defined as subclasses of the abstract `entity` type. The `Link` class shown in Figure 4.8 inherits all of the entity's methods and further adds some new, link-specific features. We can add and remove source or target entities to a link and access a link's source or target objects. Note that there are also some static methods, for example `getLinkBySource` or `getLinkByTarget`, which are used to find all the links defined for a given entity. By invoking, for example, the method `getLinkByTarget`, we get a collection of links having the specified entity as a link target.

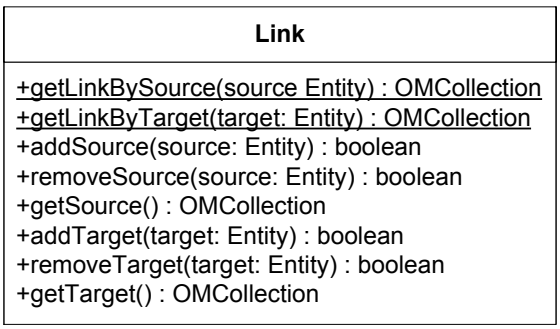


Figure 4.8: Link API

There exist similar Java classes for the remaining types of objects defined in the iServer cross-media link model such as `user`, `resource` and `selector`. In addition to the operations defined on specific Java instances, there is the special `IServer` class defining the iServer's main programming interface.

Figure 4.9 shows the most important methods of the `IServer` class. These static methods can be used to create, delete and access information stored in the iServer framework. The simple IServer API hides most of the administrative overhead. For instance, the invocation of the `createLink` method not only creates a new `link` instance, but also adds it automatically to the appropriate collections `Links` and `Entities`. Further, iServer will associate the newly created link with the specified creator, source and target instances by adding it to the `CreatedBy`, `HasSource` and `HasTarget` associations.

The implementation currently supports navigational access to links, access by classification of information and access by content-based queries supported by OM's query language AQL.

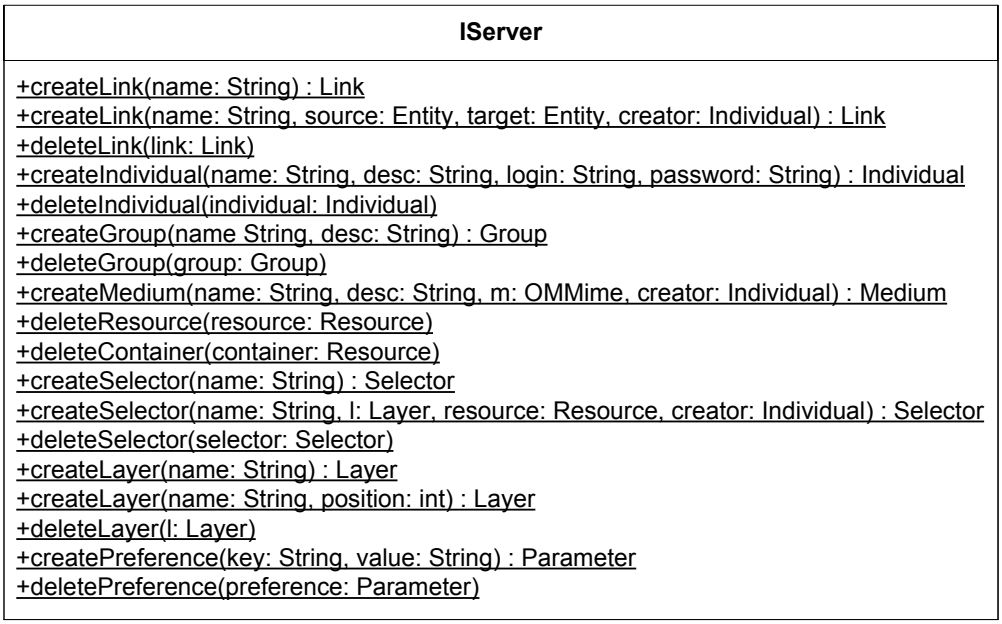


Figure 4.9: iServer API

4.3 XML Interface

The Java API is well suited for iServer applications which are also programmed in Java. However, we have decided to support other programming languages by providing an XML interface for accessing iServer link data. An example of an XML document defining a new iPaper document containing a single page and a single link is shown in Figure 4.10.

First of all, the XML document defines an `individual` with the related attributes such as name, description etc. Note that it is possible to use `ID`s and `IDREFS` for minimising the redundancy in an XML document. For example, an `id` attribute has been defined for the `individual` element, which is used by other elements to reference the `individual`. The XML document further defines a `document`, a `page`, a `layer`, a `rectangle` and an `activeComponent` element. Finally, a `link` is defined with the `rectangle` as a source and the `activeComponent` element as a target entity. Note that an `activeComponent` is a form of active content which, in contrast to static resources such as web pages or images, consists of a piece of program code that is executed on link activation. Details about this form of active content are provided later in this chapter in Section 4.5.

The same XML format is used to access link metadata and to create new links. For all of the iServer base classes, we have implemented XML wrapper components which serialise an object's content to XML and can

```
<?xml version="1.0" encoding="UTF-8"?>
<iserver>
  <individual id="beat">
    <name>Beat Signer</name>
    <description>PhD Student</description>
    <login>signer</login>
  </individual>
  <layer id="dl"><name>Default Layer</name></layer>
  <document id="paperpoint" creator="beat">
    <name>PaperPoint Presentation</name><id>paperpoint</id>
    <size><width>210</width><height>297</height></size>
    <content/>
  </document>
  <page id="pp1" creator="beat" document="paperpoint">
    <name>Page 1, PaperPoint</name><number>1</number>
  </page>
  <rectangle id="ss1" creator="beat" layer="dl" resource="pp1">
    <name>Show slide 1</name>
    <upperLeft>
      <point><x>76</x><y>102</y></point>
    </upperLeft>
    <size><width>25</width><height>8</height></size>
  </rectangle>
  <activeComponent id="s1a" creator="beat">
    <name>Show slide 1</name>
    <properties>
      <parameter>
        <key>ac:command</key><value>showSlide</value>
      </parameter>
      <parameter>
        <key>ac:slide</key><value>1</value>
      </parameter>
    </properties>
    <identifier>PAPERPOINT_CONTROL</identifier>
  </activeComponent>
  <link id="l1" creator="beat" sources="ss1" targets="s1a">
    <name>Show slide 1</name>
  </link>
</iserver>
```

Figure 4.10: XML import and export format

unmarshal a database object from its XML representation. In this way, link information can be exported in XML format and new content can be imported from XML files. To serialise and deserialise the new resource plug-in components, the corresponding XML wrapper classes have to be provided and registered with the XML registry. A detailed definition of the core iServer XML representation in the form of an XML Schema is provided in Appendix C. The iServer XML interface can further be used for flexible cross-media authoring from third party applications which can be implemented in any programming language and run on any operating system as we show later in this section when discussing the iServer Web Service.

4.4 Content and Semantic Resources

We have already outlined how the iServer cross-media platform can be applied to address parts of a resource and link them to different types of media. HTML documents, time-based data and RFID-tagged physical objects have been introduced as specific instances of resources. However, there is absolutely no information about the semantics of these "simple" resource types.

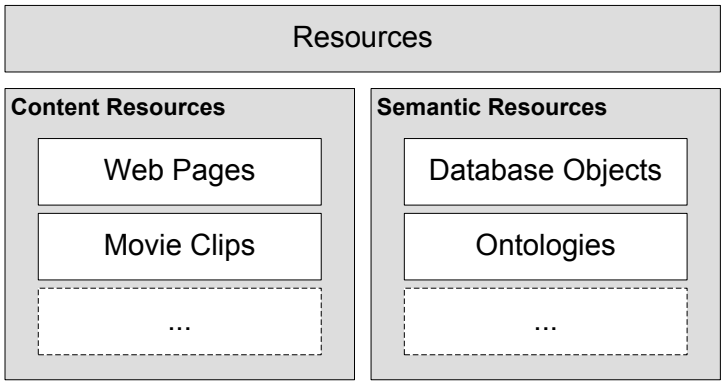


Figure 4.11: Content and semantic resources

Therefore, we introduce semantic resources as a new iServer feature and from now on distinguish between two kinds of resources as shown in Figure 4.11. A resource can either be a *content resource* or a *semantic resource*. There already exist a few simple predefined content resources such as text components, images and movies. Furthermore, it is possible to extend the system with any new type of content resource by providing the corresponding resource plug-in. Figure 4.11 introduces database

objects and ontologies as two forms of semantic resources. By linking to not only content resources, but also database entries, we gain some additional functionality.

The content resources can be classified in a hierarchical way by applying the composite design pattern which is represented by the collection `Containers` and the related association `Contains` as shown in Figure 4.12. The semantic expressiveness of such hierarchical classification structures is quite limited and hence, in addition to the content resources, the iServer architecture enables the integration of semantically rich resources which may represent domain-specific concepts or instances. In Figure 4.12, we present a small example of a movie database providing information about movies, directors and the movie stars appearing in the movies. Instead of linking to a single physical resource such as a movie, the iServer architecture enables us to link to the corresponding movie instance within the application database, thereby gaining additional semantics, such as information about the movie stars in the movie.

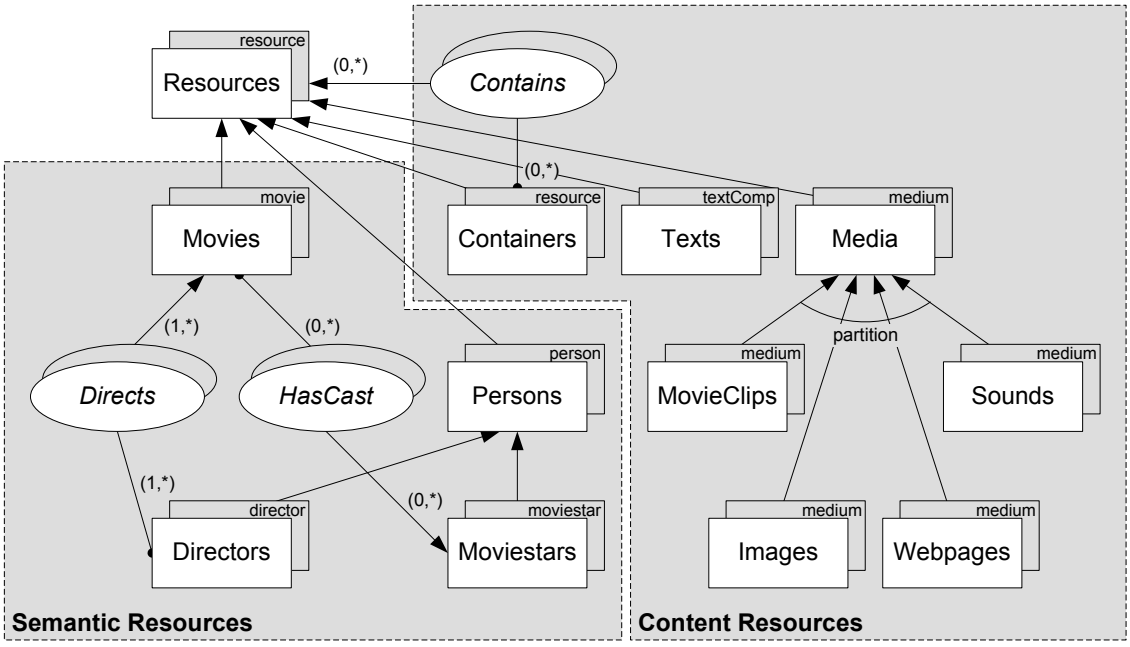


Figure 4.12: Movie database example

By having semantic as well as content resources, we support, not only completely heterogeneous and unstructured information spaces, but also well-structured database applications. Various object-oriented or relational database systems can be supported by implementing specific plug-ins for each type of database. In the case of a database plug-in, the selector can be defined by a database query, which enables parts of a

database's content to be addressed. We have implemented applications where semantically rich resources are directly stored as an extension of the iServer database as well as applications where external databases with application-specific information are accessed indirectly by iServer.

In the case of our OMS databases, a key feature is the uniform handling of data and metadata. Thus all metadata, inclusive both of schema and system data, is represented as objects and can be classified, associated and operated over in the same way as application data. This means that, as well as linking to application instances for the movie example as shown in Figure 4.12, it is also possible to link directly to metadata objects describing the concepts of `movie`, `person`, `director` etc. The latter corresponds to the use of metadata annotation in the semantic web [17] and, specifically, its use to generate links between documents.

An advantage of semantic content is that any supported iServer resource can be linked into a much richer information environment with information about application concepts and not simply content resources. If we take the example of the interactive paper plug-in, this implies that we have the potential to provide readers of printed material with a wealth of related digital materials, inclusive of dynamically generated hypermedia documents designed to meet the needs of that reader at that point in time and space. For example, we could define a link from paper to a semantic resource stored in a database which would return a customised context-dependent representation of its associated digital content. However, the new functionality comes at a price since it requires the development of application-specific solutions in terms of a database with information about the current domain of discourse. This requires domain expertise and initial large-scale investment. Nevertheless, it is important to point out the potential long-term rewards of such an investment when one considers reusability of information.

4.5 Active Content

So far we have presented simple content and semantic content as possible resources that can be linked by the iServer architecture. We now introduce *active content*, a third form of resource that was developed for the iServer platform to support the design of complex interaction components as required by many of the applications presented in Chapter 6. While regular links just return a single piece of information such as an HTML page, a movie or a semantically rich database object, active content is

represented by active components which are bound to a piece of Java code that can be executed either on the server or the client side [22]. An example of an active component that was implemented in the context of the interactive paper framework is a paper-based button. If a user points to a paper-based button, the corresponding instantiated active component may download a web page, contact a database, communicate directly with a running application or implement any other functionality. In Figure 4.13 we give an overview of the active component architecture.

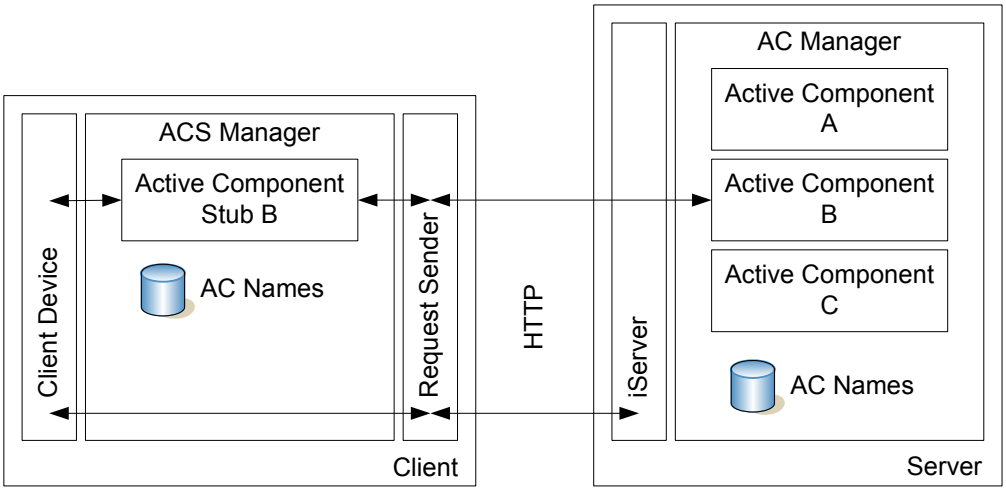


Figure 4.13: Client- and server-side active components

After some input has been generated by a client device, the client-side component has to check for any running active components. This means that 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 by the client device is sent to the server by using the `Request Sender`. On the server side, the incoming request is handled by iServer. The resolved link target can either be simple or semantic content or an active component. In the case that an active component has been selected, the Active Component Manager (`AC Manager`) will load the corresponding component on the server. 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. Since each active component is a subclass of an iServer resource, we use the properties which can be defined for each iServer entity for flexible handling of these additional parameters. Before an active component can be used within the iServer framework, it has to

be registered with the Active Component Name Directory (`AC Names`). For each active component identifier, a client (stub) and server (logic) binding has to be provided defining the Java classes to be loaded when an active component is activated. An example of an Active Component Name Directory is given in Figure 4.14.

```
<?xml version="1.0" encoding="UTF-8"?>
<activeComponents>
  <activeComponent>
    <identifier>PAPERPOINT_CONTROL</identifier>
    <logic>org.ximtec.ipaper.ac.logic.PaperpointControl</logic>
    <stub>org.ximtec.ipaper.ac.stub.PaperpointControlStub</stub>
  </activeComponent>
  <activeComponent>
    <identifier>PAPERPOINT_ANNOTATION</identifier>
    <logic>org.ximtec.ipaper.ac.logic.PaperpointAnnotation</logic>
    <stub>org.ximtec.ipaper.ac.stub.PaperpointAnnotationStub</stub>
  </activeComponent>
   ...
</activeComponents>
```

Figure 4.14: Active Component Name Directory

The server will check if an active component for the given identifier has already been loaded. In the case that the appropriate active component has already been loaded, it is reused since every active component is loaded only once by the Active Component Manager. If a new active component logic has to be loaded, its `init` method is invoked by the Active Component Manager. Any program logic can be implemented within the `init` method and, after the method has terminated, an XML description of the active component is sent to the client side. On the client side, the Active Component Stub Manager (`ACS Manager`) does a lookup in the Active Component Name Directory, loads the corresponding stub component and invokes its `init` method for initialisation. Note that active components do not persist on the client side and have to be reloaded each time.

The client notices that a new active component stub has been loaded as a result of the current request, switches to the active mode and dispatches the original request parameters, e.g. the position detected by the digital pen, to the active component's `handleEvent` method. Since the client is now in active mode, all subsequent client device events will

be handed over directly to the active component's `handleEvent` method. However, as soon as the active component's `setDone` method gets invoked as a result of the component's program logic, the active component stub will be unloaded and the client will switch back to browsing mode. An active component can also be terminated automatically if it has been idle for a specific amount of time as defined by an active component timeout.

A client-side active component stub can communicate directly with the server-side active component logic by sending a special active component request. All information encoded within an active component request is handed over to the server component's `handleActionRequest` method. The result is sent back to the client-side active component in XML format. Figure 4.15 summarises the functionality available on active component stub and logic entities. A new active component is implemented by subclassing the stub and logic classes and providing the specific program logic for the `init`, `handleEvent` and `handleRequest` methods.

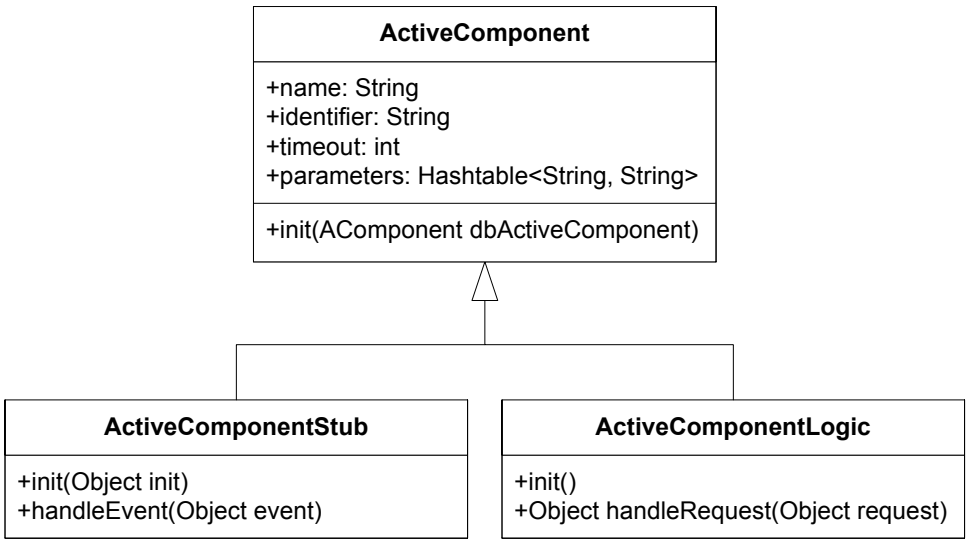


Figure 4.15: Active component stub and logic

The concept of client- and server-side active components that can communicate by sending special active component messages to each other has proven to be very useful if the client-side component has to retrieve additional information managed by the iServer database. Note that the very flexible and powerful active component concept enormously simplifies the implementation of complex interaction components. Various active components have been implemented for different interaction tasks as part of numerous interactive paper applications. The growing set of reusable active components simplifies the development of future

interactive iServer applications. For example, we have implemented active components to capture notes from pen-based input, replay animations of captured handwritten information, communicate with database systems, control web browsers, play movies and sounds, control arbitrary Microsoft Windows applications etc. Some of these active components will be described in more detail when we present various applications that have been implemented based on the iServer platform in Chapter 6.

4.6 Visualisation

As mentioned earlier, an architecture for interactive paper should be flexible in handling different forms of existing as well as emerging input technologies. The same flexibility should be offered for the visualisation of linked media. An application based on the iServer platform should be accessible from a variety of client devices and hence the architecture has to be extensible to deploy the information stored in the cross-media link server on new kinds of output devices as soon as they become available on the market. This is not a new issue in web publishing where content management systems are used to support various types of customised output channels.

To support universal client access to the iServer link space, we extended the object-oriented data management system OMS Java with an eXtensible Information Management Architecture called XIMA. Through a clear separation of the notions of information, structure and presentation, XIMA supports universal client access to arbitrary OMS Java databases [89, 128, 158]. Based on a dynamically generated XML representation of the information stored within the iServer platform, all client-specific rendering is realised using the Extensible Stylesheet Language Transformation (XSLT).

By using XIMA as an access layer to the iServer platform, we have a very flexible means of deploying customised information to different types of client devices, and also to enable variations in visualisation for the same type of device. A regular user of an application can customise the user interface by defining their own transformation stylesheets, whereas the application already provides different predefined default visualisations for specific groups of users.

The eXtensible Information Management Architecture for flexible multi-channel content delivery, which is used as an access layer to the iServer platform, and therefore also to the interactive paper framework,

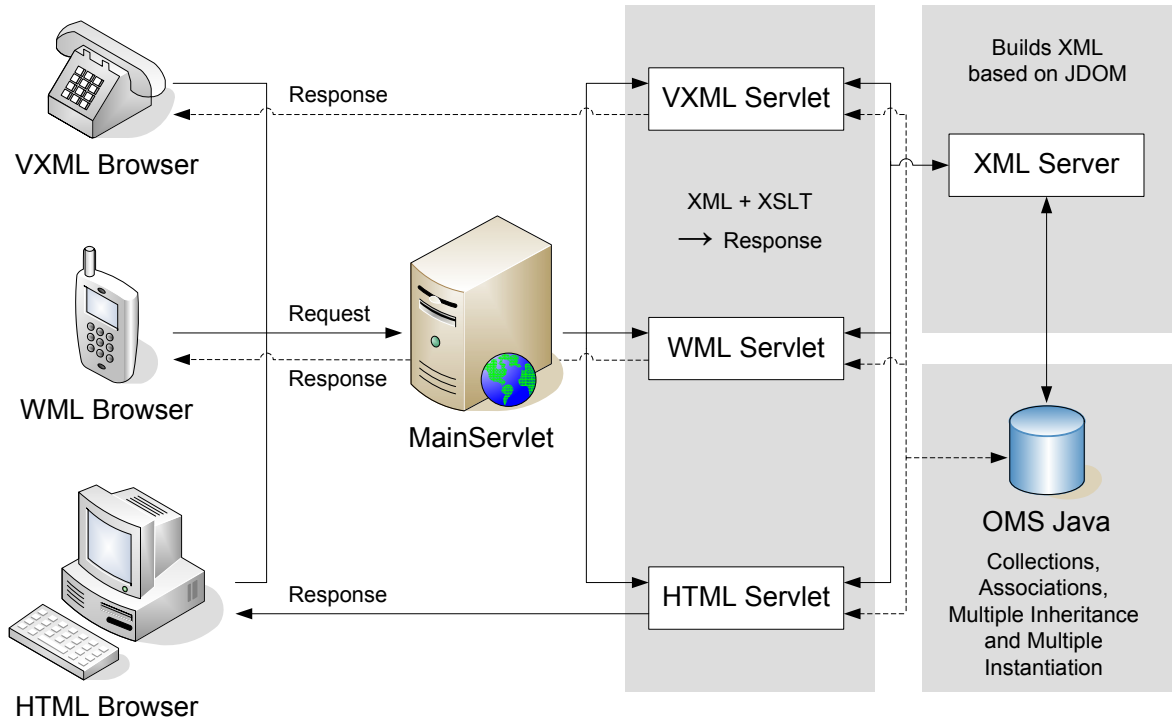
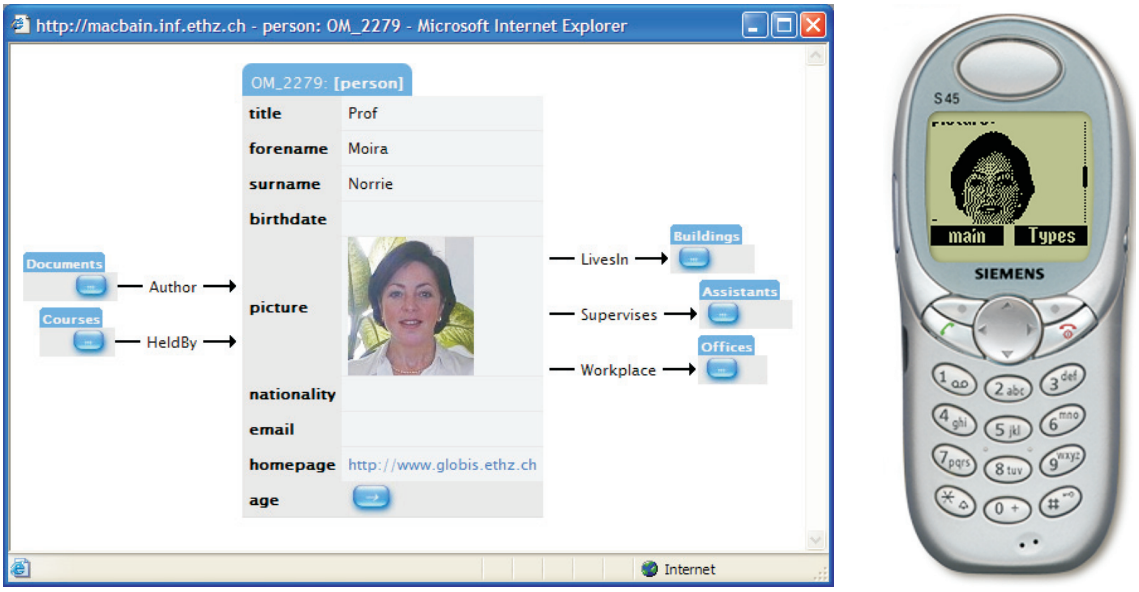


Figure 4.16: XIMA architecture

is shown in Figure 4.16. While XIMA provides flexible access to the information stored in iServer, it misses some of the concepts such as multi-lingual content delivery to make it a *full* content management system [59]. However, we can always use iServer to link directly into such a fully-fledged content management system if necessary.

For a specific application, all client access is accomplished via a single Java servlet, the `MainServlet`. This implies that only a single URL needs to be memorised for a specific website, rather than having different URLs for different types of clients. The `MainServlet` detects the user agent type from the HTTP request header and delegates the handling of the request to the appropriate servlet. Currently, the XIMA framework supports four types of input/output channels for accessing and editing information. In addition to standard HTML browsers, the information is accessible from WAP-enabled mobile phones using a WML browser and i-mode phones based on CHTML. The default representation of a database object for a web browser and a WAP-enabled mobile phone is shown in Figure 4.17. Finally, XIMA provides an interface for regular voice phones based on speech recognition and speech synthesis mechanisms provided by VoiceXML and its associated available voice servers [161, 162].



(a) HTML browser (b) WAP phone

Figure 4.17: XIMA input/output channels

The request handling servlets then access the database by connecting to an OMS Java workspace via the OMS Java API. The connection may be either direct or via the XML server component. Direct connections deal with requests that do not involve data retrieval such as checking the membership of an object in a collection or performing updates. Any requests that involve the retrieval of data go through the XML server which forwards them to the OMS Java data management system and generates XML representations for any data objects to be returned to the servlets. The requesting servlets then use the appropriate XSLT templates to transform the XML results to the corresponding responses to be returned to the client.

A point to note in this general architecture is that we are not storing any XML documents, but rather generating them dynamically from the application data which is stored according to the information model. Since the information model is a loosely-coupled graph model based on object collections and associations, this gives much more flexibility than the rather restrictive hierarchical-based models imposed by XML structures. At access time, a particular hierarchical view on data is derived and the appropriate XML structure generated.

Using generic XSLT templates for the various client devices, we are able to provide generic browsers and editors for the current set of client device types. Adding a new type of client device involves implementing the corresponding servlet and writing the appropriate XSLT templates.

Specific application interfaces are supported through the customisation of XSLT templates and the specification of document structures in terms of how document content is built dynamically from information objects.

XIMA is a very flexible data management framework that can dynamically construct information presentations from entity representations, associations between entities, pieces of content, document structures and templates at access time. Such flexibility comes at a price in terms of response times, and therefore multi-level caching schemes are vital to ensure acceptable performance [157]. It is important to emphasise that it is also possible to generate entire static websites off-line with such an infrastructure. In particular, and relevant to our work on paper interfaces, it is also supports the generation of printed documents through the use of appropriate XSLT templates as we show later when discussing authoring issues for interactive paper.

4.7 Distribution

The iServer architecture that we have presented so far stores all link information in a single centralised database. Users can access preauthored links that have been defined in the database as well as adding their own new links which then can be shared with other users based on the integrated user management component.

However, it would be good to have a more open and flexible system which does not depend on a single centralised link server. Therefore we have also implemented a decentralised iServer infrastructure where each user stores link metadata in their local iServer database and at the same time can access link information stored in others users' personal iServer instances using peer-to-peer (P2P) technologies for information discovery and exchange [38, 69].

In developing a distributed architecture for iServer, we were keen to ensure that the framework kept to its principle of being as general as possible and with a clean separation of concerns. The first stage was to define a very general interaction framework between peers as shown in Figure 4.18. Generally, the iServer Java API defines methods to retrieve, manipulate, add and remove data and the goal of a P2P service is to offer the iServer functionality of a remote system to a local system.

We now describe the various steps ⓐ to ⓗ in terms of its implementation. In the local system, a request factory is used to get a `Request` object which is handed over to a local request handler ⓐ. The handler

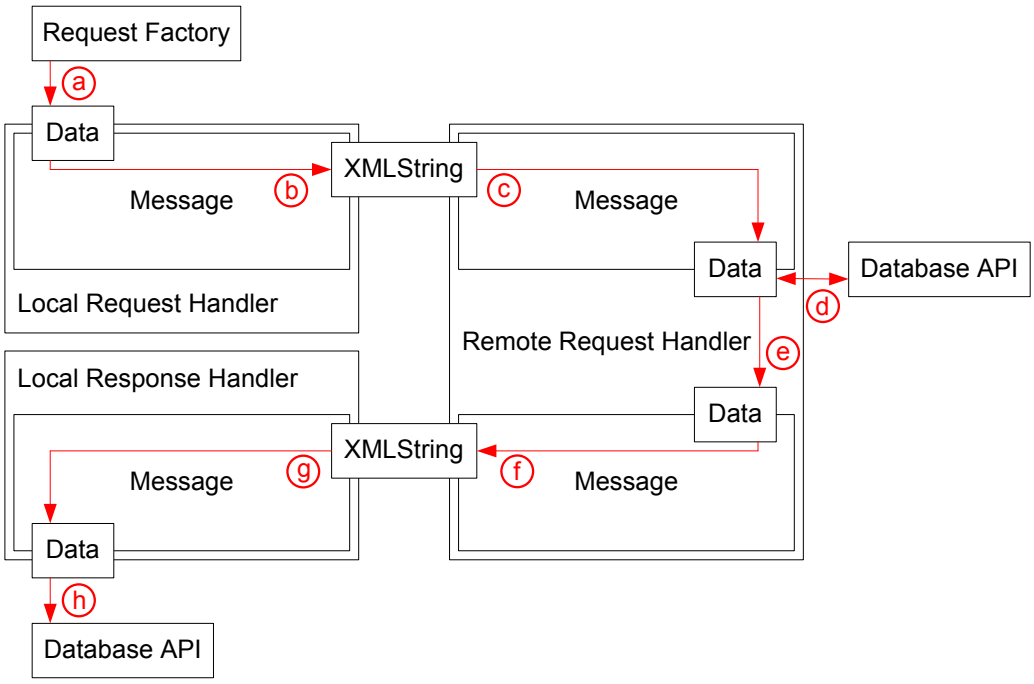


Figure 4.18: Interaction scheme for P2P services

creates a `Message` object containing the request and sends its XML string representation to a remote request handler ⓑ. There the message is reconstructed from the string value received and the `Request` object extracted ⓒ. The remote request handler initiates the processing of the request which is carried out by the `Request` object itself ⓓ. The resulting `Response` object ⓔ is wrapped with a `Message` object ⓕ and its XML string representation is returned to the local response handler. The response handler reconstructs the message and extracts the response ⓖ. The response is fed back to the local database ⓗ. Note that the `Request` and `Response` classes are derived from a common `Data` superclass.

In this interaction scheme, we identify five components which participate in the interaction between the local and the remote system: a request factory, request and response handlers, data and message objects. A service framework such as JXTA [174] provides the means to locate and send requests to a remote system as well as to respond to a request. The request handlers encapsulate the framework-specific functionality by defining methods to handle requests and obtain responses so that they are transparent to the other components.

Messages can be sent as plain text, binary representation, compressed or encrypted. Any form can be implemented within the messaging component. Only the request factory is affected if the database, its API and schema definition are altered or completely exchanged. The manner in

which a remote database is accessed is defined by the `Data` subclasses hiding the database API access from all other components. Finally, the processing of responses can be specified with the implementation of a response handler.

A request factory was implemented that replicates the iServer API and therefore provides local access to remote iServer instances. For each iServer method, we defined a factory method creating a `Request` object which encodes it using Java Reflection [46]. The object is sent to the remote request handlers which invoke the method on iServer classes and objects. The `Result` objects contain collections or single instances, which are returned to the local response handler and made available locally. Additional data such as user representations and potential remote error descriptions are stored in the message object. Data is exchanged in XML representations conforming to an XML schema definition included in the iServer platform for the general import and export of data.

Note that the same interaction framework was used to implement an iServer *Web Service* based on the Axis web service framework [61]. For the Web Service no local request and response handler had to be implemented. The only thing that had to be provided was a Web Service-specific implementation of the remote request handler providing a Web Service method for each iServer API method.

We used the JXTA framework to implement our iServer P2P service and, before explaining the P2P specific components developed for iServer, we introduce some commonly used terminology. A *peer* is an entity that sends, receives and processes messages. Multiple peers are organised in *peer groups* which they can create and join. A peer group provides *group services* to its members such as peer and group discovery, one-to-one communication and broadcast facilities.

An iServer peer group is created by any peer that cannot find an existing group. Otherwise it will join any of the groups that it can find. Requests are broadcast to all members of the group and responses sent back to the requesting peer only. This behaviour is implemented in the handling component. When a client calls an iServer API method through our iServer P2P service, it can either register itself to be notified whenever a response is received or retrieve all responses at once.

As a simple example used to test the P2P system, we have developed a small application to visualise information spaces where the resources are pieces of text. The resulting spaces are therefore very similar to elementary concept graphs as shown in Figure 4.19. Nodes and edges are coloured to indicate the original sources of the information.

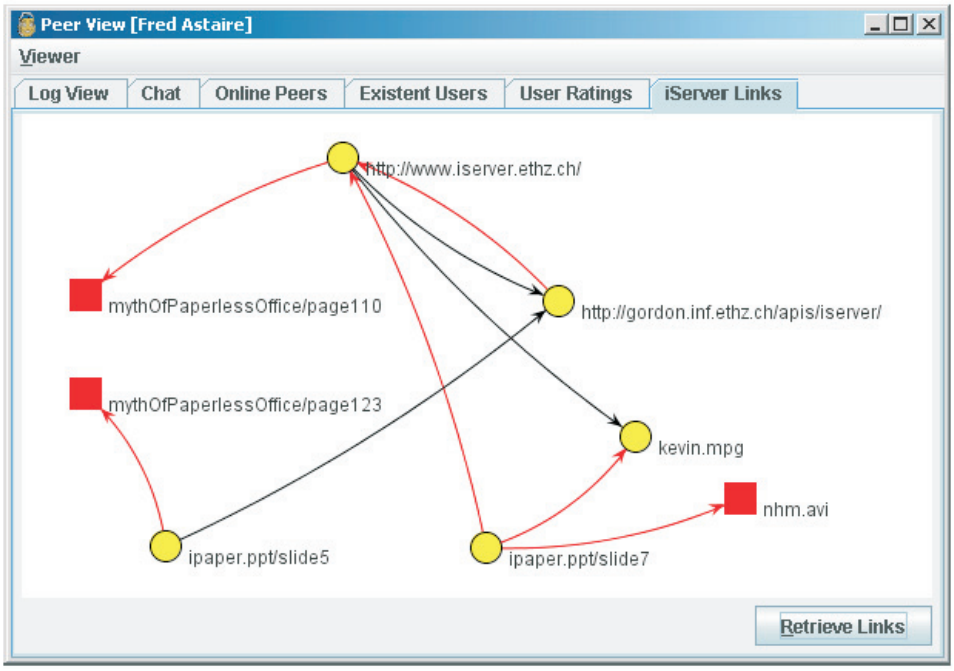


Figure 4.19: Visualisation of local and remote links

Figure 4.19 furthermore outlines the general form of interface offered by the framework and the auxiliary functionality. Users can get information about other users currently using the system and also chat with them which provides another important channel for knowledge exchange. They can also view information about user ratings and specify their own explicit ratings where desired. We now go on to describe how these ratings are used to filter information obtained from remote peers.

In a cooperative community of publishers and consumers equally responsible for the available content, several issues must be addressed. Brookshier et al. [23] name three typical problems:

- A *leecher* is a participant who consumes without contributing. Communities consisting of too many leechers produce little content and all network traffic is focused on a few publishers which decreases their availability.

- Any community supporting democratic publishing can be a target for unsolicited content such as advertisements i.e. *spamming*.

- *Malicious content* can be published for various reasons with the goal to reduce the community's worth.

Some systems try to avoid the first problem by keeping designated publishers to ensure the information supply, and of course it is always

possible to support this model in the iServer framework. Since the "content" in our information spaces consists of resources as well as links and we primarily envisage shifting the authoring of links between resources to the user community rather than the resources themselves, the effort required of the user is minimal and therefore should not discourage them from cooperating.

Our approach to dealing with issues of spamming and malicious content is to introduce a user rating system. Individual users can decide on their own levels of trust in certain users, thereby accounting for any lack of social agreement about the trustworthiness of individual publishers. User ratings are combined with response ratings to filter responses returned from remote peers. Not only does this improve the quality of information presented to the user, but it also reduces the quantity thereby preventing the information overload that could result from a large, highly-connected information space.

User rating is a frequent approach to content filtering and various techniques of rating users and interpreting rating values have been proposed and deployed. Jøsang et al. [79] compiled a survey of the current state of research in this area. The fact that a user rates another user with a value between 0 and 1 can be interpreted as a probability with which the rated user fulfills the expectations of the rating user [50]. Such a value reflects the reliability of a rated user. Another interpretation has been proposed by McKnight and Chervany [115] where the rating value denotes an amount of trust which combines reliability with a weight reflecting the importance of an expectation to be fulfilled. A third interpretation widely used in web communities such as web forums and online markets is that the value represents the reputation of a user. In this case, the rating of a user consists of an aggregation over the values of all rating users, e.g. the sum or average. Another issue is the transitivity of ratings. A set of user ratings can be represented by a graph where a directed weighted edge connects a rating user with a rated user. If two users are not connected by an edge, a path may be found connecting them transitively.

The aggregation of weights along a sequence of edges is called *discounting*. Multiple paths may exist, in which case, multiple discounted values must be considered in order to obtain a rating value. The process of aggregating discounted values over parallel paths is called *fusion*.

We encode user ratings as tuples containing the rating user, the rated user and the rating value. Such a tuple is created by the rating user and propagated to all other members of the peer group. A user rating

manager ensures that all peers have the same set of tuples stored locally. This tuple synchrony is achieved as follows:

1. On startup, the peer tries to read a file containing tuples stored in previous sessions. If the file does not exist, an empty one is created.

2. The peer creates a tuple set S_{local} containing all tuples in the file. Whenever S_{local} changes, the file is updated.

3. When a peer joins the group, it requests the tuple set S^i_{remote} from all other members i of the group.

4. Every incoming tuple set S^i_{remote} is compared with the local set S_{local} as follows:

   - If S_{local} does not contain all tuples in S^i_{remote} then the local set is updated.
   - If S^i_{remote} does not contain all tuples in S_{local} then the local set is broadcast to all other members.

5. When no more sets are broadcast, all tuple sets contain the same tuples.

6. Whenever a new rating is set locally, the local set of tuples is broadcast to all other members of the group.

We implemented a `UserRatingManager` class which encapsulates the process of synchronising rating tuple sets. It maintains a graph containing all user ratings. A rating manager is associated with a peer and is used to process specific responses as described below. We interpret a rating value as the amount of trust flowing from a rating user to a rated user. Whenever a user is to be rated, the manager runs a max flow algorithm on the graph with the owning user as source and the rated user as sink. Note that the use of a graph allows arbitrary discounting and fusion techniques.

Since every iServer P2P API request is broadcast to all members of the groups, a peer possibly receives multiple responses. Therefore, as shown in Figure 4.20, we provide a response rating manager where incoming responses are filtered before they are made accessible. The selection is based on rating values that are computed by response raters for every response. A response rater returns a rating value given a response, the

responding user and the collection of responses previously received. The rating values of multiple response raters are weighted combined by an interchangeable aggregator function such as addition, multiplication, average etc. We implemented two response ratings, one returning a rating for the responding user and the other one based on the number of times the same response is returned (frequency).

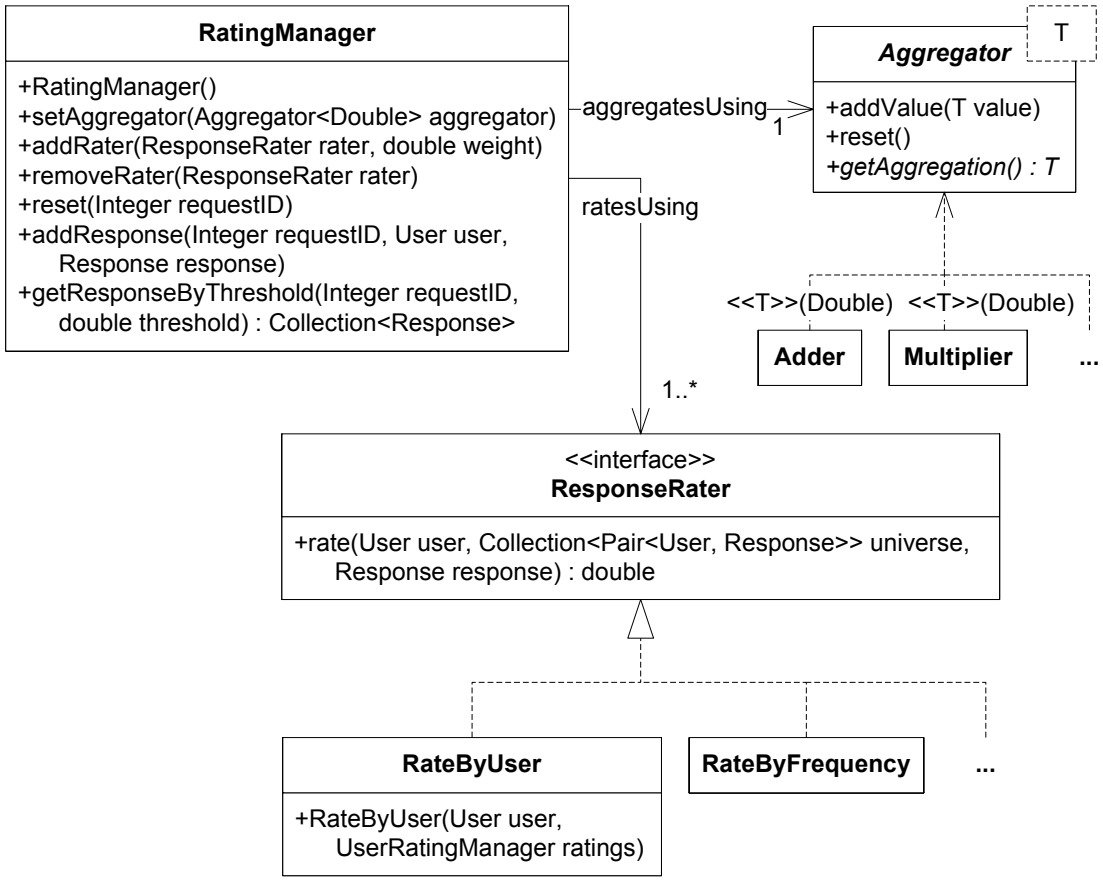


Figure 4.20: Response rating

A response is identified by its associated request and the responding user. For every request, the rating manager maintains a set of responses, each paired with the responding user. It also manages a set of registered response raters that are used to compute an overall rating value. Each rater is associated with a weight which is multiplied with the rating value computed before being aggregated with the overall rating value. The local response handler adds incoming responses to the respective set. There are two ways for a user to access the responses: Either they explicitly request the set of responses to a particular request or they specify that they should be notified whenever a response arrives that passes the filtering. In the former case, the rating manager iterates through all responses

in the respective set, aggregates the rating values returned by every registered response rater and returns all responses having an overall rating value greater than a threshold.

As stated above, there are many different techniques for handling ratings and the one described was developed based on experimentation with some of the alternatives. So far, it seems to perform well in terms of the behaviour that users might expect from the system. Note that the framework is independent of the specific filtering techniques used and it is even be possible that specific applications can customise the rating system.

4.8 Summary

In this chapter we have presented iServer, an information platform providing the core functionality for flexible cross-media information management. The iServer platform has been implemented on top of the OMS Java data management system and provides a Java API as well as an XML interface for language independence. Our architecture not only allows the linking of various types of digital information, but also enables the digital augmentation of physical objects as well as the physical annotation of digital objects. The concepts of multi-layered selectors and integrated user management together with a very general concept of a link build the fundamental concepts used by all the resource plug-ins presented in this chapter.

By providing specific resource and selector implementations based on an object-oriented plug-in mechanism, it becomes easy to integrate new types of media and also associated forms of linking supported in existing hypermedia systems. In addition to the plug-in for interactive paper, currently there exist three other plug-ins for XHML documents, movies and RFID tags. Semantic content and the concept of active components have been introduced as a tool enabling the implementation of complex interaction designs by providing the possibility to integrate database objects as well as small pieces of Java program logic that are executed at link activation time.

While many of the features of the iServer framework and its cooperative version can be found in other projects and systems developed within the hypertext community [62, 140, 184, 186], the main contribution of our work is to combine these in a single, extensible framework. We were able to achieve this by adopting an information systems approach

and applying established techniques of metamodel-driven system engineering and object-oriented database technologies. Our goal was not to simply support mainly browsing activities, but complex interactive services based on the underlying information space. Furthermore, associative inter-application and cross-media linking as supported by the iServer architecture enables effective information management across isolated application domains. The wide variety and complexity of interactive systems that we have been able to develop based on iServer within the short period of time that it has been available has demonstrated the benefits of the approach.

The eXtensible Information Management Architecture enables universal client access to all information managed by iServer. Based on XML as an intermediate data representation, XIMA dynamically transforms and deploys the resulting information in the appropriate output format.

In addition to the client-server approach, we implemented a distributed iServer prototype for collaborative information sharing based on completely decentralised peer-to-peer computing. By applying user and content rating mechanisms, a certain quality can be guaranteed in an open link system where everybody can add new information.

After presenting an implementation of the iServer cross-media platform together with various resource plug-ins, including one for interactive paper, in the next chapter we present the remaining components that had to be implemented to achieve an integration of paper and digital information spaces.

Can you imagine that they used to have libraries where the books didn't talk to each other?

Marvin Minsky [94]

5 Interactive Paper

This chapter gives an overview of the main software components that we have developed as part of the iPaper plug-in for integrating paper and digital information [102, 130, 131]. While details about iServer and its extension for different kinds of media have been presented in the previous two chapters, in the following sections, we describe the framework for interactive paper that was built on top of iServer. We start by outlining the requirements for interactive paper and present our solution to solve these problems.

5.1 Requirements

Some of the requirements for achieving a true integration of printed and digital information sources have already been introduced in the background chapter, where we analysed existing interactive paper projects. The requirements that are addressed by our platform for interactive paper and the underlying iServer cross-media information management architecture include:

- Support for all types of links between paper and digital media, including paper-to-digital, paper-to-paper, digital-to-digital, and digital-to-paper links.
- Active areas and flexibility in defining the granularity of links.

- Support for simple media types, semantically rich database objects and active content.
- Flexibility in supporting different input devices and position encodings.
- Integration of arbitrary heterogeneous information sources.
- Context dependent content delivery based on output device etc.

Since we can never be sure which resource types should be supported in addition to the interactive paper medium in the near future, we decided to implement the interactive paper framework as an extension of the iServer cross-media information management platform. This has the major advantage that all of the iServer's functionality, such as multi-layered links, user management etc. can also be used by the interactive paper framework. Furthermore, based on iServer, we can not only link to simple resources, but also semantically rich database objects or active content as described earlier. Finally, on demand, the interactive paper framework can be extended to deal with new types of resources by implementing new resource plug-ins for the iServer platform.

Generally, the requirements of any interactive paper system will depend not only on the application domain, but also on the envisaged sphere of interaction and scope of information. It is therefore essential that the underlying information structure be able to support all forms of systems — including the basic linking of printed documents to multimedia files as well as advanced specialist information environments that freely link back and forth between the digital and paper worlds.

5.2 System Components

The interactive paper framework is based on a classic client-server architecture as shown in Figure 5.1. On the client side, a special input device (e.g. digital pen) which is connected to a computing device such as a regular PC in a desktop environment, a portable computer or a PDA is used to detect (x,y) coordinates within a paper document. 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 and page number together with the positional information are transmitted via an HTTP request to the server component responsible for further data processing.

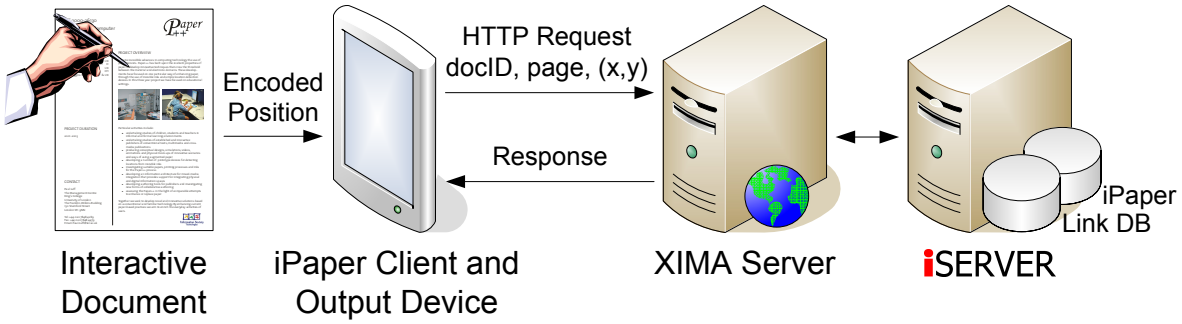


Figure 5.1: Interactive paper architecture

The server side consists of XIMA and iServer with the iPaper plug-in. iServer itself uses the OMS Java data management system for persistency which results in the three main server components shown in Table 5.1. First, the appropriate information is looked up by iServer and its iPaper plug-in based on the positional input parameter. Any linked information can either be accessed from the underlying OMS Java data management system or retrieved from external resources. XIMA then transforms the information into the appropriate output format, e.g. HTML for a regular web browser or VoiceXML for voice applications, and sends it back to the client device.

Component	**Task**
XIMA	Channel Management: Java Servlet framework to handle requests from different client devices.
iServer	Link Management: General cross-media link platform using OMS Java for persistency.
OMS Java	Data Management: Database handling content, application, and contextual information.

Table 5.1: Information server components

In Figure 5.1, the digital presentation device is a Tablet PC which also handles the communication with the input device and the appropriate transformation of the information encoded in the paper documents. However, note that, in theory, it is not necessary that the processing of the information from the input device and the result visualisation take place on the same device. For example, in a public space, the result could be presented on a large public screen, while the transformation takes place on a user's individual portable device. The architecture shown

in Figure 5.1 has been used in many of our applications for interactive paper. Nevertheless, as we show in applications described in Chapter 6 there exist alternative set-ups for request and response handling.

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 requested by the server is the document identifier, a page number and an (x,y) position. The server component does not depend on a specific input device, although certain technologies may be able to support extended functionalities in terms of user interaction.

5.3 Input Devices

In Figure 5.2 we show an example of the position encoding used by the Paper++ inductive reader presented earlier in Section 2.2.2. The detection of positions within a page is realised by printing on standard paper a grid of barcodes encoding information about the page number and the corresponding position within a page in terms of (x,y) coordinates using a conductive invisible ink. Within the Paper++ project, these encoded paper documents had to be produced in an additional preprocessing step on special printing machines. However, in the future it should be possible to print the inductive grid and the corresponding artwork in a single step by, for example, using an additional ink jet cartridge containing the conductive ink. By swiping parts of a page with the inductive reader which is capable of measuring differences of inductivity, the barcodes are read and decoded to the corresponding page number and the actual position of the reader.

The document identifiers are handled separately and are not encoded within each page of a document to save address space. The user has to register a document by scanning a special code on its cover before they start working with the document. All subsequent information captured by the pen will be processed in the context of this document identifier. An alternative approach we experimented with is the use of passive Radio Frequency Identification (RFID) tags to distinguish different documents.

Applying technologies that enable invisible, or nearly invisible, encodings brings us back to the issue of how users know where links are located within a page. In terms of published content with embedded active links, highlighting can be used or even recognised conventions that have arisen from the use of web browsers where users have come to expect

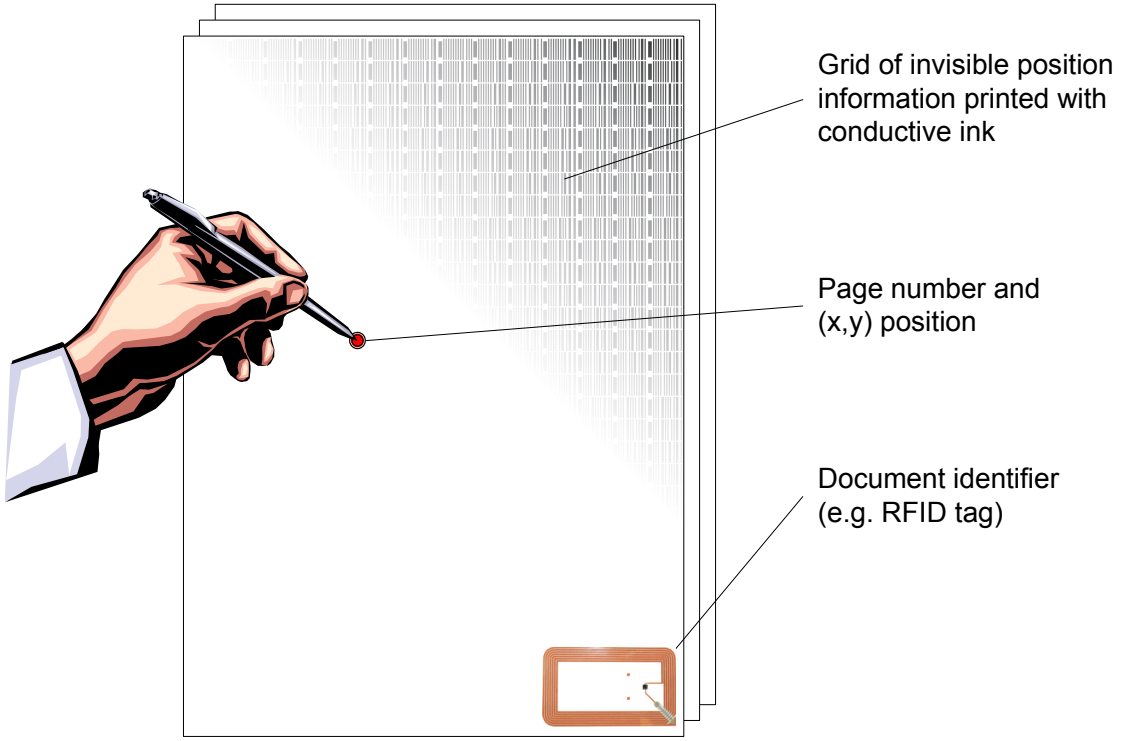


Figure 5.2: Document encoding

content items such as headers, images and underlined or blue text to be possible links. For specific applications, it may even make sense to omit any visible hints about additional information and leave the task of finding the links completely to the user. For example, in a children's book, it may be part of a learning task to find additional information such as realised in the LeapPad suite of books [97]. In the case of user generated links, the simplest way to visualise the information anchors is to combine the wand device with a marker pen allowing the user to highlight areas on paper while adding a new link. We also experimented with the use of audio feedback to inform users that they have activated links. With the further development of reader devices, other forms of audio or force feedback from the reader device itself could be used to inform a user of the existence of a link on rollover.

The development of printing technologies, barcode schemes and input devices on which such systems could be based is very much an active area of current research. Solutions do exist, but at present they tend to be expensive and based on visible encoding patterns. However, it is expected that cost-effective solutions which are almost invisible and have encoding schemes that provide a good level of position granularity will be available in the near future. Therefore, an important factor in the development of

the interactive paper framework was to ensure that it is independent of any particular technologies and encoding schemes. The platform should support the integration of different input devices with minimal effort. The only assumption we make is that an input device can deliver the position within a document.

It was already clear when we started implementing the interactive paper platform that different input device prototypes for retrieving positional information from a paper document would be integrated in the future. Our software platform for integrating paper and digital information therefore, not only had to be flexible in supporting different types of input devices, but also multiple forms of encoding schemes. The resulting solution is based on a few abstract interfaces with a specific implementation for each input device and encoding technology.

By implementing only a small set of Java classes conforming to these interfaces, we were able to easily integrate new types of input devices or position encoding schemes such as, for example, the commercially available Nokia Digital Pen [123] using the Anoto encoding pattern which has been introduced in the background chapter. Note that the Nokia Digital Pen does not support the continuous transmission of information by default. However, we have access to a prototype version of the Nokia Digital Pen modified by the Anoto engineers which, in contrast to all other existing Anoto solutions, works in a streaming mode, sending position data immediately and continuously to the computer. This enables us to use the Nokia Digital Pen as an interaction device as well as for writing capture. More recently we integrated Maxell's *Magicomm G303*, the first commercially available digital pen that offers streaming functionality based on Anoto technology.

To easily support the integration of new hardware prototypes developed within the project as well as commercially emerging solutions for position detection, the abstract input device interface shown in Figure 5.3 has been designed. The `InputDevice` interface has to be implemented for any input device available within the interactive paper framework. It provides methods to add and remove `InputDeviceEventListeners` which are informed about information processed by the input device. Each implementation of a specific reading device handles the low level communication with the pen over the serial port interface (RS232), the Universal Serial Bus (USB), Bluetooth or any other connection and at the same time conforms to the `InputDevice` interface. The abstract class `AbstractInputDevice` implements the functionality for managing `InputDeviceEventListeners` and can be used by any specific input

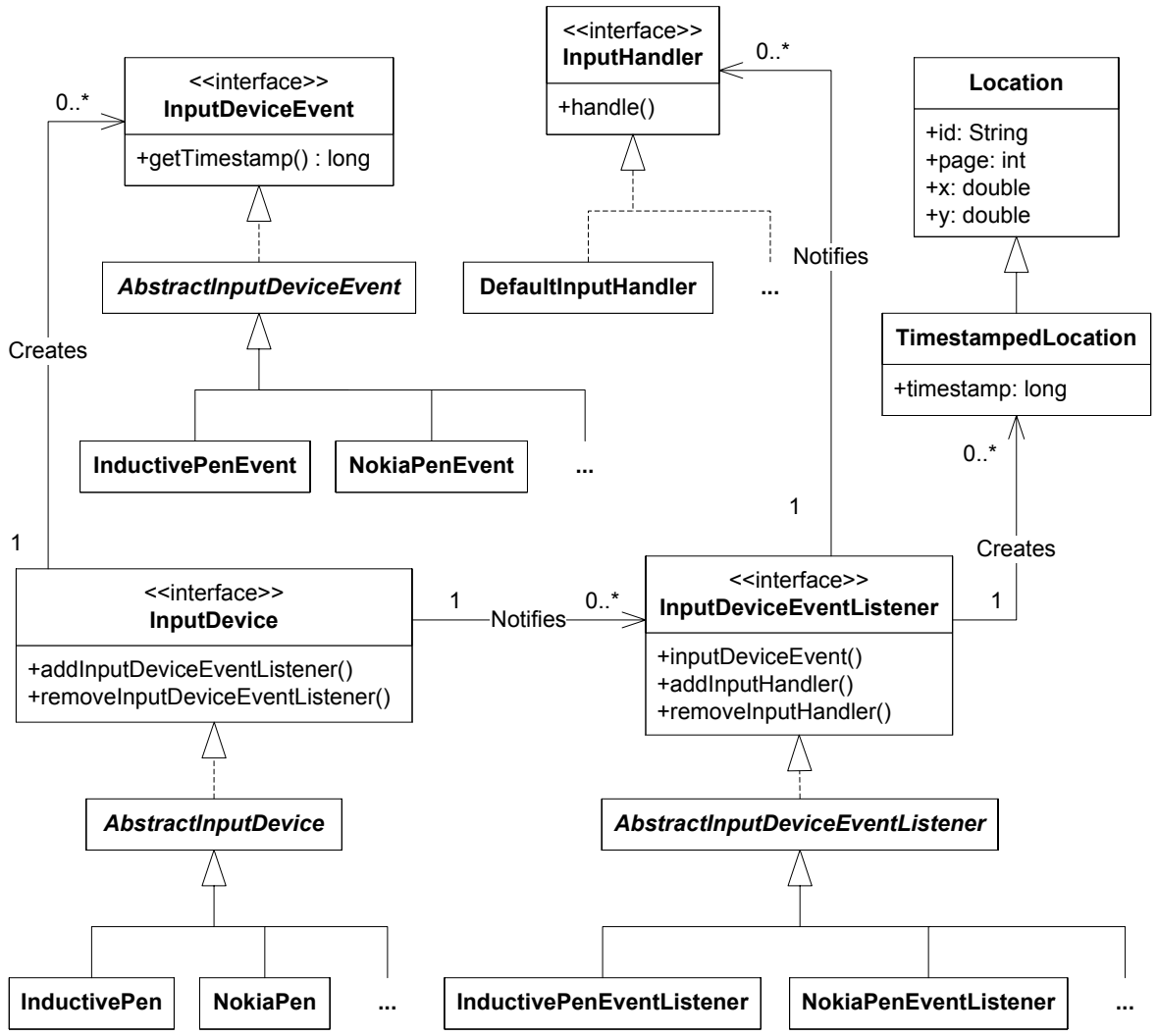


Figure 5.3: Input device interfaces

device implementation. In Figure 5.3 we show two of the input device implementations that have been developed for the interactive paper framework. The `InductivePen` processes information from the Paper++ inductive reader whereas the `NokiaPen` is an input device implementation for the Nokia Digital Pen based on Anoto technology.

After the pen captures some data, it delegates this information to all registered `InputDeviceEventListeners` in the form of an `InputDeviceEvent`. Again, an `InputDeviceEvent` is only an abstract interface which is implemented by concrete events fired by specific input device implementations. Common to all `InputDeviceEvents` is that they provide access to a timestamp with information about when a data packet was captured and the event created. Again, an `AbstractInputDeviceEvent` implementing the basic functionality required by the `InputDeviceEvent` interface is provided as a base class for device-specific events.

The InputDeviceEventListener is the third interface that has to be implemented to support a new kind of input device. An InputDeviceEventListener processes the information stored in the InputDeviceEvent provided by the InputDevice. This information is still in a input device-specific representation and has to be transformed to a TimestampedLocation, the format that can be handled by the interactive paper framework. A TimestampedLocation contains information about the document identifier, the page number and the (x,y) position within a page. Further, each TimestampedLocation provides temporal information about the time when the location was captured. Details about the address space mapping from the reader-specific information to the address space of our interactive paper architecture in the form of a TimestampedLocation are provided later. After applying the address space mapping, the InputDeviceListener sends the TimestampedLocation object to all InputHandlers which are registered for a specific InputDeviceListener.

In the default setup of the interactive paper infrastructure there is only one InputHandler which directly sends an HTTP request containing the positional information stored in the TimestampedLocation to the server as shown earlier in Figure 5.1. However, different input handlers are applied for information capture in some of the applications presented in Chapter 6. By defining all the input handling functionality in abstract interfaces, it is possible to support new types of input devices with a minimal amount of effort. It is also possible that the same InputHandler registers to different InputDeviceEventListeners and hence different technologies can be applied for input processing in parallel by the same application. For example, one user could control an application with the inductive Paper++ pen while, at the same time, another user interacts with the application using a Nokia Digital pen.

5.4 Address Space Mapping

We have seen how information captured by a specific input device is delivered to all input device event listeners and finally sent to the input handlers as timestamped location objects. The information captured by an input device has to be transformed to the address space of our interactive paper platform. Therefore, each InputDeviceEventListener has a transformation component taking positional information in the input device's native format and translating it to a TimestampedLocation.

As described earlier, the Paper++ encoding consists of a grid of barcodes encoding information about the page number and the (x,y) positions. While the page number can be decoded directly from the barcode detected by the inductive reader, the positional information has to be computed from information stored within the barcodes. A single barcode does not directly encode its (x,y) position in terms of millimetres as required by a `TimestampedLocation`. Each barcode only encodes information about its position within the grid covering the page in terms of a row and column number. For each document, we further have to define its size defined by the width and height in millimetres. Based on the dimension of a page and the number of rows and columns, the `TileMapper` that is used by the `InductivePenEventListener` calculates the size of a single tile and computes the exact position of each single barcode tile. The centre of a tile is returned as the (x,y) position of the reader which means that the resolution is limited by the size of the tiles. This positional information, in combination with the page number encoded within the barcode and the document identifier resolved from a barcode printed on the cover page or a separate RFID tag, is transformed to the required `TimestampedLocation` and delegated to the `InputHandler` for further processing.

The conductive barcode encoding that was applied in first demonstrator applications used tiles with a width of twelve millimetres and a height of seven millimetres resulting in a grid with the corresponding resolution on the horizontal and vertical axes. While such a relatively coarse resolution was appropriate for initial user studies on enhanced reading, it cannot be applied for applications that require a much finer resolution such as pen-based capture of handwritten information. We therefore decided to support digital pens based on the Anoto technology, providing a much higher resolution of 1/1000 mm, as an alternative input device.

As described earlier, the integration of a new input device such as the Nokia Digital Pen involves the implementation of a class handling the basic data exchange with the reading device. In addition, a new mapping component has to be implemented to transform the native encoding for a specific hardware solution to the address space of the interactive paper framework. In the following paragraphs, we discuss the mapping components shown in Figure 5.4 that were developed for a transformation from the address space used by the Anoto encoding to the timestamped location objects that can be processed by the interactive paper architecture.

The Paper++ pen prototype captures information based on measuring the inductivity of an encoded document whereas the Anoto input devices

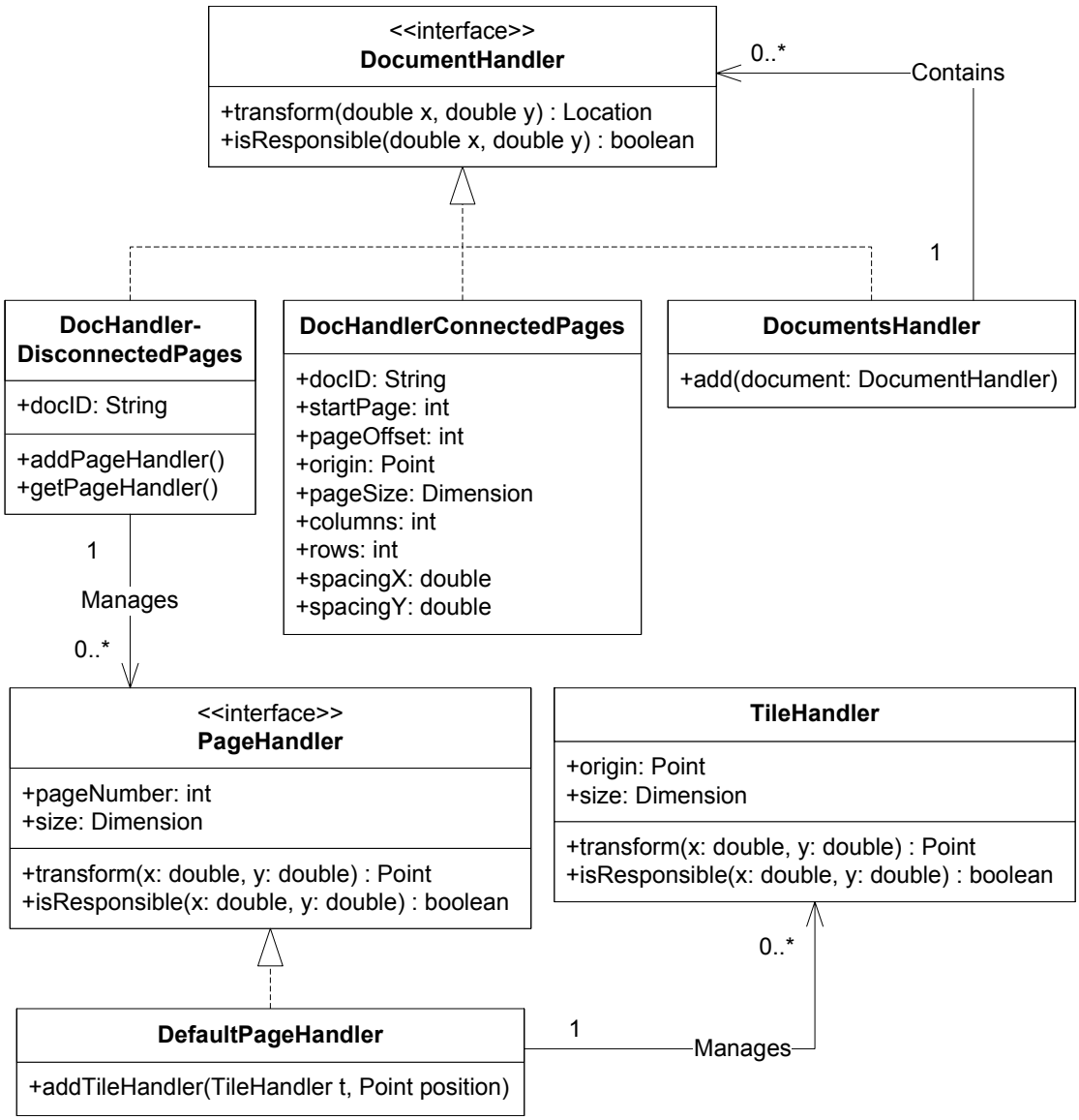


Figure 5.4: Address space mapping

use optical recognition to process the information stored on a paper document. While the different hardware approaches have only a minor influence on the specific software implementation of the input device interface, variations in the information encoding have a major impact. An important difference of the address space used by the Anoto technology is that, in addition to the non-existing document identifiers, no information about different page numbers is directly encoded within the pattern. Furthermore, the pattern covers a single huge virtual page and the pen always returns its absolute position in Anoto grid units within this single virtual master page which has its origin in the upper left corner. Specific Anoto enabled documents then use different parts of this address space and assign it to a document identifier and page number.

The main functionality for transforming Anoto input to location instances handled by the interactive paper framework is defined by the `DocumentHandler` interface. A `DocumentHandler` processes any Anoto position information and returns the appropriate location object or `null` if the `DocumentHandler` is not responsible for the specific pattern. Again, by defining the `DocumentHandler` as an interface we have the flexibility to implement different approaches for mapping from the Anoto address space to that of the interactive paper platform. Furthermore, multiple documents may be combined and managed by a `DocumentsHandler`.

At the moment we support two different `DocumentHandlers`: the `DocHandlerDisconnectedPages` and the `DocHandlerConnectedPages`. The `DocHandlerDisconnectedPages` can be used to independently cover document pages with different parts of the Anoto pattern space. A page can either be entirely covered by the pattern or only parts of the page, i.e. isolated tiles, may be covered with Anoto pattern by registering the appropriate `TileHandlers` on a document page. To easily cover successive pages with parts of the Anoto address space we implemented the `DocHandlerConnectedPages` which is shown in Figure 5.5.

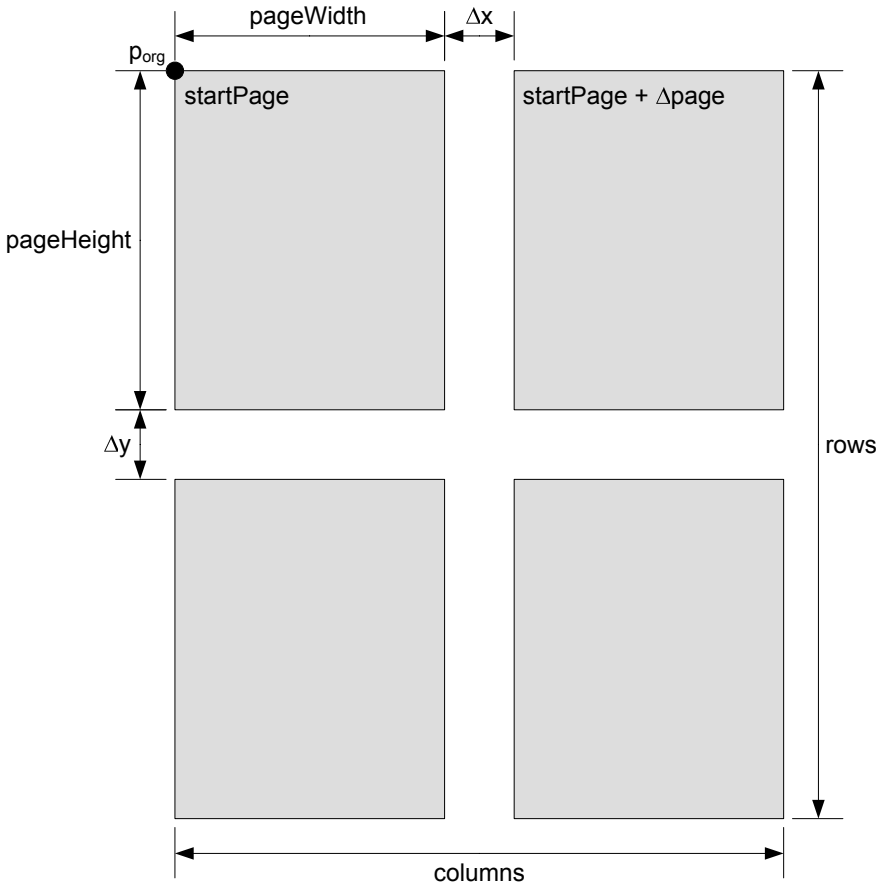


Figure 5.5: Connected pages

The idea is to layout successive pages defined in the augmented paper address space within the Anoto encoding space in a grid of a number of columns and a number of rows (pages ordered line by line, from left to right). Only the upper left corner of the first page (p_{org}) has to be registered with the corresponding position in the Anoto space. All the pages have the same width ($pageWidth$) and height ($pageHeight$). Optionally, a gap can be defined between pages in the grid on the horizontal as well as on the vertical axis represented by Δx and Δy, respectively.

Note that these gaps in the Anoto address space occur since, for each page, there is a reserved area to define special Anoto functionality such as paper-based buttons. In contrast to our solution, an application based on Anoto technology does not cover a page with a unique pattern but for each paper-based button a section of the specially reserved pattern is pasted over the base pattern covering the page. The Anoto solution not only requires additional pattern space for active areas but also puts constraints on the minimal distance between adjacent paper-based buttons.

Any position p_1 lying somewhere within the space covered by the matrix of document pages can be mapped automatically to the corresponding *page* and position p_3 of the interactive paper framework's address space with the following transformation algorithm where each point p_i is defined as (x_i, y_i).

TRANSFORM(p_1)
1 $(x_2, y_2) \leftarrow (x_1, y_1) - (x_{org}, y_{org})$
2 $totalWidth \leftarrow pageWidth + \Delta x$
3 $totalHeight \leftarrow pageHeight + \Delta y$
4 $x_3 \leftarrow x_2 \bmod totalWidth$
5 $y_3 \leftarrow y_2 \bmod totalHeight$
6 $column \leftarrow x_2 \div totalWidth$
7 $row \leftarrow y_2 \div totalHeight$
8 $page \leftarrow (row * |columns|) + column + 1$
9 $page \leftarrow startPage + (page * \Delta page) - \Delta page$
10 **return** *page* and p_3

First, we have to translate the (x,y) coordinates to get a new position p_2 which is relative to the upper left corner of the page matrix by subtracting p_{org} from the position detected by the pen. The position p_3 relative to a single page can then be computed by applying a modulo operator with the new position p_2 and the total page width and height as arguments. Finally, we get the page number by computing the

corresponding row and column of the page matrix. The page and positional information is then returned and augmented with the document identifier for further processing by the interactive paper framework.

Note that we reach high flexibility for supporting new types of input devices by defining the relevant interfaces. In addition, by implementing the entire reader logic and address space mapping components in the Java programming language, it is easily possible to run the input processing client software on any device providing a Java virtual machine.

5.5 Resource Binding

An example of an interactive page is shown in Figure 5.6. The page on the left hand side is covered with invisible barcodes that encode document positions in terms of coordinates. Logical elements within the page can be made selectable by defining corresponding logical shapes describing a selectable area of a page. The digital pen returns an (x,y) coordinate and the transformation component maps this positional information to the corresponding shape. Each shape is associated with one or more digital or physical objects which are returned as the result of the pen request.

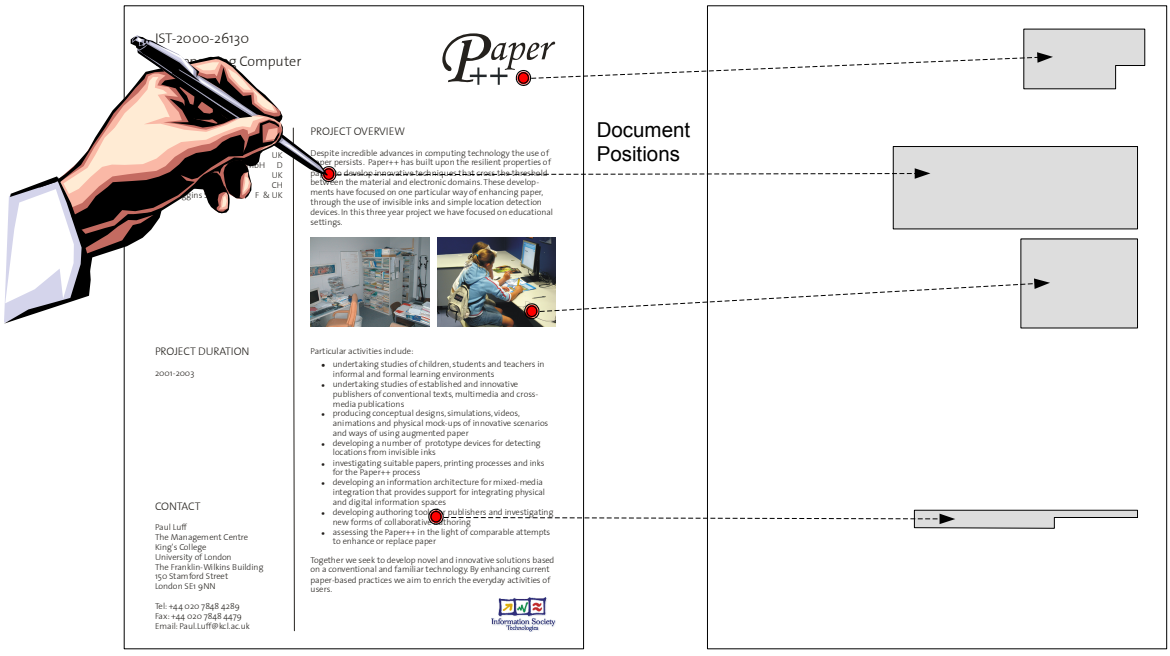


Figure 5.6: Mapping of page positions to shapes

Figure 5.6 shows a page containing a title, two images, multiple text paragraphs and a bullet list. The author of the interactive paper document has specified four selectable elements which are the title, a text

paragraph, one of the two images and a single bullet point. Each selectable element is specified by a shape as indicated on the right hand side of the figure. Shapes may be of various forms such as rectangles, polygons or complex shapes which are a composition of multiple simple or complex shapes. Each of these shapes is then associated to one or more digital or physical objects. For example, the image's shape may be linked to a project database object providing information about the project setup shown in the picture or possible further digital multimedia resources such as images, videos or sound.

To bridge physical paper and the digital medium, the transformation model defines how a document's meta information is represented and stored in the iPaper plug-in. While a UML diagram of the necessary Java classes was presented in the previous chapter when discussing the iPaper plug-in, in Figure 5.7 we show the necessary extensions in an OM model. The actual link source or target anchors are defined by `shapes` such as `rectangles`, `ellipses`, `circles` or `polygons`. The transformation data is used to check whether, for a specific position on a document's page, there exists one or more anchors, i.e. it allows the system to test if a point lies within one or more shapes. Finally, every anchor is associated with a digital or physical piece of information.

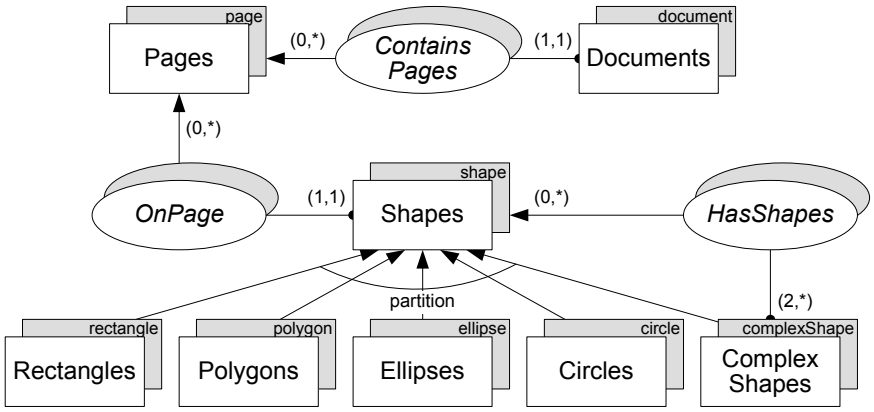


Figure 5.7: Transformation model

Figure 5.7 only shows the iPaper specific information concepts. However, the `page` represents a specific implementation of the general iServer `resource` whereas the `shape` is the appropriate implementation of an iServer `selector`. An integrated version of the iServer model and iPaper plug-in is presented in Figure 5.8.

In addition to specific implementations for the `document`, `page`, and various `shape` classes, the `IPaper` API shown in Figure 5.9 provides the

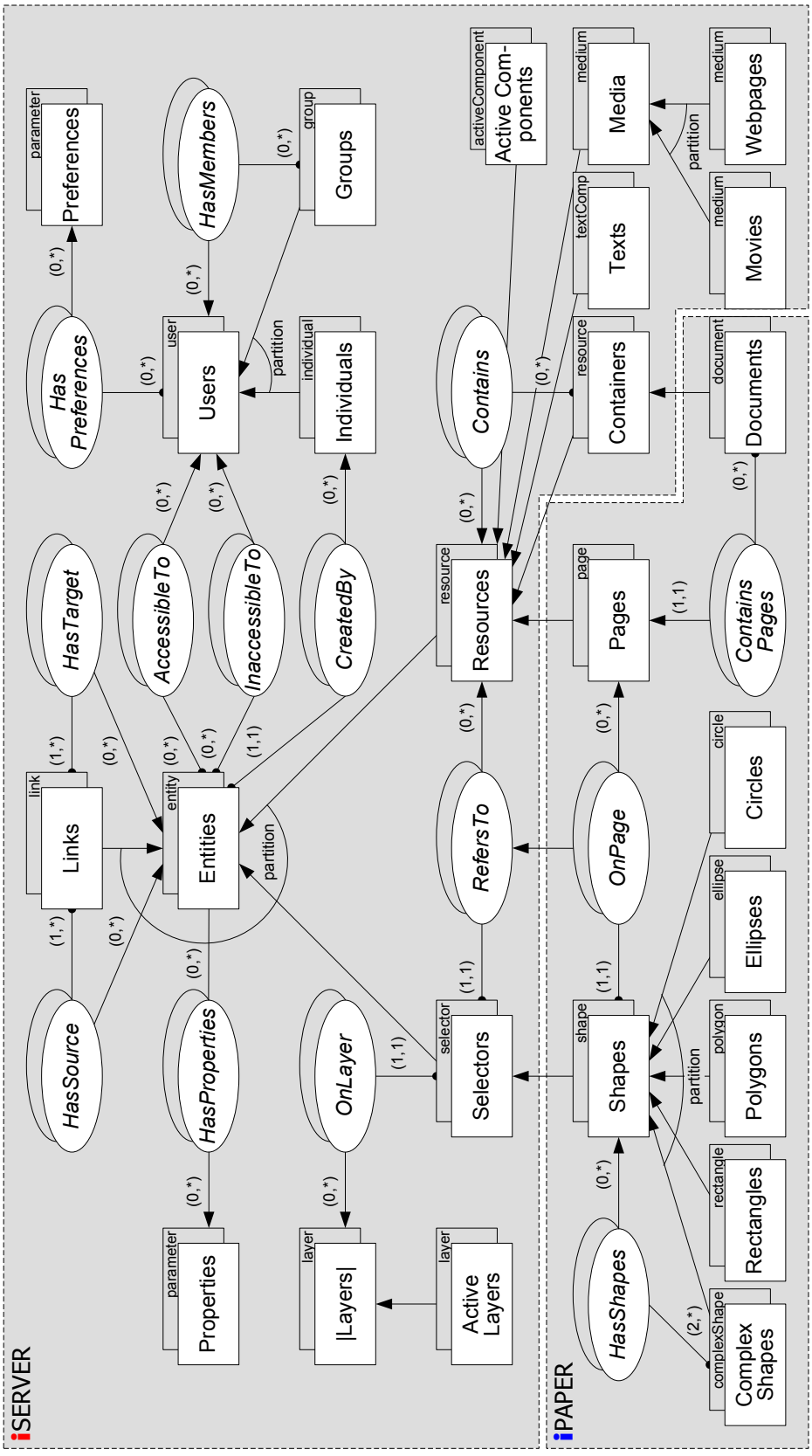


Figure 5.8: iServer with iPaper plug-in

necessary functionality to create instances of the new resource and selector types in a way similar to the `IServer` API presented earlier.

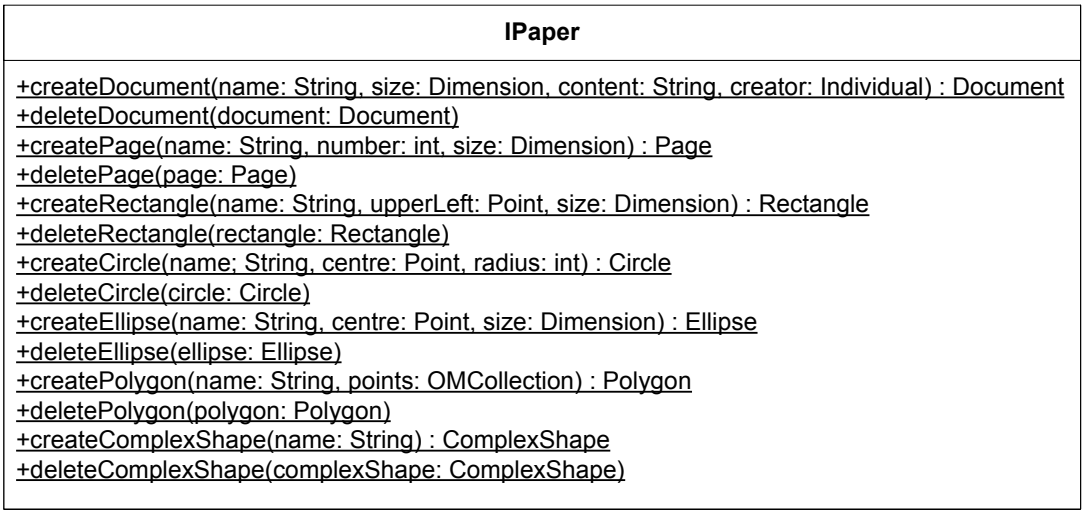


Figure 5.9: iPaper API

Since each `shape` is a specialisation of the iServer `selector` concept, it is also associated with a specific logical layer. There can be no overlapping shapes within a single layer. As a consequence of this, overlapping shapes must belong to different page layers. In the case that the selected position lies within two or more shapes, the shape within the uppermost layer is selected. In addition, specific layers may be activated and deactivated which enables us to generate context dependent results by binding a particular position on a page to different resources according to the current set of active layers. For example, this could be used to provide a "zoom in" effect on images if we were to define the shapes and layers such that repeated selection of a position on a page caused the uppermost layers to be deactivated in turn, thereby moving down to smaller image parts defined on the lower levels.

The zoom in/out functionality entails a whole range of new interaction and user interface possibilities in working with static paper, where the digital media can be used to give feedback about the context represented for example by the currently active layer within an image. The link from a single shape to the corresponding resource is not shown in Figure 5.7 since it is managed by iServer. Note that the layers can also be used to deploy personalised content to different users, for example by assigning a specific range of layers to every user and only activating the user specific layers.

Virtual page layers provide a whole new set of interaction possibilities for paper interfaces, where digital media can be used to give information about the current layer context. It is a concept that offers much flexibility, but will require significant investigation to establish guidelines for the use of layering in order to avoid users becoming lost in their navigation of these virtual layers placed over physical documents. In the future, a possibility to overcome this loss of context awareness could be to have a paper area with limited display functionality based on technologies described in Section 2.2.5. This electronic paper area could then provide feedback about active layers and other contextual information. Another possibility would be to add more functionality to the digital pens, e.g. a small display integrated in the pen, which again could be applied for delivering contextual information.

5.6 Summary

In this chapter we presented the requirements for a true integration of paper and digital media in terms of the underlying information system. After presenting the overall software architecture for interactive paper and introducing the different system components, we pointed out the different tasks that have to be performed on the client side before a request can be sent to the server. The reader and address space mapping interfaces have been highlighted and two different address space mappings for an inductive pen as well as for the commercially available Nokia Digital Pen have been presented.

While the focus of this chapter has been on the various components which were necessary to build the interactive paper architecture and the flexible integration of different hardware solutions, the next chapter introduces various applications that actually have been realised based on the interactive paper platform.

In theory, there is no difference between theory and practice. But, in practice, there is.

Jan L. A. van de Snepscheut

6 Applications

So far we have presented a cross-media information model, introduced the iServer cross-media platform as an implementation of this model and discussed the interactive paper plug-in for iServer in detail. In this chapter, we evaluate the presented architecture and illustrate the potential of tightly integrating paper and cross-media information spaces by introducing various applications that have been implemented based on the interactive paper framework. Each application focuses on specific issues of integrating paper and digital media, thereby supporting the process of evaluating different parts of the interactive paper platform. Various user trials helped in validating new concepts introduced by our information server for interactive paper and further provided valuable input for incrementally refining the iServer platform as well as the iPaper resource plug-in.

The applications presented in this chapter not only focus on different aspects of augmenting paper with digital information, but are also based on different hardware solutions. As described in Chapter 5, we decided to make our interactive paper platform independent of any specific hardware solution and rather make it extensible in a way that different approaches for defining physical link sources can be easily integrated.

When we started implementing the interactive paper platform, some mechanism was required that would enable us to implement first demonstrator applications in parallel to the design and testing of the paper, printing and digital pen technologies. We decided to use off-the-shelf

barcode readers and conventional visible barcode technologies in an early phase of the project when the required hardware was not yet available. However, our approach used existing barcode technologies in a rather unconventional way in order that we could simulate the technologies under development. We did this by using barcodes to encode positional information rather than applying a simple mapping approach based on unique object identifiers. For each active area defined on a paper document, we also created a barcode encoding a single position within that area and positioned it in the document alongside that area. By applying this technique, the information delivered by reading a barcode is equivalent to the information delivered by the various interactive paper technologies when any position within an active area is selected. An example of a document page which has been augmented with this barcode-based approach is shown in Figure 6.1. While this simulation using regular barcodes was sufficient for initial interactive paper applications, already this simple example illustrates the validity of our statement that barcode-based approaches for augmenting paper do not scale well and are not feasible if a large number of links are to be defined on a single page. In Figure 6.1 specific objects on a timeline, represented by the circular shapes, have been digitally augmented, and a barcode has been placed next to the object as described before. The resulting artwork is disrupted significantly by the visually obtrusive barcodes and one can imagine that it gets even worse when more objects are to be linked.

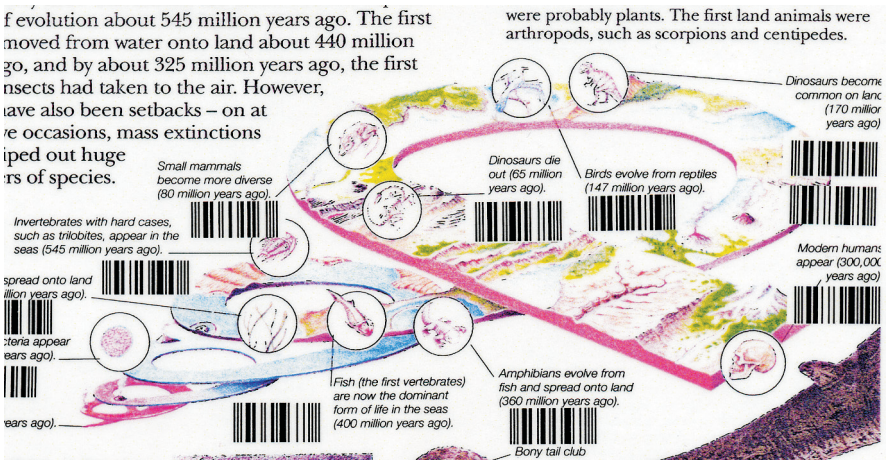


Figure 6.1: Barcode-based simulation

As soon as the first pen prototypes developed by our Paper++ project partners were available, we could use the very same applications with the new input devices. No modification to the stored link data was necessary since all the links were defined on active areas and resolved based

on positional information delivered by the pen device. The only necessary effort was to provide a new implementation for the general reader interface described in Section 5.3 that maps from the device's coordinate space to coordinates required by the interactive paper platform.

More recently, after the completion of the Paper++ project, additional applications have been developed within our research group based on iServer and its interactive paper plug-in. However, these applications are no longer based on the inductive pen hardware that was developed within the Paper++ project, but on the commercially available Anoto technology. The Anoto technology provides better resolution and performance than the inductive pen prototype. This becomes important if one wants to support not only enhanced reading but also enhanced writing as required by some of our more recent applications. Further, it is easier to print the Anoto pattern than the conductive Paper++ pattern since regular laser printers with an appropriate resolution can be used.

6.1 Nature Encyclopaedia

The very first application that was developed based on the cross-media information platform dealt with a children's nature encyclopaedia published in book form by Dorling Kindersley [84]. In addition to the book, there also exists a software application delivered on a CD-ROM which covers the same topics as the book version of the encyclopaedia. Discussions with the publisher and examination of the material revealed that both the book and the CD-ROM were not only marketed as independent products, but also designed and developed by separate departments with little or no cooperation and as a result contained quite different content. The goal of the Nature Encyclopaedia application was to take the existing physical and digital content and create a new integrated version of the encyclopaedia making use of interactive paper by linking parts of the book to pieces of digital information from the CD-ROM.

For this first application, we did not provide the full potential of the underlying iServer architecture to the user. A very simple and controlled setup was chosen for initial user studies. Only paper-to-digital links were available and, after a link to the digital media had been activated, there was no possibility to follow further links in the digital media. These restrictions were enforced to prevent users from switching completely from the physical to digital media, thereby abandoning the book version of the encyclopaedia.

A more technical goal for the nature encyclopaedia application was to have the application running on different output devices. As described earlier, iServer uses the eXtensible Information Management Architecture to deploy information on different output channels. Based on XIMA, two different user interfaces, one for desktop computers and a second one for handheld computers with limited screen size, were designed for the Nature Encyclopaedia application. Figure 6.2 shows the Nature Encyclopaedia application running on a Laptop Computer and on an iPAQ 3660 Pocket Computer. It was easily possible to run the Java code of the iPaper client application on a limited Java virtual machine installed on the Pocket Computer. The communication of the Java application running on the iPAQ with the pen input device was a bit more difficult since special drivers for the serial port communication had to be installed.

(a) Laptop (b) iPAQ Pocket PC

Figure 6.2: Client devices

Three different versions of the encyclopaedia — book only, a book together with CD-ROM and the integrated interactive paper version — were compared in a user study where groups of children were given specific tasks to solve using only one of the three versions. Details of this evaluation carried out by some of our project partners are presented in [171].

A first outcome of these initial user studies was the insight that the system lacked some form of feedback after the processing of a pen position had been initiated. It could take up to two seconds to process the pen data, send a request to the remote server, look up the linked information

and transform it to the appropriate output format before sending it back to the client device. Since users were not aware that the system was already processing a request, in many cases they repeated the selection activity — either of the same or another object — and this could lead to confusing feedback. Based on these observations, an acoustic feedback component was added to the interactive paper framework. When a new position is detected by the pen, the client device immediately plays a sound and, at the same time, sends the request to the server. With this simple acoustic feedback, users become aware that their selection has been detected and is currently being processed by the server.

Another result of the user studies was that the screen-based user interface is too restrictive in not allowing users to interact with activated link targets. The users reported that for movies and sounds some kind of paper- or screen-based controls, e.g replay or volume control functionality, should be provided to allow direct interaction with the played resource. Further, users wanted to have a possibility to navigate in the digital media similar to the Web or other multimedia interfaces. The HTML page resulting from a specific pen interaction should contain links to further related material as indicated in Figure 6.3 in the form of the hypermedia documents.

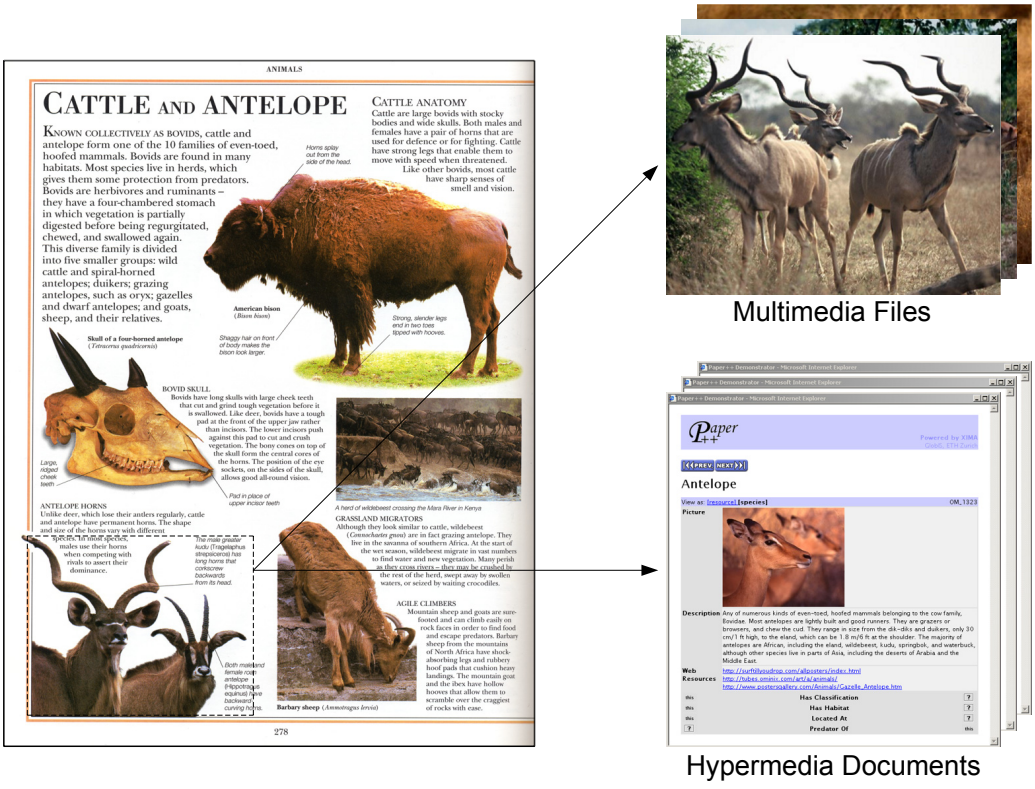


Figure 6.3: Content types

As mentioned earlier, the screen-based interface was deliberately implemented without any controls and links for further navigation in the digital media, based on the requirements of our project partners who were in charge of the user studies. However, the underlying information model for the Nature Encyclopaedia was designed with many associations between different information entities and restrictions were introduced on the interface layer only. By implementing a domain-specific nature database, we could not only link to simple media files, but also integrate semantically rich database objects. These objects could store metadata about simple media files and, through associations to other information objects, could support deeper levels of linking. The resulting schema of the nature database is presented in Figure 6.4.

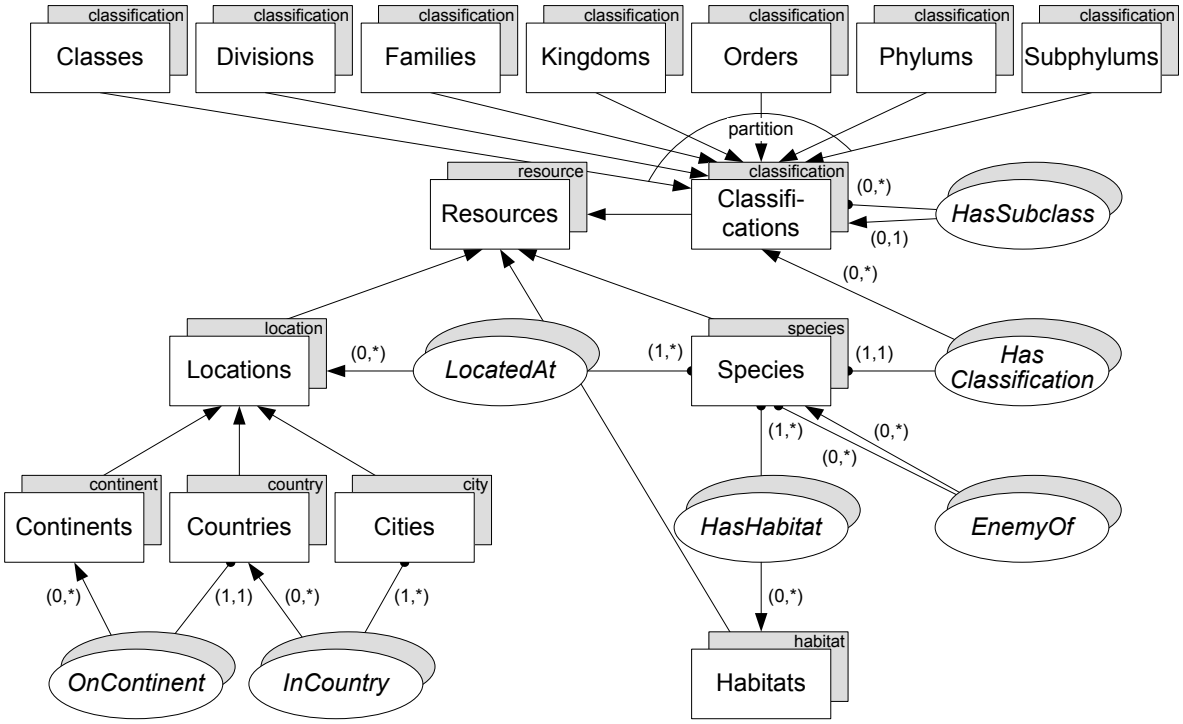


Figure 6.4: Nature database schema

To show the advantage of using a domain-specific application database, let us assume that we want to link the image of the antelope situated in the lower left part of the booklet shown in Figure 6.3 to a digital image of an antelope. Instead of mapping directly to the `antelope.jpg` file, we map to an instance of objects of type `species` that defines information such as the species name, average size, description and habitat. Not only can we display all, or some subset of, this information in addition to the image, but also links to associated information objects such as predator species or a species habitat. In other words, we can generate fully

hyperlinked web documents directly from the database content. It is important to note that with this general solution, we can support all forms of linking between active areas and digital resources — whether these are simple media files or hypermedia documents as discussed in this example.

6.2 Interactive Worksheet

An interactive worksheet based on the interactive paper technology developed in Paper++ was designed for the Natural History Museum in London. Groups of two or three children first visited two galleries in the museum dealing with the topics of mammals and insects. Different questions had to be answered while visiting the galleries using the paper worksheet and a normal pen. A single paper worksheet was handed to each group and was collaboratively processed by the group members. While one child was reading a question, another one, or even several group members, could simultaneously write on the worksheet.

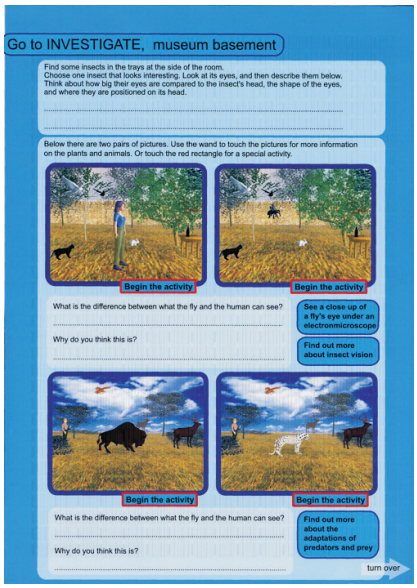
Figure 6.5: Interactive paper workplace

After visiting the two galleries, the groups spent some time in the museum's *Investigate Room*, a room containing computers, microscopes and other investigative devices, where we had installed a special place equipped with the interactive paper technology. The same paper worksheet that was used in the galleries could now be applied in combination with the inductive pen to access supplementary information and thereby answer any open questions. Two children working at the interactive paper workplace are shown in Figure 6.5. Note that next to the digital

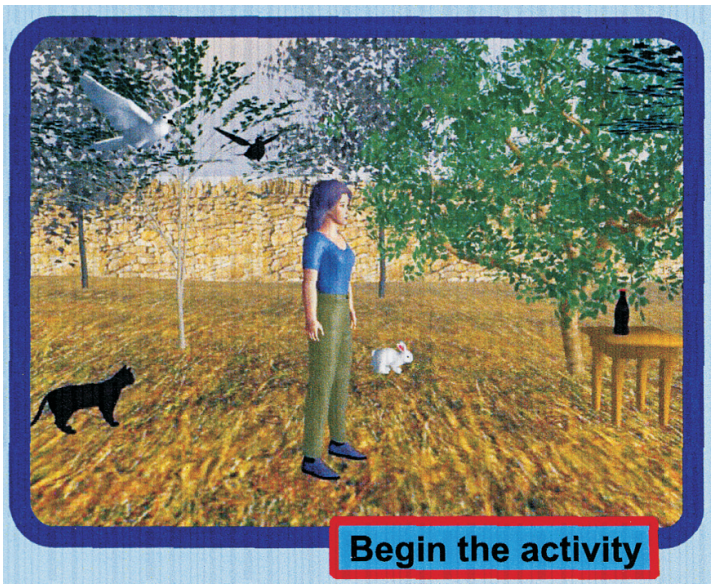
pen there is also a computer mouse on the desk, which could be used to interact with parts of the screen-based user interface. In addition to the digital pen there were some regular pens on the desk for filling in the missing answers. More details about the user trials in the Natural History Museum are reported in [102].

A major goal of the interactive worksheet application was to show the benefit of, not only integrating paper with simple digital resources, but also providing paper-based links to other paper documents. As shown in Figure 6.5, next to the interactive paper worksheet there where other books on the desk. The worksheet contains links to relevant parts of these books. After selecting such a paper-to-paper link, a message was shown on the screen providing information about a book's title and the number of the page where supplementary information could be found in printed form.

A single page of the paper worksheet designed for the Natural History Museum by our Paper++ project partners is presented in Figure 6.6(a). The page contains different kinds of information such as pieces of text, textboxes and four large images. The users can access additional digital information in the form of web pages or short flash movies by pointing to specific parts of the text or some of the textboxes. In addition to this browsing functionality, the four images build the paper interface for some interactive games. A detailed version of one of the four images is shown in Figure 6.6(b).



(a) Single page



(b) Paper-based game

Figure 6.6: Interactive paper worksheet

In addition to linking digital content in the form of simple media types or semantically rich database objects, the interactive worksheet provides paper-based user interface functionality to play different games. In this way, paper becomes a more active component and a primary control device for specific tasks in the digital world. Furthermore, in the paper-based application for the Natural History Museum, we investigated the potential of multi-layered links and their dynamic activation and deactivation based on an application's current state. Four active components were implemented in the form of simple games. They make extensive use of multi-layered links and apply the pen's input for game control. In the following, we address some of the games' functionality and outline how the active component concept was applied in providing this functionality. We first describe the requested interaction with the paper worksheet and then provide some implementation details.

An innovative aspect of the paper-based games is not only the fact that paper and some digital functionality are combined to build a multimodal user interface, but that the delivered information depends on the application's current state. Information bound to the four images can be accessed in two different modes. When a user starts to interact with the worksheet, it is in the so-called *information mode*. By selecting different objects within the image — for example the person, the cat or another animal — a user can access supplementary information about the selected object in terms of a digital image combined with some text, a movie or some other form of digital media.

By selecting the 'Begin the activity' button printed on the paper worksheet, the user can switch to *game mode* and activate a game. He first gets some audio instructions about the goal of the interactive game and is then requested to further interact with the paper-based user interface. In game mode, detailed information for specific objects, such as the cat, is no longer available. The whole image becomes an input area for a game about the vision of different species, where a user's task is to select all the surrounding objects that the creature in the centre of the image, which happens to be a person in Figure 6.6(b), can see. This means that if the game mode has been activated and the user points to the cat standing behind the person, they get visual and acoustic feedback informing them that an incorrect object has been selected since the person cannot see the cat.

The game runs for a specific amount of time, accompanied by a background sound and, after overall feedback about a user's performance, the state of the application changes back to the information mode. In the

case that a user missed too many objects or selected some incorrect objects, they are asked to repeat the game. However, if they managed to get everything right, thereby proving their knowledge about the creature's vision, they are requested to use a regular pen to highlight the creature's field of vision. By highlighting an area with the regular pen, knowledge which has been acquired through playing the interactive digital game finally becomes recorded on the paper worksheet. Note that through this process, the digital media is not only augmenting the physical paper, but paper is used to record the result of some digital interaction.

Let us now examine how the concept of multi-layered links were used to distinguish different application modes. After a game has been activated, the information layers are deactivated and active areas bound to these layers are no longer available. Only 2 active areas for each game in the form of shapes are defined in this game mode. The first shape is located on the so-called `Game` layer and covers the entire rectangular area of the image. Figure 6.7(a) shows the printed artwork, together with the active areas which are defined as visualised by the digital authoring tool which is presented in more detail in the next chapter. In this figure we can see the blue rectangular shape on the `Game` layer covering the entire image. The second one shown in Figure 6.7(a) is on the `Game Correct` layer and defines the central creature's effective field of vision by a triangular shape.

(a) Game mode layers (b) Information mode layers

Figure 6.7: Game and information mode layers

The `Game Correct` layer is positioned on top of the `Game` layer and therefore each time the user selects a position within the creature's view as defined by the appropriate shape, the link for this shape will be

activated. The game components are active objects which become instantiated for a specific amount of time. They process each selection of the user and keep track of the number of correct and incorrect objects a user has selected. Further, the games provide immediate visual and acoustic feedback to inform a user as to whether his selection was correct or not. At the end of a predefined game period, the game component provides overall feedback based on the data gathered during the game and informs the user if they were successful or should play the game once more.

It is important to note that while the application is in game mode, a user can no longer access the information about single objects such as the cat. However, as soon as the game is finished, the game layers become deactivated and the information layers are again activated as shown in Figure 6.7(b). It is then up to the user to access information about specific objects within the image or start another game.

The interactive worksheet application for the Natural History Museum is an effective application of our interactive paper software infrastructure's facility of being able to link, not only from paper to simple digital resources such as images or sounds but also to program logic. Active content becomes integrated in an interactive way in the form of the four games leading to a very tight integration of paper and digital information. In these games paper becomes an output device for memorising the results of a game. Further, the interactive paper worksheet demonstrates how the concept of multi-layered links can be applied for context-dependent information delivery based on an application's current state.

6.3 Annotation of Scientific Publications

While the Nature Encyclopaedia example as well as the interactive paper worksheets for the Natural History Museum are based on an educational setting for children, the next application is targeted at the research community. The publication annotation application was designed to support researchers in their annotations, recommendations and cross-referencing of articles.

The application domain chosen for this demonstrator was our own Paper++ project database providing information about related technologies, publications and contact persons. Some of the publications stored in the database were augmented in the printed version with links directly

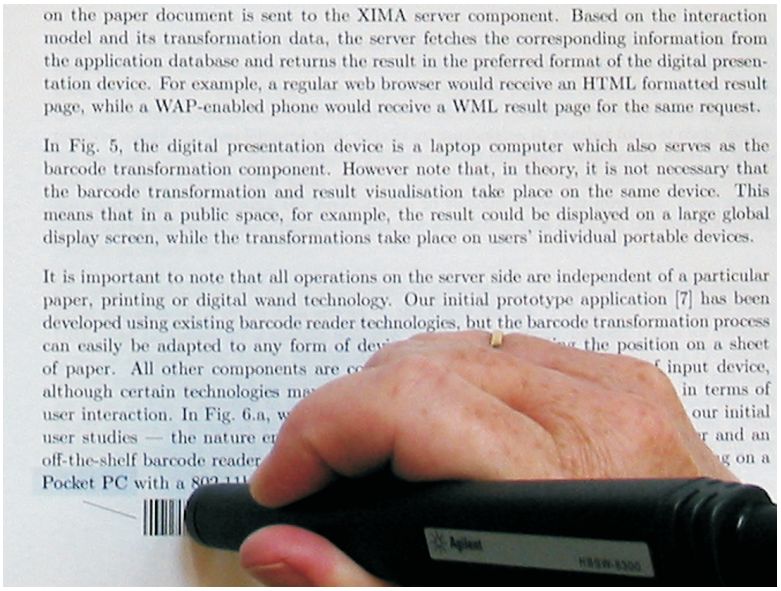


Figure 6.8: Annotated research papers

into the database. The prototype of the publication annotation application was another early application that used barcode technology to simulate interaction with digital pens. A publication augmented with digital information from the database is shown in Figure 6.8. After selecting the highlighted word 'Pocket PC', a user immediately gets detailed information displayed in a web page about the Pocket PC concept introduced in this part of the document together with links to various relevant websites and also lists of partners and publications based on this technology as illustrated in Figure 6.9.

What is the domain of discourse of such a publication annotation application? There are two possible answers — the specific research domain, e.g. interactive paper, or the general research activity of literature search and survey. Clearly, the former is much more specific and a system to support it requires a database about interactive paper representing domain-specific concepts such as digital pens and position encodings. The second is much more general and only requires a database that knows about concepts such as citations, references, annotations etc. which enables linking between different articles and information about the authors of those articles. While the data for this prototype is based on the interactive paper domain, the application's information model is a very general one representing concepts such as articles, authors and technologies. This means that the prototype is not limited to be used in a specific application domain such as the Paper++ project, but could be used by any scientific research community.

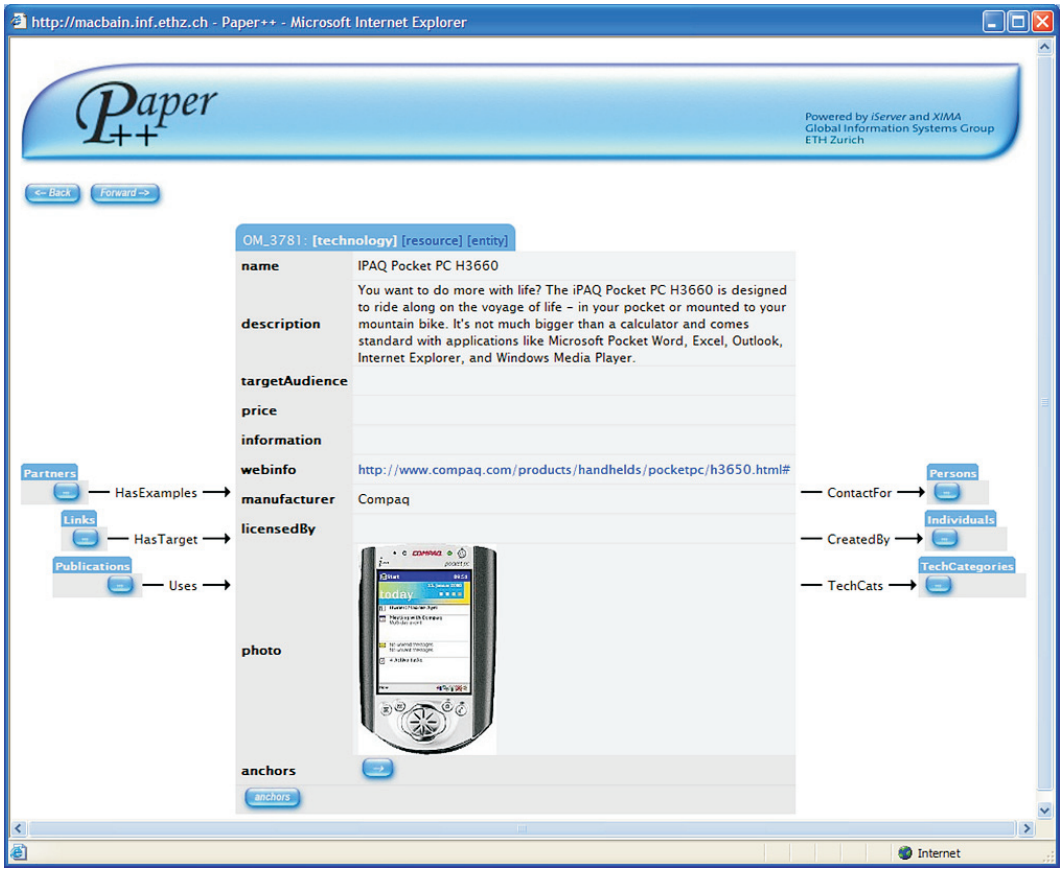


Figure 6.9: Result page with metadata

For the publication annotation application we made use of XIMA's capability to support multi-modal user interfaces. The simplest form of using the annotation application for digitally augmenting paper documents is to enable links from active regions to specific URLs. On link activation, a browser is started to visualise the appropriate information. Through this simple mechanism, it becomes possible to have *active citations*. A user no longer has to go to a library or even to turn to the bibliography at the end of the paper. Instead, by just pointing to a printed citation, they get direct access to the digital version of the cited paper which, if necessary, can then also be printed out.

An activated link may lead to a simple resource, for example a PDF version for a citation that has been selected. However, if a link has been defined to an object of the domain-specific application database, rather than just to a simple web resource, additional information can be accessed. In addition to the title and list of authors for the cited paper, through a dynamically generated web page one gets information about technologies that have been used in the cited paper, other papers where the same technologies have been used etc. Thus this application

provides the digital browsing functionality that was missing in the Nature Encyclopaedia application, thereby providing users with greater freedom to navigate back and forth between the paper and digital world.

One problem of such a system for handling scientific annotations is that while reading a paper, a user always has to switch to the computer screen to view supplementary information. We therefore decided to design a multi-modal user interface by taking advantage of XIMA's power to generate voice output. In addition to returning the information about a citation's title and other related information in textual form, XIMA transforms this data to spoken output using text-to-speech (TTS) technology. This has the advantage that a user can access multiple channels at the same time for gathering information. After selecting an active area on a research paper, the user can go on reading while simultaneously listening to the voice information about the linked resource.

Even without adding any additional digital information and using the digital media solely for defining associations between research papers, the sharing of these links is a potent tool that somehow extends the very static concept of today's citations as used in research papers. In this way, new citations and links to related work can be added to a research paper after it has already been published. If a user later activates such an annotation, they either get access to digital versions of the associated documents or at least get information about the title, the authors and the publisher of an associated document. Note that by using the selector concept, it is not only possible to define entire documents as links targets but also to reference specific parts of a document. The activation of such a *paper-to-paper* link not only returns the title of a document which has been defined as a link target but also details about the page number and where on that page the relevant information can be found. Further, by using a single iServer database for multiple clients or applying the distributed iServer architecture, it becomes possible to collaboratively build up a literature database together with experts in the same or other research domains.

We have introduced the publication annotation application as a tool for augmenting research papers and building associations between them. However, the application domain is not limited to research papers only and can be used to build up a knowledge base for any application domain based on paper documents and digital information. By applying the distributed iServer approach, not only experts of the same research area can share their knowledge, but also different research domains can be cross-linked forming a global information space.

6.4 Mammography Screening

While the first three applications only supported paper-based browsing of information, also known as enhanced reading, we now present a scientific application dealing with enhanced writing in the form of pen-based capture of new information. Within the scientific domain, one frequently finds that paper forms are heavily used as a means of both collecting data and also reporting on the analysis of data. For example, in breast screening clinics, mammograms are analysed by experts referred to as *readers* to determine whether or not the patient should be recalled for further tests. For each patient, four mammograms are taken — two views of each breast — and the resulting film is analysed by a reader who reports the findings on a paper form. The form contains breast outlines corresponding to the four mammogram views and the reader will annotate these to indicate any abnormalities or special features and, generally, any reasons for their decision whether or not to recall the patient.

To investigate the approach of annotation and link authoring on paper, as well as the needs and possible benefits of cross-media annotation in scientific environments in general, we developed a prototype system to support the work of the mammography screening process. It uses an Anoto-based solution for the authoring of both links and annotation content as described in this section.

The X-ray images used in mammography are available on film and the readers use a paper form to report on the results. Some breast screening centres have already switched to digital technologies or are considering it. The X-ray images can be displayed on large computer screens and computers can be used to report on the X-ray images. However, a digitalisation of the entire process brings some problems. First, the clinics currently see the generation of digital versions of the mammograms as simply creating an extra step in the process, resulting in extra delays and costs. Second, it is much more comfortable and quicker for a reader to enter data on a paper form, especially the annotations. It is also interesting to see that readers will often process mammograms in batches, analysing a number of mammogram sets until a significant feature is detected and then filling in the forms for the whole batch.

The main advantage of a purely digital solution is that data is available in a digital form for querying, ubiquitous access and for archiving. Researchers are also working on automated analysis of the images. However, switching to digital media results in a significant change of the working process and it appears that it also brings disadvantages [65].

Therefore, in our mammography screening prototype, we use our interactive paper platform to digitally store and retrieve reports while interfering as little as possible with current work practices.

It is common in most clinics to have double readings to reduce the chances of missing a possible tumour. In many cases, the second reader will first analyse the mammograms and annotate the form before checking the analysis of the first reader. This gives them the chance to double check whether or not they have overlooked something. The ability to read the annotations of other readers also plays an important role in helping less experienced readers learn from more experienced ones. To better understand the needs and possible benefits that could be afforded by cross-media annotation in scientific environments, we have had many discussions with researchers who have studied the practices of various breast screening clinics, inclusive of their training processes. Details of these practices of breast screening clinics and their use of annotations are given in [64].

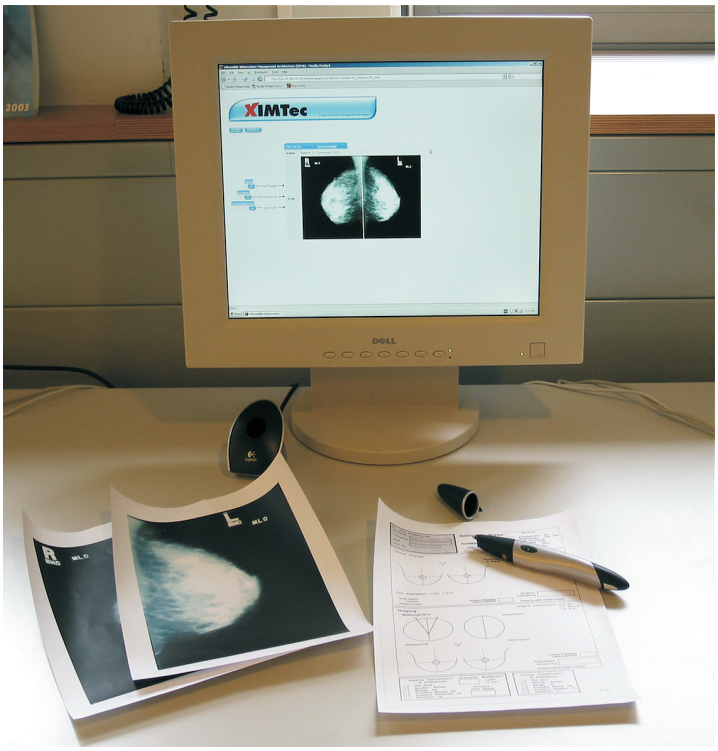


Figure 6.10: Mammography reading workplace

The mammography reading workplace based on our interactive paper technology is shown in Figure 6.10. A special application database has been designed for the mammography screening prototype based on an OMS Java database. This database contains information about patients

and screenings as well as the annotation forms. Furthermore, the database provides rich functionality and querying facilities for the stored data.

It is not convenient to print the Anoto dot pattern on each of the X-ray images for position detection. However, for the mammograms the *mimio Xi*, which captures handwriting based on high-resolution ultrasonic positioning, could be used as an input device without requiring changes to the X-ray film. Since we only have access to a limited number of X-ray images on film, we also used paper copies as substitutes in our prototype, applying Anoto technology for both the mammograms and the report forms. For each patient, the reader has multiple sheets of paper, covered with the Anoto pattern — the mammograms of the patient and a screening report form. The report form contains an active area linking to a patient's medical record.

To fill in the report form, the reader marks checkboxes and writes free text in the corresponding areas. The content of the checkboxes is analysed and stored in the corresponding database objects. The free-text annotations are stored as images and added to the database as an annotation of the mammography. The free-text annotations will only be interpreted by the other human readers of the mammogram. Therefore, there is no need to store the annotations in a machine-readable form.

If a doctor wants to annotate a specific area of a mammogram, they can use the pen to draw a shape around it before entering the annotation text on the report form. The system creates an active area on the mammogram in the form of a polygon and links this area to the digitised annotation text, thus creating a digital annotation to the paper document. This gives the readers the possibility of creating annotations on different levels of granularity. They can annotate the whole mammogram as well as very specific details referring to small parts of the X-ray images.

Existing annotations are available to the current users of the paper documents. The images corresponding to the annotations can be displayed on the computer screen along with additional information such as the creator of the annotation or the date. A smaller version of the mammogram is also displayed together with the active areas of the paper documents. However, in order to support blind double reading, i.e. independent analysis of the mammograms by different readers, annotations of other users are not visible by default. The readers perform their analysis and first create their annotations before consolidating their findings with those of the other readers. For controlling access to different annotations and comments based on a user's profile, the mammography

screening prototype accesses the user management functionality provided by the iServer platform. Since iServer defines access rights at the entity level, we have full control if access should be restricted to specific resources or single links and further can use the group concept for classifying different users.

The context-dependent visualisation of the annotations based on contextual information about the user is one of the biggest advantages of our prototype over the traditional non-digital solution. In current work practices, multiple versions of the paper form have to be accessed to support blind double reading. Another advantage is the instant availability of the annotations in a digital form. There is no need to scan or transcribe the report forms.

The mammography screening process is just one example of many where users are very reluctant to move away from paper. Sometimes it may be part of general resistance to change, but studies of the working environment and work practices often reveal that there are very good reasons for remaining with paper and other forms of non-digital media. On the other hand, if we consider the breast screening example, there are certain activities such as the reviewing of previous reports and annotations that can be improved with digital support. Summarising, we think it is important to investigate not only which parts of a work process can be digitised, but also which parts should be digitised in order to increase the effectiveness of work practices. In the end, an integrated solution where parts of the mammography screening application still remain paper-based but become tightly interweaved with a digital information system might be the appropriate approach.

6.5 Zurich City Guide

Mobile information systems require platforms that not only deal with the challenges of data distribution and dynamic networking, but also entirely new forms of interaction and information delivery. Ideally, users should receive *the right information at the right time and place* and in a way that does not restrict their mobility or their interaction with other people and their environment. This means that devices must be either wearable or very portable and easily placed in pockets when not in use.

If we consider the domain of tourism which has been a focus of several research projects in mobile information systems [139], many of these requirements are not met in terms of the devices and forms of interaction

provided. Tourism is generally a social activity and part of the enjoyment is planning activities together with family and friends. The screens of Pocket Computers that are often used in mobile applications are small and difficult to read in outdoor conditions, especially by more than one person at a time. Further, the small screen size limits the amount of information that can be viewed at one time and does not support the actions of comparing and combining information which is often what tourists want to do [24]. Some researchers have therefore experimented with Tablet Computers to provide better functionality. However, these further restrict mobility as they are much heavier than Pocket Computers and require the use of both hands.

The idea of the Zurich City Guide was to build a mobile application based on paper documents, a pervasive medium which is very portable and therefore well suited to mobile tourism. To gain first experiences in the area of tourist applications, we decided to focus on paper maps, only one of the many paper documents used by tourists, and investigate how they could be enhanced by using our interactive paper platform. Since different groups of tourists may have different interests, a goal was to develop a personalisable city map which could provide different information based on the very same paper map.

Therefore, a city map of Zurich was augmented with various links on three different information layers, a `Background` layer, a `Region` layer and a `Detail` layer, to provide supplementary tourist information about Zurich. Parts of the resulting interactive map are shown in Figure 6.11 together with the active areas defined on the different layers.

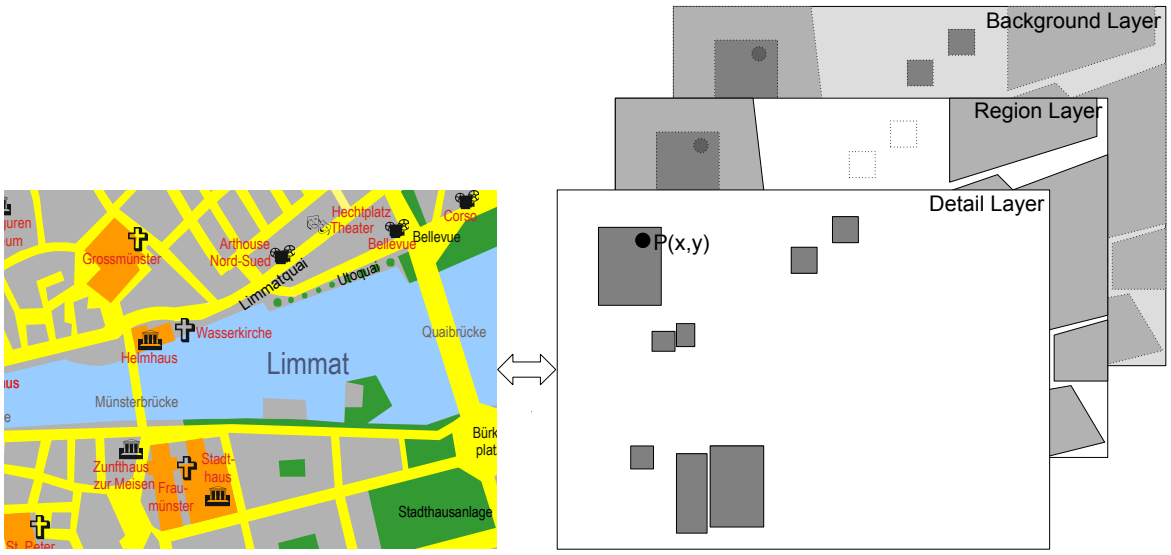


Figure 6.11: Zurich city guide

A single rectangular active area, covering the entire map and linking to some general information about Zurich, has been defined on the `Background` layer. By covering the whole map with this active area on the `Background` layer, we can guarantee that, in case no other information is linked to a user's selection, at least information associated with that active area is returned. The active area on the `Background` layer is linked to some general information about Zurich, which is always accessed in the case where no detailed information is available. The second layer, the `Region` layer, contains links to information about larger regions of the city whereas the topmost layer, the `Detail` layer, defines link anchors for information about specific buildings or places.

What happens if a user selects a position `P(x,y)` with the digital pen as shown in Figure 6.11? `P` lies within three different active areas defined on each of the three layers but the system will only resolve the link bound to the shape on the topmost layer, in our case the detail layer, and therefore returns the most specific information which happens to be some data about the Grossmünster cathedral.

To personalise a paper map, specific layers may be activated and deactivated based on a user's preferences. This activation and deactivation of layers enables us to deliver context-dependent annotations based on the current set of active layers. For example, by defining different layers for different activities/interests, such as shopping, eating etc., we can dynamically customise the city map by activating or deactivating some of these layers. Furthermore, by applying the zoom in/out functionality introduced in Chapter 5, repeated selection of the same map area can be used to deactivate the uppermost layers making accessible information linked to active areas defined on lower layers.

Mobile users are, not only limited in terms of the available computing power, but also in the amount of attention they are willing to pay to any digital devices. Most tourists prefer to visit a city with open eyes instead of focusing their eyes on the screen of a Pocket Computer all the time. For the Zurich City Guide, we therefore started to experiment with non-visual feedback in the form of spoken information generated based on XIMA's voice output facilities. This enables a user to interact with the city map by the means of both paper and voice recognition for input and voice feedback as an output channel. While the Zurich City Guide was limited to paper maps only, it helped in investigating the general field of information technologies and tourism and provided a foundation for a more complex tourism application that was developed for the Edinburgh festivals and is described in the next section.

6.6 Edinburgh Festivals Guide

Each year during the month of August, the city of Edinburgh hosts a number of international festivals, including an international arts festival, a book festival, a film festival and also the world's largest arts festival, the Edinburgh Festival Fringe, with more than 250 venues and 1700 shows, many of which take place daily throughout a four week period. These festivals bring a large number of tourists to Edinburgh who spend several days discovering the city, and its numerous bars and restaurants as well as visiting events.

During the festivals, the city is full of various forms of printed information including special venue maps, official brochures, daily programmes and event flyers. In addition, lots of information is available on official festival websites as well as websites of various newspapers.

There are many strong arguments for retaining paper in mobile environments, including the fact that it is light, cheap, robust and easily annotated in various ways [125]. Also, the planning of activities during a city visit often involves combining and comparing information within and across documents such as maps, event brochures and guidebooks and this is easier using paper documents than working with digital mobile devices with small screens. The Edinburgh festivals provide an ideal environment for testing technologies for mobile information systems and appropriate means of delivering relevant information in a timely and convenient manner. We therefore chose to investigate the use of emerging technologies for interactive paper in mobile tourist environments and particularly in the context of the Edinburgh festivals where tourists may want to enter and share reviews, as well as plan their activities, while on the move.

Based on the experiences already gained from the Zurich City Guide application presented in the previous section, we decided to build a mobile information system for the Edinburgh festivals based mainly on a paper user interface. The resulting *EdFest* system contains the interaction components shown in Figure 6.12, namely a special interactive paper brochure containing a map and event list, a digital pen and an earpiece with a built-in microphone used for voice interaction. We considered various options for the display of information and decided to dispense with any form of visual display such as a Pocket Computer, Tablet PC or head-mounted display and instead focus on audio output for the first demonstrator.

For the EdFest application we used the modified Nokia Digital Pen which works in streaming mode. A central server has a database with

Figure 6.12: EdFest interaction components

information about venues, events, pubs, restaurants and also user reviews. The brochure contains a map which is marked with venues and the user can request information about a venue by simply pointing with the pen at the appropriate location on the map. The system will then initiate a voice dialogue that allows a user to get general information about the venue or events being held there.

In addition to the interaction components, the users also carried a wearable computer with a Global Positioning System (GPS) module for location tracking, enabling the system to detect a user's location and support locator and navigation tasks [133]. For example, there is a 'Where Am I?' button located at the bottom of the map. The system helps the user locate their position on the map by telling them the general grid position, together with a general guide to the placement within the grid e.g. "Grid F5, top right". If the user then points with the pen within that grid, the system will give feedback telling them where to move the pen to arrive at the precise location, thereby helping users find exact location on a map which can often be a frustrating and time-consuming task. Users can also use this functionality to locate events

listed in the brochure by pointing to the venue and being told where to find that venue on the map. This is a form of paper-to-paper link where different parts of the physical booklet are linked together through digital media.

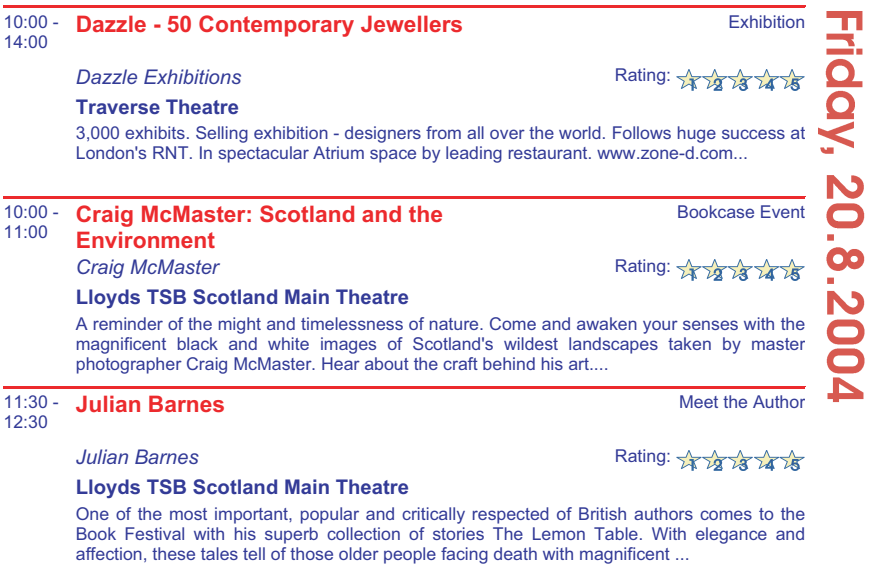


Figure 6.13: EdFest booklet page

Figure 6.13 shows parts of a booklet page listing different events. The user can access additional information about an event by simply pointing to relevant areas within the event listing. As described earlier, pointing to a venue, for example the Traverse Theatre, will result in audio instructions about where to find the venue on the map. The user can also get information about the artist, a description of the event, other events of the same category and ticket availability. In many cases, the choice of the exact type of information required is determined through a voice dialogue. There is also a rating area where users can input their rating by selecting a star rating between 1 and 5. The average rating is accessed by pointing to the text 'Rating'. It is also possible to set a reminder for a specific event by pointing to the event's timing information with the pen.

Last but not least, the users can enter their reviews either by writing comments alongside the event listing or by writing them in a separate notebook with the Anoto pattern. Notes that are written in a separate booklet can be linked to a specific event by selecting the event's title after the note has been captured. These reviews will be sent to the central database server and can then be accessed by other users requesting information about the corresponding event.

While iServer has been used to link paper and digital information, various other systems developed within our research group were necessary to implement the required EdFest functionality. These include a general web publishing platform OMSwe that is an object-oriented database management system extended to support web publishing by integrating a notion of web components, a context-dependent versioning mechanism [126] and a general context engine [15].

A system overview of the EdFest architecture [16] is given in Figure 6.14. It is based on a client-server infrastructure with iServer and OMSwe, the publishing framework, on the server side. The two server components manage all relevant information, whereas the client-side components provide the user interface as well as aggregate sensor information.

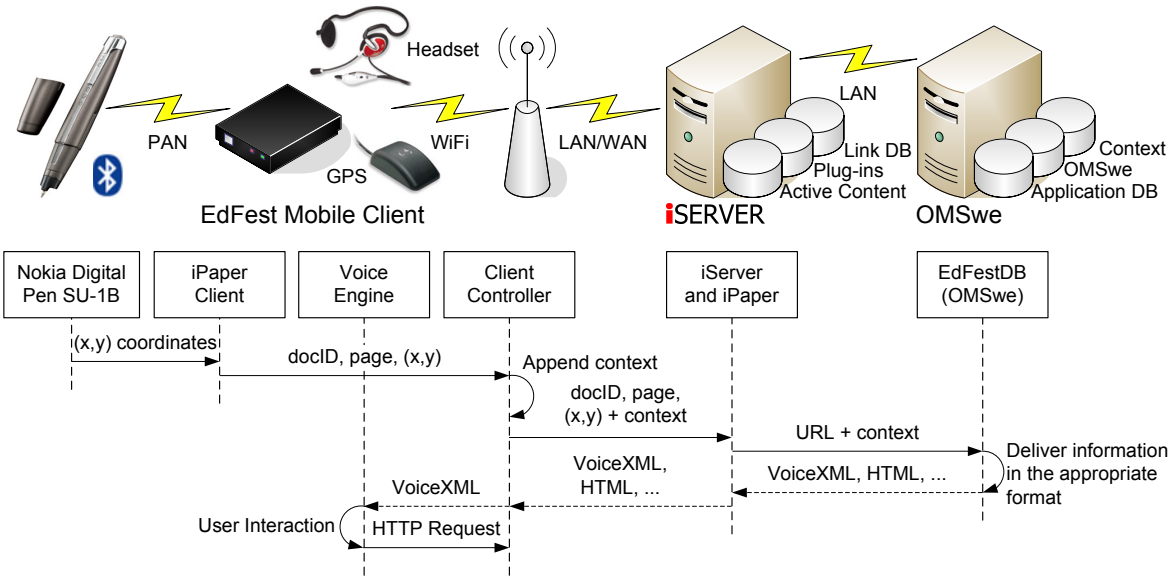


Figure 6.14: EdFest system architecture

As mentioned earlier, iServer with the iPaper plug-in mainly stores metadata about active regions within documents. Even though it is possible to manage application data within the iServer framework, for the EdFest architecture, we decided to have two separate databases: iServer to manage all the link metadata and an OMSwe application database to handle the actual festival data about venues, performances, actors and actresses, reviews etc.

As the user writes or points somewhere in the booklet with the Nokia Digital Pen, the iPaper Client sends a request to iServer. The iServer interactive paper plug-in resolves the positional information and retrieves the appropriate information. Most interaction is managed by active components and therefore in Figure 6.15 we present two scenarios where

active components have been used on the server as well as on the client side to manage complex operations within the EdFest prototype.

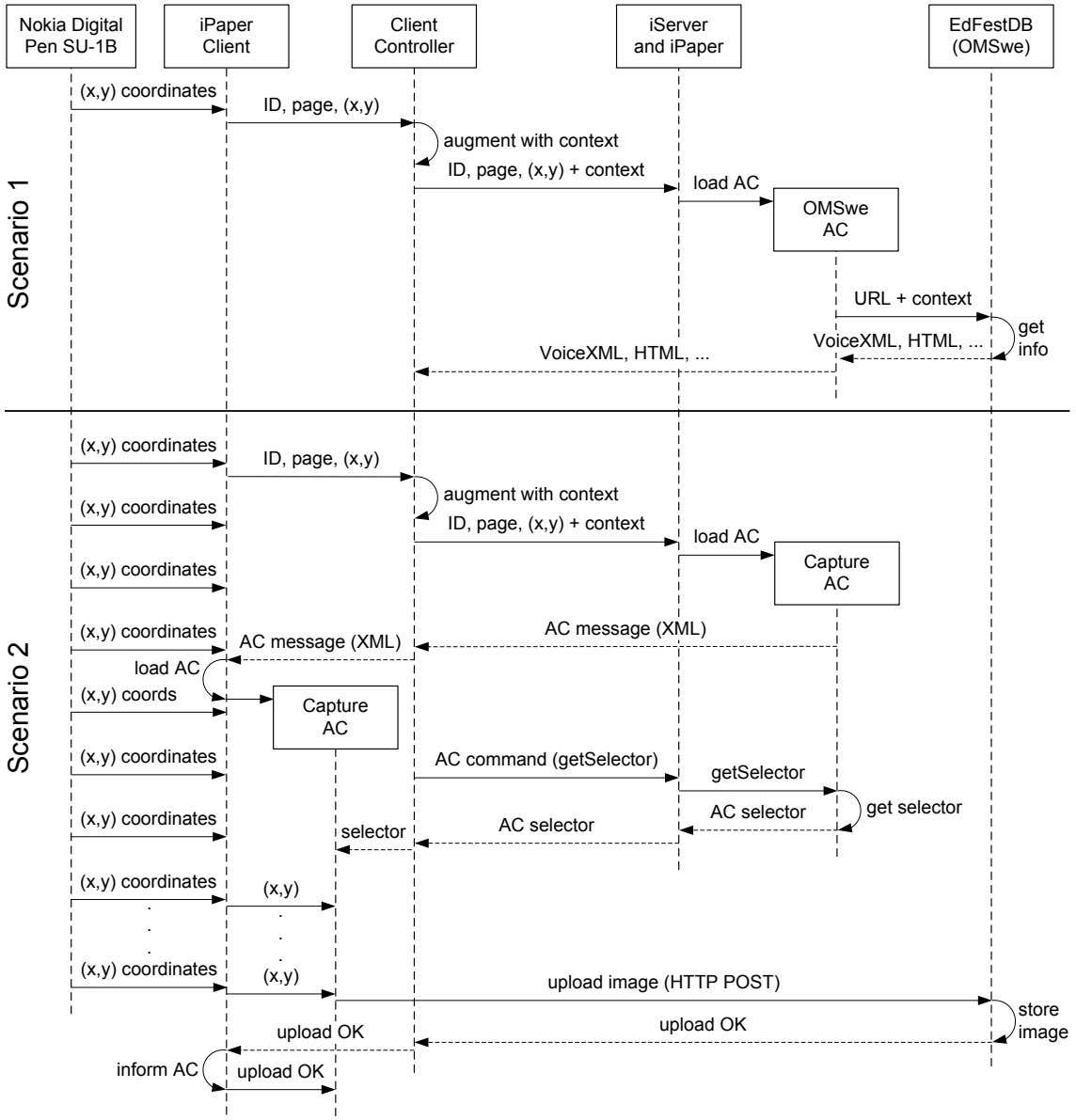


Figure 6.15: OMSwe and capture active components

The first active component that we present, the OMSwe component, runs on the server side and mainly acts as a proxy for information that is effectively stored in the external OMSwe application database. This active component is typically used when the user points somewhere in the EdFest booklet with the pen to get additional information in the form of audio output. The iPaper Client sends an HTTP request to iServer and the plug-in for interactive paper resolves the positional information to the appropriate target resource as shown in Scenario 1 of Figure 6.15. In the

case that the information is stored in the external EdFest database, the resolved resource will be an `OMSwe` active component. Based on the active component's identifier field, an object of the corresponding Java class is instantiated and initialised with the active component's supplementary data stored in the iServer database. In the case of an `OMSwe` component, this data includes a query encoded as an HTTP request that can be sent to the EdFest application database. Before this request is actually sent to the OMSwe server, it has to be augmented with the contextual information of the incoming iServer request that has been added by the Client Controller. After the query has been sent to the external OMSwe database, the `OMSwe` active component sends the response directly back to the Client Controller which dispatches it to the appropriate output channel. We have used a single database for managing information about the festival but the concept of an active server-side proxy component could be used to integrate various heterogeneous data sources.

The lower part of Figure 6.15 shows a second scenario, involving a client-side active component, where a note is captured based on the user's interaction with the festival booklet. If a user starts to write in an active area that has been defined as a capture area, first of all, the iPaper Client sends a single event to iServer as is normally done in browsing mode. The iServer paper plug-in performs a lookup for the specific pen position and returns a `Capture` active component. An instance of the `Capture` component is instantiated on the server side, based on the active component's configuration information stored in the iServer database as described in the previous scenario with the OMSwe proxy component. In the case of the `Capture` component, this information includes an upload address, i.e. a URL where the captured note finally should be uploaded, as well as a timeout parameter which is used for non-explicit termination of the capture process as described later. The active `Capture` component which has just been loaded on the server side, sends an XML message including its identifier (name) as well as various configuration parameters back to the iPaper Client. The MIME type of the HTTP response which is sent back is set to `application/ipaper.client` such that the Client Controller knows that it has to be dispatched to the iPaper Client.

The iPaper Client receives the XML message and identifies it as an active component response. An instance of the appropriate `Capture` active component stub is instantiated based on the identifier of the active component XML message and the additional information stored within the message. During this initialisation phase, the `Capture` active component has to obtain information about the active region, i.e. selector,

to which it is actually bound to. Therefore, the client-side active component sends a special `getSelector` active component command to the server-side active component which looks up this information and sends back a response containing the requested selector. Note that in the case where this information is only needed once during the initialisation phase of the client-side active component, it could always be directly integrated into the first active component message to reduce the number of requests and therefore improve the system's performance. When the client-side stub for the `Capture` component is created, it asks the Pen Client for the time when the last request was sent to the server, which is the time when the request for the `Capture` component itself was initiated. This information is used later to fetch the appropriate information from the input buffer. After the active component has been loaded, the pen switches from browsing mode to active mode which simply means that subsequent pen events are delegated to the client-side active component instead of directly being sent to the server. As explained before, the client-side `Capture` component requested information about the active region that has been defined as a capture area. The capture process is finished if the pen either leaves the active capture area or the predefined timeout, which is also a parameter of the `Capture` component, elapses. In the meantime, all pen events are stored in the input buffer. After the capture process has been terminated by one of the two possibilities just described, the `Capture` component does a lookup in the input buffer to get all positional information acquired during the capture process. This lookup is based on the temporal information that the active component requested in its initialisation phase. Finally, the captured information is sent to the predefined upload URL either as an XML document, a JPEG image or as a Scalable Vector Graphics (SVG). The OMSwe server sends a response message confirming that the upload of the image was successful which is delegated to the `Capture` active component. The active component informs the iPaper Client that it has finished its work. The iPaper Client immediately unloads the `Capture` active component and switches back to the default browsing mode waiting for new pen input to be handled.

The Client Controller, a proxy component designed for the EdFest prototype, is responsible for dispatching the response that is returned from iServer to the appropriate client component. For example, in the case of a voice response in the form of a VoiceXML document, the response is forwarded to a voice engine which generates the required audio output. In addition, an HTML interface is provided to display handwritten annotations and comments of other users.

Tests and user trials of the EdFest system took place in Edinburgh during August 2004. Usability trials were carried out during a three-day period at various locations in the city, including public places and locations in and around festival venues. They involved the testing of the EdFest prototype, a mix of semi-structured interviews, observations based on video and audio recordings and also user questionnaires. Overall the response to the interactive brochure was positive although there were a number of both design and technical issues raised.

A number of issues resulted from the fact that the physical environment was, of course, very different than our lab environment. The ambient noise in restaurants, pubs or public places outdoors, as well as the wind on the street often interfered with the voice interaction between the user and the system. In Figure 6.16(b) we see a user who tries to overcome this problem by shielding the microphone using his hand.

(a) Browsing (b) Voice interaction (c) Writing

Figure 6.16: User trials at the Edinburgh festivals

Other problems of the voice interaction revealed themselves in a social context. The system could not distinguish whether a user was actually using the system or just talking to a friend or one of the conductors of the user trials. A solution to this problem might be to let the user control when they are talking to the system, for instance by operating the volume control of the microphone and the earpiece.

Finally, we mention that users sometimes showed a certain reluctance to point with the pen in case they marked the brochure. We have found this to be a general problem associated with the dual mode of the modified pen which can act both as a selection and writing device. Ideally, the pen itself should have a mechanism to switch between modes, for example, by clicking a button at the back of the pen, it could retract the writing stylus and switch from writing to selection mode. We consider

such amendments to the design of digital pens essential if they are to become devices with this dual functionality.

In the interactive festival brochure that was used in the 2004 user trials, all texts were interactive and a user could access additional information by pointing to any printed part of the booklet. However, since paper-based access to digital information was new to most users, the user interface based on active parts of printed text was not clear to many of them. Therefore, for an improved version of the EdFest system that is going to be evaluated in August 2005, we decided that special pictograms, as shown in the new set of interactive festival documents presented in Figure 6.17, should be used to link any additional digital information. All active pictograms have the same shape and colour making it very clear to a user where they should point with the pen.

(a) Brochure (b) Map (c) Bookmark

Figure 6.17: Interactive festival documents

Another major change from the system used in 2004 to the current interactive festival guide in terms of the interaction design was the removal of the voice input channel. For many of the users in the 2004 trials it was unclear that they could also talk to the system in order to navigate through the voice dialogues. Even after learning that they had this possibility, some users went on using the pen for input, in parallel to the voice input. As a result of the negative feedback concerning the voice input channel, we decided to remove this functionality from the 2005 prototype and use voice only as an output channel and design the interface so that all interaction could be controlled from the paper documents.

The EdFest prototype has shown how iServer and the interactive paper plug-in make it possible to provide interfaces based on interactive paper for complex information systems alongside standard web-based interfaces. The concept of active components has proven to be an effective and flexible mechanism for designing and implementing interaction components based on configurations stored in the iServer database. Last but not least, the EdFest prototype introduces a completely novel form of defining links to parts of a paper document by applying the pen to find a specific location on a page based on continuous voice feedback. The resulting mechanism for finding locations on a paper map is a new form of linking from digital to physical material that could also be applied in other application domains.

6.7 PaperPoint

The PaperPoint application is a simple but very effective tool for giving PowerPoint presentations. The slide handouts are printed on Anoto paper together with some additional paper buttons for controlling the PowerPoint presentation. An example of such a slide handout is provided in Figure 6.18. A modified Nokia Digital Pen is used to remotely control the PowerPoint presentation over wireless Bluetooth technology.

Figure 6.18: Paper-driven presentations

The PaperPoint printouts of the slide handouts contain various buttons for interacting with the PowerPoint application. Below each slide

there is a 'Show' button which is used to switch to the corresponding slide in the digital presentation. Further, a user can point directly to a slide, which switches to the corresponding digital slide version, and annotate the digital version by writing on the printed slide. At the bottom of each 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.

A first evident benefit of the PaperPoint application is the fact that the presenter no longer has to stand next to the computer to control the slide presentation. The slides can be navigated remotely by pen and paper via a Bluetooth connection, which provides more freedom to the presenter for interacting with the audience. However, many other devices are available for controlling presentations remotely including wireless mice or even more specific presentation aids. While it is convenient to control the slide presentation remotely, this is not a unique feature of the paper-based user interface.

Compared to the conventional PowerPoint user interface, the paper-based PaperPoint control enables a more flexible way of giving presentations. The printed handouts provide a perfect overview over the available slides and, without switching to PowerPoint's slide sorter and interrupting the flow of the presentation, slides can be presented in a non-linear order. The presenter can jump from one slide to any other slide by selecting the 'Show' buttons or pointing directly to specific slides and the audience will not recognise at all that the slides are not presented in their original order. This is not only very convenient if some slides have to be skipped because of time constraints, but also for showing specific slides while answering questions coming from the audience.

A similar idea was investigated in the *Palette* project [120] which was discussed in the background chapter. The Palette project uses a single paper card for each slide. The default setting of the PaperPoint application is based on paper handouts containing six slides on each page. However, it is possible to adapt the system and print any number of slides on a single page or print each slide on a small paper card as was done in the Palette project. While the developers of Palette argue that it is very handy to have the possibility to annotate the paper cards and thereby write down comments or questions coming from the audience, we go a step further and provide the possibility to integrate handwritten information into the presentation in real-time.

If a presenter starts to annotate one of the slides on the paper handouts, the written information is digitised and immediately integrated into

the digital presentation by adding it to the appropriate slide. This functionality can, not only be used for highlighting existing parts of a slide, but also for spontaneously annotating a slide with textual information or diagrams. An example showing the highlighting of parts of a slide to emphasise important sections is shown in the lower left slide of Figure 6.18. The presenter highlighted the header of an XML document by circling it with the pen and drawing an arrow pointing to the text. Although this highlighting could also be directly done in the digital slide version, a major advantage of the paper-based solution is that it can be applied more easily than using a mouse. Note that of course it is also still possible to draw on slides using existing technologies, i.e. 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. Our PaperPoint application brings this functionality back to the appropriate tool for creating handwritten annotations — paper and pen.

Another very powerful feature of the PaperPoint application is the fact that it can be controlled by any number of digital pens and not only a single pen. By having multiple input devices, collaboration between multiple users is supported. 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 presentation. All annotations made on specific slides can be stored in a database and made available to other users which, for example, could be used by students for sharing notes that they have taken during a lecture. A student no longer has access to their personal physical annotations only, but can also use the handouts to retrieve comments from other students that have been stored in the database.

Especially in decision-making tasks, the collaborative presentation navigator can improve the overall performance by providing fast shared access to the relevant resources. If, for a moment, we forget about the content of a presentation, we get another application scenario where the PaperPoint tool can be applied in brainstorming sessions. Such a session can start either from an empty slide or be based on a slide containing some initial thoughts. A user can write down ideas on a paper document and share them with other users by projecting them on a screen.

The PaperPoint application has been realised mainly by implementing active components which interact with the PowerPoint application. To access the functionality of the PowerPoint Windows application from

Java, we used the Java/Win32 (Jawin) 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). PaperPoint focused on accessing and controlling PowerPoint functionality. The same approach could easily be applied to integrate any Windows application and it is even possible to access functionality from multiple applications based on the very same paper document, thereby integrating paper with digital functionality provided by different applications. While only a small subset of PowerPoint's functionality has actually been used by the PaperPoint navigation control to date, potentially any PowerPoint features could be accessed and integrated with the paper-based user interface.

6.8 The Lost Cosmonaut

The Lost Cosmonaut project is based on a collaboration between an artist and our research group investigating the application of the interactive paper platform in an artistic setting for interactive narratives and story writing [177]. The work is part of the *Artist in Labs* programme, a Swiss initiative trying to bring together artists and scientists.

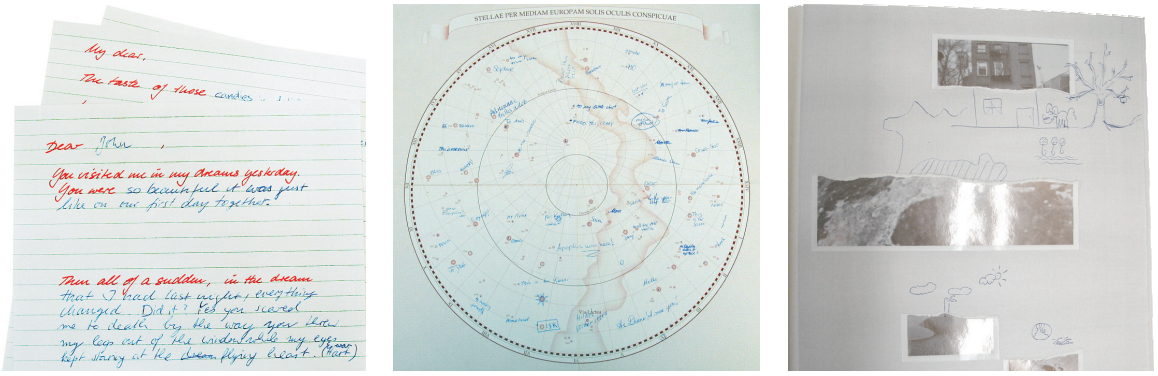
(a) Interactive desk (b) User interaction

Figure 6.19: Lost Cosmonaut installation

The setup of the Lost Cosmonaut installation is shown in Figure 6.19. A user sits in a fairly dark room in front of a semicircular desk. The wall in front of the user contains a large round hole which is used as a screen for projecting digital information. On the desk there is a Nokia Digital Pen and three documents forming part of an interactive narrative: a star map,

a book, and a collection of love letters. Ambient sound in combination with artificial lighting is applied to generate different moods.

The content of the interactive narrative is based around the idea of a cosmonaut lost in space. The user interacts with the three documents on the table containing information about the lost cosmonaut. While a visitor is interacting with the documents, the content presented on the round screen as well as the ambient sound and the lighting are changing (ambient moods). There is some preauthored content in these documents, but visitors are encouraged to add texts and drawings to the artefacts themselves. While the linearity of the story is already broken by giving a user the freedom to select arbitrary information in the three documents, each user further becomes an author of the story by adding their own content. Thereby, the interactive narrative collaboratively written by different users evolves over time. Note that the information is written into physical space as well as captured and integrated into the narrative in digital form.



(a) Love letters (b) Star map (c) Broken images

Figure 6.20: Lost Cosmonaut documents

The three documents building the interactive narrative are shown in Figure 6.20. The leftmost picture shows a collection of love letters. The letters mainly contain parts of handwritten sentences or even single words with gaps in between. The user is encouraged to fill in the missing parts and to continue the love letters. The second document, an ancient star map contains stars labelled with numbers. Visitors can dedicate stars to family and friends by writing names alongside stars. Finally, there is the book of broken images containing torn images of cosmonauts, landscapes etc. where half of each image is missing. Some of the missing half images have been drawn by somebody else and the visitor is encouraged to follow suit.

In the Lost Cosmonaut installation, the ambient mood changes based on the document a visitor is working on. Using simple document tracking, the system detects which document lies at the centre of the desk. The documents have been tagged with RFID identifiers and an antenna has been placed underneath the table. The RFID antenna detects when a new document is placed at the centre of the table and initiates the activation of a new mood. An RFID plug-in has been implemented for the iServer platform to support the necessary document tracking.

The overall Lost Cosmonaut architecture, shown in Figure 6.21, is similar to the one used in the EdFest demonstrator with the iPaper Client, the Client Controller, iServer and the OMSwe publishing component. A new Service Message (SMessage) format was introduced for controlling various output channels such as light control, ambient sound and video. The Client Controller communicates with the RFID antenna and sends any recognised tag identifiers to the iServer RFID plug-in.

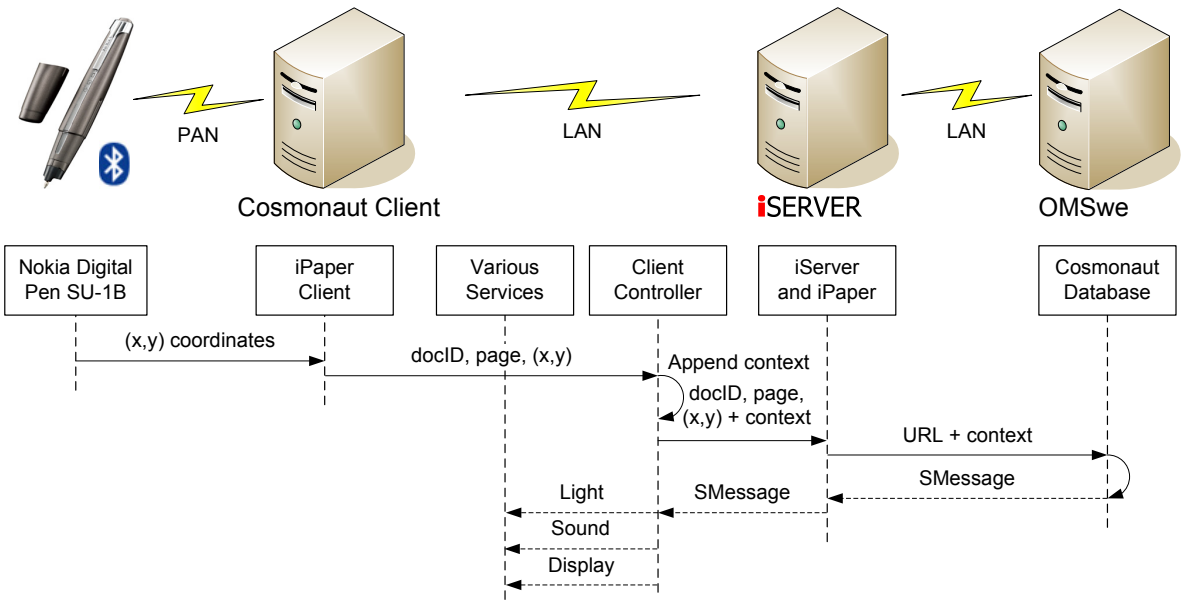


Figure 6.21: Lost Cosmonaut system architecture

The requirements of the installation changed as the artist developed his ideas and it was essential to have an extensible system architecture such as the one presented in Figure 6.21. The chosen architecture not only supported a rapid application prototyping in terms of content and services, but also enabled easy integration of new input and output channels.

An interesting and innovative aspect of the Lost Cosmonaut installation is the fact that information written into the physical space is handled

in three different ways to integrate it into the interactive narrative. To illustrate what we mean by the three different ways of content handling, we discuss the interaction with one of the three documents and outline how content from all three documents becomes interweaved to a single story space. Figure 6.22 shows an abstract representation of the three different documents. As described earlier, for each of the three documents there exists some preauthored content and new content generated by each user. In Figure 6.22 this fact is represented by separating each document into *existing content* and *new content*.

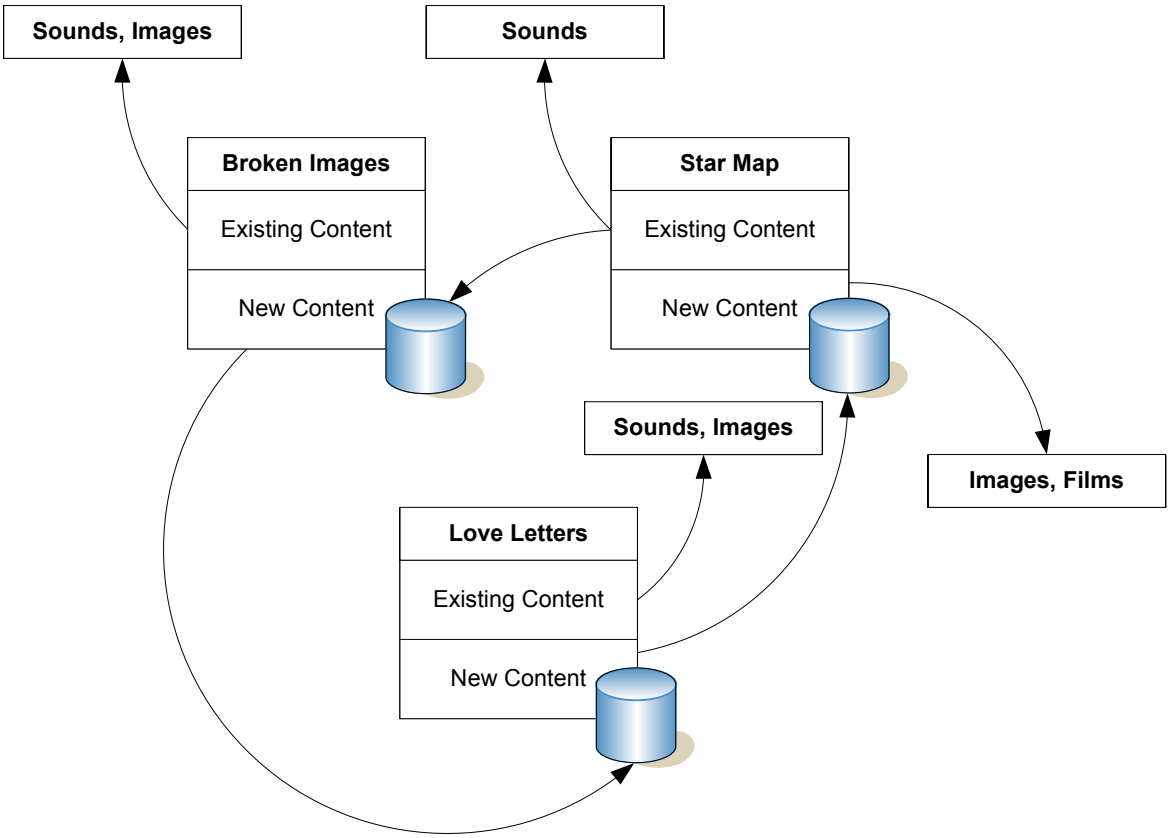


Figure 6.22: Cross-linked resources

Let us have a look at the star map document to explain the linking between the digital and physical space and show the three different ways of handling information written on a document. If a user writes a new dedication on the star map, first of all, the information is physically "stored" as new content on the document and forms part of a subsequent user's experience. By writing a dedication on the star map, a new active area is generated for the handwritten information. As illustrated in Figure 6.22 the new active area is linked with image and film material. If later, the same or another user touches the dedication, an image or

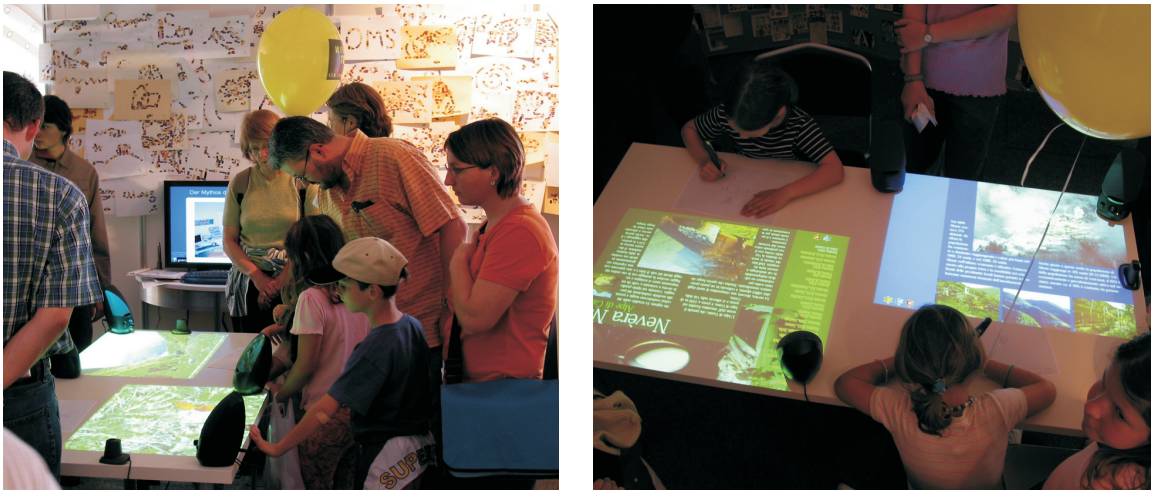
film is shown. Finally, the dedication is captured and stored in a database as an XML document containing single positions together with a timestamp and the pressure of the pen nib. The digitally captured information is dynamically linked from active areas in the love letters and gets activated when the corresponding part of a sentence is selected in one of the love letters. The temporal information that is stored together with the stroke information is used to replay the writing of the text as an animated drawing.

The Lost Cosmonaut installation showed how interactive paper can not only be integrated with digital information, but become part of a whole interactive environment including video screens, ambient sound and light. By cross-linking physical and digital information snippets, iServer has been used to author an interactive narrative where each user experiences their own story in a non-linear way. The Lost Cosmonaut interactive narrative encourages visitors to generate new physical and digital content resulting in an active cross-media story evolving over time. A first prototype of the Lost Cosmonaut was shown in December 2004 in an exhibition at ETH Zurich.

6.9 Generosa Enterprise

A second interactive art installation was realised as part of the 150 year jubilee of ETH Zurich and presented to a wide public for two and a half weeks in the *Welten des Wissens* exhibition at Platzspitz Zurich. The Generosa Enterprise [51] installation provides information about Monte Generoso, a mountain located in the southern part of Switzerland. The visitors of the exhibition are taken along a journey where they can experience the world of Monte Generoso combining art, science and interactive paper technology.

Information about the worlds of air, water, earth and fauna at Monte Generoso have been collected and stored in a database in the form of movies, photographs, sound clips and texts as well as pieces of art and scientific research results. The idea is that users can access information stored in the database by drawing a picture. At the same time, they create a new artwork consisting of their personal drawing augmented with digital information about the mountain which is retrieved from the database. Each user is given an empty sheet of paper and a Digital Pen. The user takes a seat at one of the two workplaces shown in Figure 6.23 and starts to draw a picture, write some text etc.



(a) Exhibition space (b) Workplace

Figure 6.23: Generosa Enterprise installation

Although the initial paper seems to be empty, it hides digital information which can be activated by drawing with the Digital Pen. A virtual grid with a lattice spacing of one centimetre covers the whole page, where each square centimetre links to a specific piece of information about Monte Generoso in the form of an image, a sound or a movie. Each time a user "selects" a new square while drawing a picture, the information associated with this specific square gets activated. The users do not know which information is hidden in specific areas of the drawing sheet and therefore have to explore the initially empty page rather than deliberately accessing information about the mountain. Movies and images are projected directly onto the table, in front of the drawing area, by using over desk projection. Each workplace is further equipped with a pair of loudspeakers for sound output. The system architecture that was used for the Generosa Enterprise installation is similar to the EdFest and Lost Cosmonaut architectures. The major difference is that we did not use the Client Controller but apply active components to directly control all input and output channels.

The augmentation of the drawing process with digital information about Monte Generoso had different effects on visitors of the exhibition. Some of them did not look at the projections at all and were just very focussed on drawing their picture. Other users were inspired by the movies, images and sounds and integrated parts of it into their personal drawing or started discussing the presented material with other visitors. A key factor for the success and popularity of the Generosa Enterprise installation was its simple user interface. Everybody knows how to use

a pen to draw a picture on a piece of paper and therefore even people who normally have no access to computers had no problems at all using the system. The only guided interaction with the non visible computer takes place when a user has finished their drawing. There is a field a the bottom of each page where the user can sign their artwork and finally a special box that has to be ticked with the Digital Pen. The ticking of the box is the signal for the computer that the visitor has finished their drawing which is ready for further processing.

All captured strokes are used to rebuild a digital version of the original artwork. This line drawing is augmented with any digital information that was accessed by a specific user while interacting with the system. For each movie or sound, a small thumbnail was created in advance and stored in the database. These thumbnails are now combined with the original drawing, leaving traces about the digital information that was accessed. Each time a stroke enters one of the virtual squares linking to the digital information, the corresponding thumbnail is inserted as a background image. The final artwork is sent automatically to a printer which is hidden behind one of the booth walls and, a few seconds after a visitor has finished their drawing, they can collect the printout through a small slot in the wall shown in Figure 6.24(a).

(a) Printer (b) Transformed drawing

Figure 6.24: Transformation and printout of drawings

The original drawing together with the augmented printout, an example of which is shown in Figure 6.24(b) could be taken home by the visitors. Although a massive amount of computing power and other technical equipment was necessary for realising the Generosa Enterprise installation, all computers and other devices, for example the printer,

where hidden from the users. They could interact with latest IT technology and access information about Monte Generoso in a very natural way, namely using paper and pen.

6.10 Summary

In this chapter we have presented a variety of applications dealing with different aspects of the iServer architecture. While some of the applications dealt with enhanced reading only, others combined enhanced writing with enhanced reading functionality to realise highly interactive information spaces.

In the process of building these applications, the iServer architecture was continuously extended and new functionality added. For example, a new RFID plug-in for tracking physical objects was realised as part of the Lost Cosmonaut project.

Note that all applications are based on the same iServer information platform. This has the major advantage that functionality which was designed for one application could be reused by other applications. For example, some of the active components that were implemented as part of the EdFest festival guide could be reused in the Lost Cosmonaut installation. Since all applications are based on the same iServer model, it even becomes possible to enable cross-application information sharing.

The implementation of the different applications validated the flexibility and extensibility of the cross-media information management platform and its interactive paper plug-in. A number of hardware technologies for input processing have been tested in the presented applications and since the interactive paper plug-in resolves links on logically defined active areas, it was even possible to operate a single application with different input devices.

While the interactive paper framework is general and extensible, most applications profit from the concept of active content where the whole program logic becomes encapsulated in small objects. The active component concept simplifies the development process of new paper-based applications and supports the rapid prototyping of interactive paper applications. A major issue in realising a new application is the content aggregation. In the next chapter we therefore describe various forms of authoring for the iServer platform and show how these were applied in realising the applications presented in this chapter.

Information is a source of learning. But unless it is organized, processed, and available to the right people in a format for decision making, it is a burden, not a benefit.

William Pollard

7 Authoring and Cross-Media Publishing

While in the previous chapter we introduced various applications that have been implemented based on iServer and the interactive paper platform, in this chapter we discuss the necessary authoring tools to create such cross-media applications and present different solutions for cross-media publishing.

When considering the activity of authoring cross-media information, there are a number of fundamental factors to take into account. These factors deal with issues such as the intended usage of the system and the source of materials. They will greatly influence the types of authoring to be supported and hence the approaches and tools to be used.

A first factor to consider is whether the content is already available in the form of digital and printed materials. In this case, the main authoring activity is link authoring where associations are defined between existing material and a tool is required to support the creation of links between the existing physical and digital information entities. If, however, we want to support a publisher of new cross-media materials, we need to offer content authoring as well as link authoring.

Another factor is whether only the publishers can author links or also the users of the material are allowed to add new information. In the latter case *dynamic link authoring* must be supported. If users can

author their own links, the next question is whether there is an *open link authoring* scheme, in which they can link arbitrary materials and not only those provided by a single publisher. For example, a student may wish to link concepts in a set of lecture notes to textbooks or to web pages. Clearly, the ability to freely create links between arbitrary materials implies a major shift to innovative information systems for cross-media publishing.

These and many other issues are going to be discussed in this chapter. We first provide a classification of authoring types including link authoring, content authoring, annotations etc. We then present our solutions for authoring interactive paper applications and introduce a general cross-media authoring tool for the iServer platform. We conclude the chapter with a general discussion on how publishers and customers could profit in the long run from a switch from traditional publishing to more powerful forms of cross-media publishing.

7.1 Classification of Authoring Types

Before presenting various authoring tools that have been developed for interactive paper and cross-media publishing, we provide a classification of approaches for creating information environments that integrate different types of resources such as paper and digital information. While discussing different authoring types we mainly use the integration of paper and digital information as a specific use case of the more general cross-media authoring process, where arbitrary cross-media spaces, containing physical as well as digital resources, are designed.

7.1.1 Link Authoring

Let us begin by discussing the authoring for existing digital and physical information, where content is already available in the form of different resource types and has to be integrated by linking these digital or physical information entities. The Nature Encyclopaedia application that was presented in the previous chapter is an example of such an application where all content was already available in the form of a printed book and a CD-ROM provided by Dorling Kindersley. The main authoring task for the Nature Encyclopaedia therefore was to define the appropriate associations between parts of the printed book and information available in digital format.

As we have seen when introducing the iServer cross-media information model, the concept of a selector is applied to address parts of a resource. For link authoring, where associations are defined between existing resources, a major issue is the definition of these selectors for specific resource types, which can then be used as link source or target anchors. For example, for an interactive paper link authoring tool this implies that we need a mechanism to define active regions (shapes) which then can be linked to supplementary digital or physical information.

Link authoring tools can be further classified according to whether they allow only a publisher to define cross-media links or also users, enabling them to define their personal associations between related resources. In a *static link authoring* system, links between different resources are fixed and have been preauthored by the publisher of an application. Users can freely navigate back and forth between printed and digital information, according to the links provided by the editor of the cross-media information space, but they cannot extend the link space by adding their personal link data. This is very similar to the situation in today's standard web browsers where users cannot change the content of a web page without being the owner of the page as discussed earlier in Section 4.1.2. These static link authoring systems have a clear separation of the publisher who has the role of a content provider and the user who is only consuming the information. It is the responsibility of the publisher to ensure that the content is accurate.

As mentioned earlier, dynamic link authoring systems enable users to create personal links. In the case of web browsers, dynamic link authoring is being introduced through the adoption of standards such as XLink [41] in combination with research prototype browsers such as the W3Cs Amaya web browser [5]. The dynamic link authoring from existing printed documents to digital resources is relatively simple in that the documents are static and no special consideration has to be given to maintaining link consistency under changes. The authoring tool must simply provide a means for the user to specify an active area of a page and the object to which it should be linked. The main issue in this case is how to make users aware of the existence of such links which have not been designed by the document publisher. One possible solution could be to integrate some highlighting functionality with the reader device [7].

A major advantage of the iServer cross-media information management platform compared to related technologies is the possibility to support links back from a link's target to all of its sources without any additional authoring effort. In the case of the interactive paper plug-in

for iServer, this implies that, after having defined a link from paper to digital media, one can link back from that digital resource to the corresponding link source anchor defined in the paper document. Based on the meta information consisting of a document identifier, a page number and a shape that is stored together with every bidirectional link, the system can automatically generate links from a digital information entity to all of the documents referencing it.

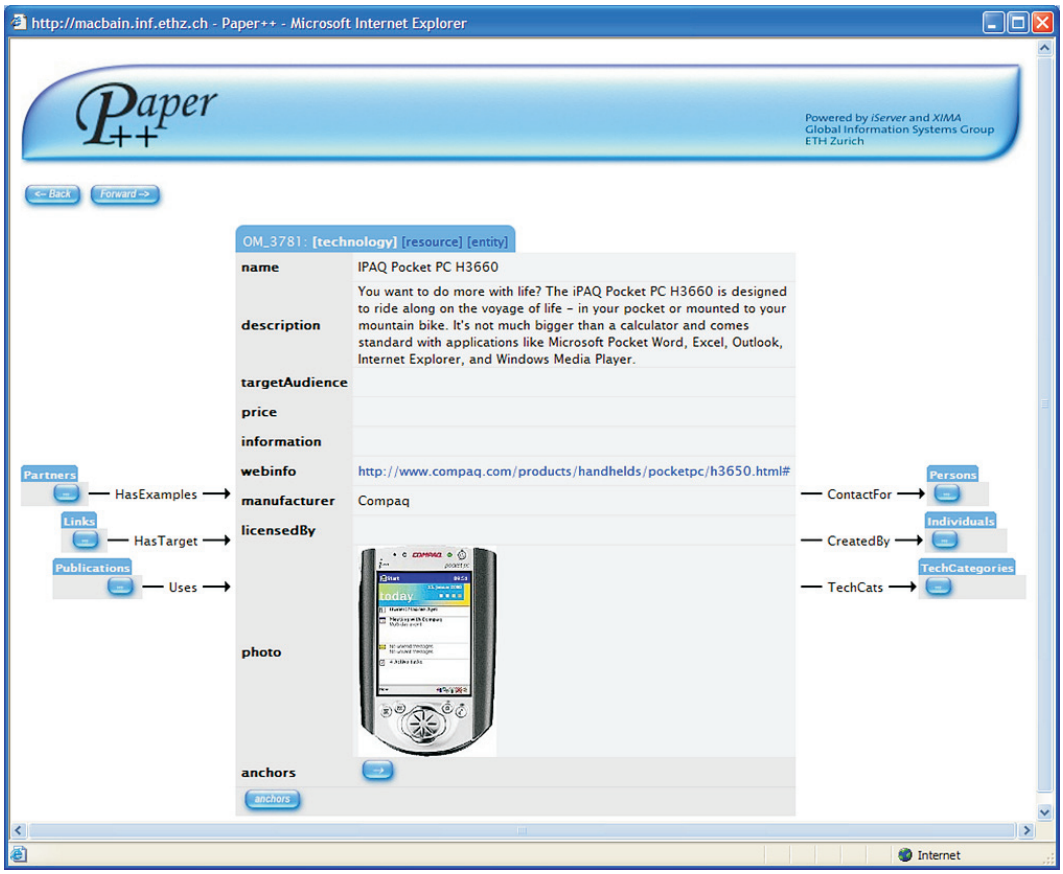


Figure 7.1: Default HTML representation

To illustrate this implicit authoring of backlinks, let us revisit the publication annotation application. In Figure 7.1 we presented the default representation of a digital information entity as an HTML page. Each digital information entity provides an `anchors` method represented by the 'anchors' button at the bottom of the entity's HTML view. The activation of this button results in a collection of all sources of links which have the current entity as a link target and therefore provides functionality for backward link navigation. The result includes information about the document's name and page number that each shape is bound to and further uses the shape's location information to visualise the shape as a highlighted area as shown in Figure 7.2.

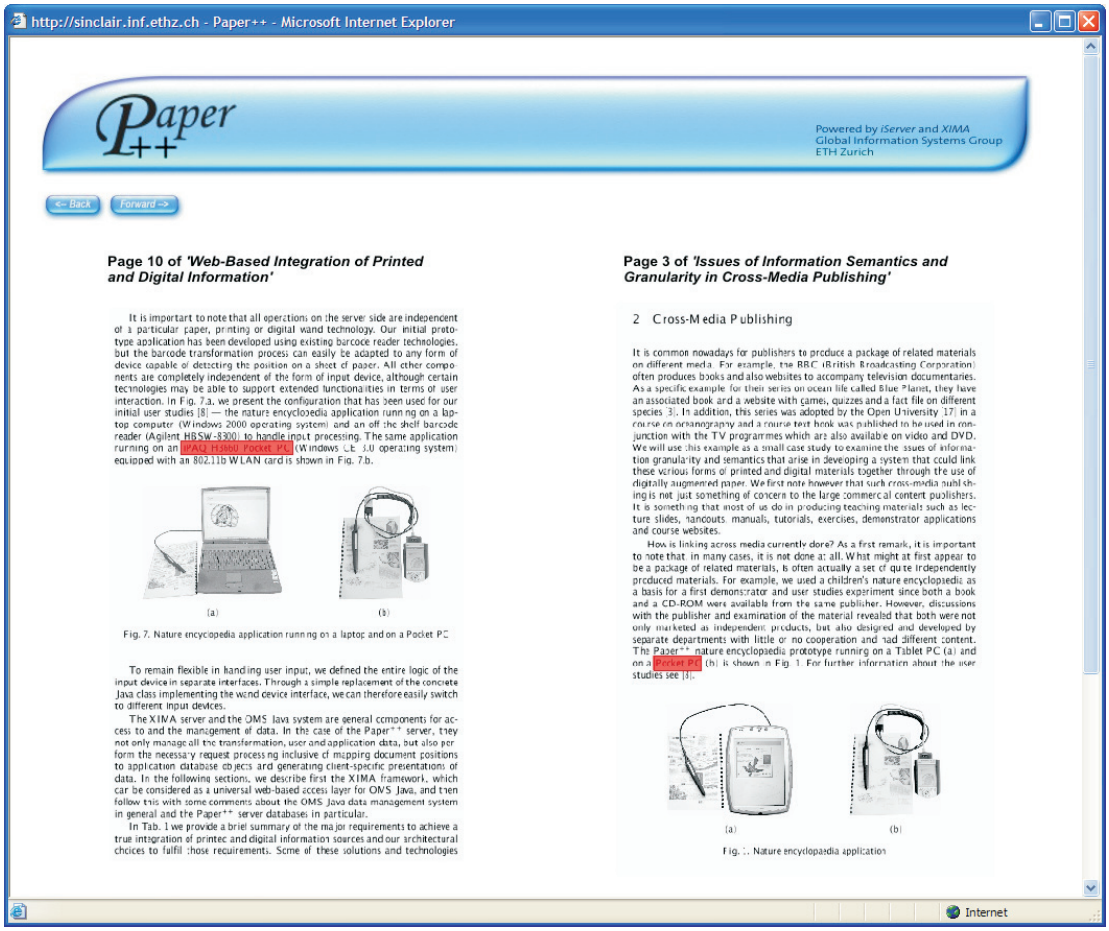


Figure 7.2: Links from digital content to paper

In our example, the interactive paper platform returns two anchors defined in two different paper documents. The first document that is shown on the left hand side of Figure 7.2 is the document that we are currently working with and from which we have activated the link to the digital information about the Pocket PC. The document on the right hand side also contains a link to the same digital information unit about the Pocket PC and we could have a look at the paper to get information about another project using Pocket PC technology. Of course, we still have to locate the physical instance of the corresponding document but, in the near future, even this task could be simplified by the tagging of paper documents with visual or acoustic tags.

The backlinks become even more powerful as soon as multiple users start to use the system to augment papers that they are reading. As we show later, our cross-media authoring tools allow for sharing of link information. Thereby, a user can profit from other users' work and knowledge by getting access to their link metadata. For the publication annotation

application this implies that a user can get information about other publications in which the Pocket PC technology has been mentioned based on link information added by other users.

Note that, for effective link authoring, we assume that any existing content is fixed and will not change over time. This does not mean that we cannot handle new content, but as soon as a resource has been added to the link space, its content should no longer change. For example, for an interactive paper document this implies that the text of the document should no longer be modified after link authoring has been applied to the document. If the consistency of links is to be maintained under editing changes to documents, then the authoring tool needs access to the document rendering engine so that link positions can be updated in new versions of the printed document. Details about this form of link definition are provided in Section 7.1.3 when introducing content authoring.

An author of new links can define the visibility of the created link by granting access to individuals or groups of users. In this way we can distinguish between links for private or public consumption.

As outlined in this section, link authoring defines associations between existing resources. An authoring process that may involve an additional step, namely the creation of the content to be linked, is presented in the next section where we introduce annotations as an extended form of link authoring.

7.1.2 Annotations

Annotation is some means of marking up a resource to augment the existing material. For example, annotations could be used to explain terms occurring in a document, thereby providing direct access into a glossary during the reading process. Annotations come in many forms and have a variety of uses. They may be private or public, permanent or transient, and formal or informal. *Informal annotations* often take the form of free text but could also be sketches, images, or audio for voice commentary. For example, marginalia — the comments that we write in the margins when we read a paper — are informal annotations.

Formal annotations follow defined structures and conventions that enable them to be interpreted by other persons or computer programs. Typographic markup for the editing of documents is one such example, where it enables someone else to unambiguously interpret the changes to be done and carry them out. Another example of formal annotation that has been an active area of recent research within the scientific database

and semantic web [17] communities is the use of annotation to mark up digital data with metadata. This metadata describes the semantic content of the data and later enables either human readers or programs to access and process that data. For example, the annotation of scientific images with metadata can aid both search processes [53] and the integration of scientific data sources [52]. Further, within the hypermedia and semantic web communities, metadata annotation is also used to generate links between documents.

Most annotation tools support one particular form of annotation as they tend to support specific activities such as collaborative writing, data integration, search and so on. However, if one considers the working environment of most people, they have to work with lots of different forms of information and perform a wide variety of tasks that may require different forms of annotation. For example, in a study of university textbooks, it was shown that some kinds of annotations were used to support reading, while others were used to support writing [111]. In the case of reading, notes are often written alongside text and figures to aid interpretation in future readings, either by the same reader or other readers, and important sections highlighted or underlined.

Another study examined the task of writing document summaries and how users would annotate the document and take notes [134]. In this case, readers not only wished to highlight important items, but also to extract and re-order them according to the final structure of their summary. Annotations alongside the text heavily used references to structure outlines. Further, this study examined the differences between performing this task using only digital documents as opposed to the use of only paper. It was shown that there are many problems with digital annotation systems in terms of both inputting the actual annotations and also working with various documents alongside each other.

Figure 7.3 shows an example where a printed image has been augmented with various forms of digital information including a video, a text and a web page. This form of annotation using existing digital or physical material is closely related to link authoring presented in the previous section. The difference between annotations and link authoring is actually quite subtle. Besides the fact that the process of annotating frequently includes adding new content to the system, we regard an annotation as basically being just a special classification of a link. Classifying them as annotations simply provides an easy means of handling annotations in a special way, such as making them visible or invisible depending on whether we want to see the augmented or original document.

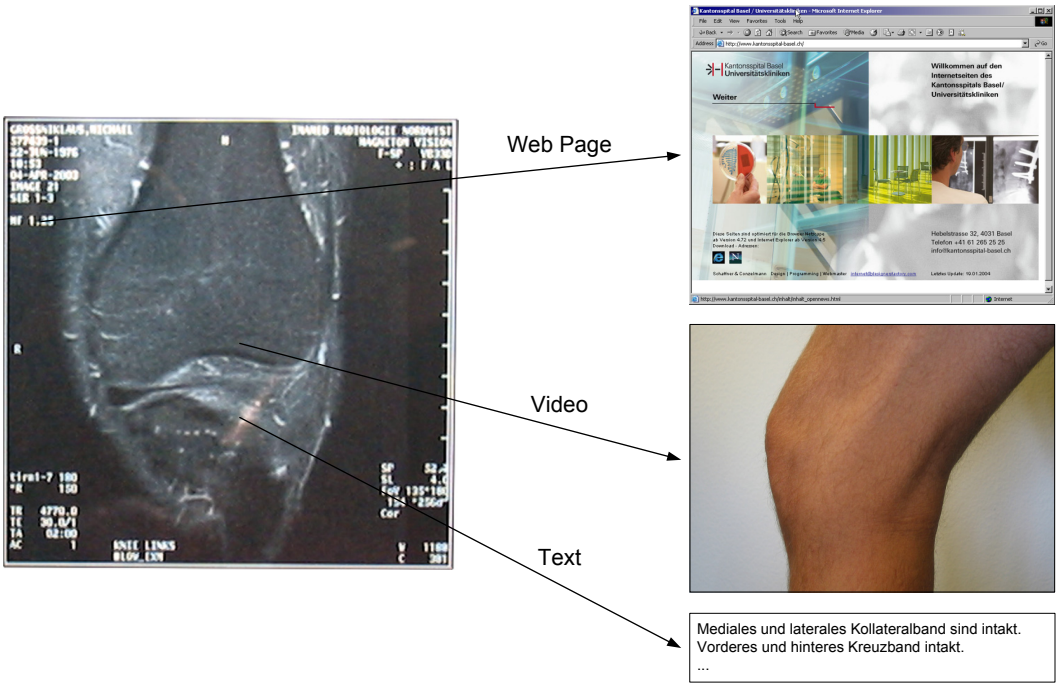


Figure 7.3: Annotation of printed images

Annotating differs from normal linking through the fact that the authoring of the target resources is often an important part of the annotation process. Thus, in addition to link authoring, annotating also includes the creation of new content, for example, based on handwriting capture.

Figure 7.4 provides an overview of our annotation framework based on iServer [39, 40]. The annotation of an artefact basically corresponds to the combination of the authoring of a link along with the authoring of the content to which the link points. However, note that, instead of adding new content, users could also make annotations by linking to existing resources.

We use the term *uninterpreted resource* to denote a resource that can be referenced by the annotation framework, but is interpreted only by external applications used to view the resource. In contrast, an *interpreted resource* is one for which the framework understands the structure and is able to reference elements within the resource. At the moment, both paper documents and databases are interpreted resources as the framework can reference elements within a paper document through the notion of active areas and also objects within a database through a querying mechanism.

As described in Chapter 4, the resource-specific plug-ins of iServer include extensions of the generic data model's concepts for resources and

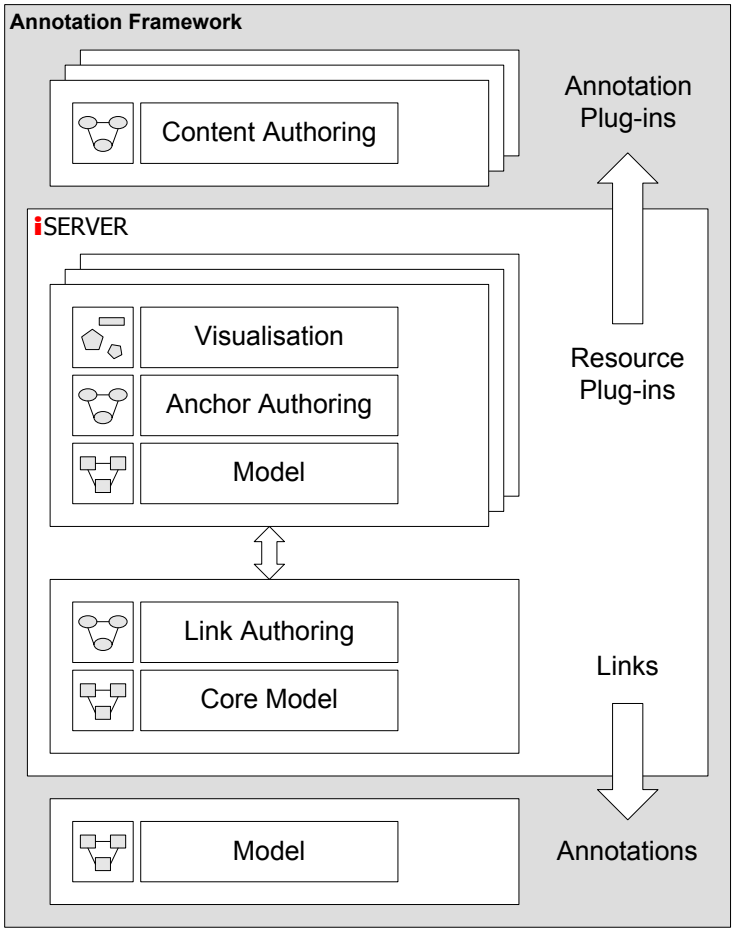


Figure 7.4: Annotation framework

selectors and application components for the visualisation of the resources along with the source anchors and links. The visualisation component for the image plug-in, for example, displays the image document overlaid with semi-transparent shapes representing the source and target anchors of the links. These links can be activated by clicking with the mouse on the active areas. Note that, in the case of the interactive paper plug-in, the visualisation component is the augmented paper document itself.

Figure 7.5 gives an overview of the annotation model which basically is an extension of the iServer model introducing further classifications of annotations represented by the iServer link concept. This has been achieved by modelling `Annotations` as a subcategory of `Links`, i.e. links can be classified as annotations. This design leaves a maximum amount of flexibility to the application and allows for a tight integration of link server, open hypermedia and annotation issues. The classification of annotations can be refined even further by introducing subcategories such as `Comments`, `Explanations`, `Examples`, `Formal` or `Informal`. Note that

an annotation can be classified in multiple categories at the same time and applications can also define their own categories.

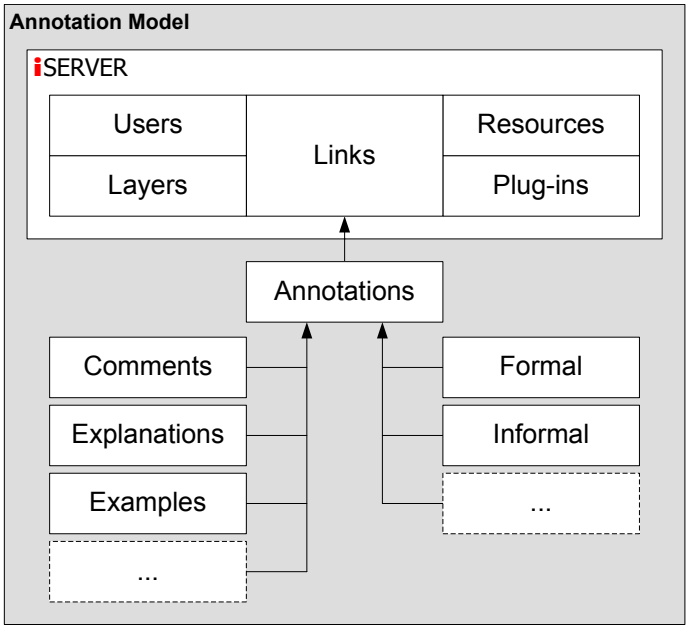


Figure 7.5: Annotation model

7.1.3 Content Authoring

So far, we have considered link authoring where the content to be linked already exists and will not change over time and annotations, where the link targets may be created as part of the authoring process but link sources are existing entities. If the content to be used as a link source does not already exist, then the authoring activity will consist of both content authoring and link authoring. In this case, the content can be developed with the resulting cross-media system in mind and tools can be used that generate both printed and digital documents along with the links between them. Thereby, instead of using traditional authoring tools such as a word processor, a content management approach of the form now prevalent in website engineering can be extended for cross-media publishing.

Instead of defining links at a rather low level, for example by specifying active areas in the case of interactive paper, the goal of content authoring is to define them at a conceptual level. This is only possible by managing all content in a specific publishing framework. In the publishing system we then can specify the concepts to be linked. For example

we could define that a link has to be generated for an occurrence of the word 'ETH Zurich'. The publishing framework then will generate the required link metadata for iServer when the document containing the specified words gets published.

A major advantage of such a content authoring approach is that the link metadata can automatically be generated for different output channels. For the example of our document containing a link defined for the occurrence of 'ETH Zurich' this implies that the document can be published as a web page as well as an interactive paper document. In both cases the publishing framework generates the necessary link metadata and adds it to the iServer cross-media platform.

Content authoring not only solves the problem of link consistency after changing a document's content, but also provides a whole set of new authoring functionality. For example, instead of linking only a single occurrence of the word 'ETH Zurich', it becomes easy to define that all occurrences of this word in any document have to be linked to a specific resource. Again, every time a document containing this word gets published, the necessary link metadata is generated automatically by the publishing framework.

The two main issues in content authoring are how to automatically create the links between different information entities when information is delivered on a specific output channel and, secondly, to guarantee link consistency if the underlying information changes over time. We have investigated automatic link generation based on exporting this information from our database system to the iServer infrastructure and this kind of link authoring was used in the EdFest application. Details about a first content authoring prototype are provided when discussing the interactive paper authoring tools.

7.1.4 Adaptive Authoring

The explicit authoring of links either by the publisher in the case of static link authoring or by a user in cases where dynamic link authoring is also supported, is only one possible way to build up a rich cross-media information space. Analogous to ongoing work on adaptable links by the hypertext community [13, 116, 147], we have investigated the concept of *derived links*. Let us presume that we have two paper documents, `document A` and `document B`, as shown in Figure 7.6. Further assume that from both of these two documents we have authored links to the same application domain database.

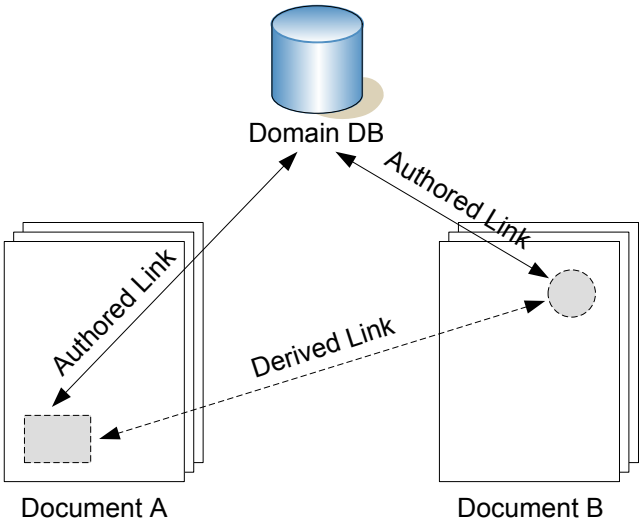


Figure 7.6: Derived links

Just by analysing different users' browsing behaviour, we may conclude that many users starting from the rectangular active area in document `A`, end up following some links corresponding to associations in the domain database, but finally come back to physical paper, namely to the circular shape defined in document `B`. This implies that there must be a strong correlation between the concept represented by the rectangular active area in document `A` and the concept represented by the circular active area in document `B`. In such a case, iServer may dynamically generate a direct paper-to-paper link between the two different concepts based on the user access statistics. These derived links are only a first step towards an iServer extension for adaptive hypermedia based on user access statistics which may automatically evolve over time.

In addition, user profiling and access paths can be used to dynamically generate *trails*, suggesting links to users based either on their own or the community's experiences similar to the ideas suggested in Bush's Memex system [28]. By introducing virtual users in the form of software agents as a special kind of system user, we can provide components that autonomously generate new dynamic links based on gathered user profiling information similar to Bush's trailblazers in the Memex system or other recommender systems [27, 34, 157].

Note that most approaches proposed by the hypertext community for adaptable hypermedia could be directly applied on the iServer cross-media link architecture. For example, access statistics could not only be used for dynamically generating new links between resources but also for automatic classification of information or to find users with similar interests by analysing their information access statistics.

7.1.5 Dynamic and Collaborative Authoring

The explicit authoring of links by a publisher can certainly be time-consuming and it may be based on the knowledge or experience of a single individual. Especially in the case of open, dynamic link authoring systems, it is desirable to share links among communities of users, thereby enabling users to benefit from the knowledge of others.

Generally, we can distinguish preauthored, personal and dynamic links. Preauthored links are generated by publishers who are domain experts and made available to specific groups of system users. As mentioned above, the authoring of such predefined *tours* can be very time-consuming and, furthermore, it is often difficult to know different users' needs in advance. Therefore, in addition to the preauthored links, each user should be given the possibility to create their own set of links and either keep them private or share their link knowledge with the community of users.

Many of the interactive paper applications presented in the previous chapter were exclusively based on a clear publisher/consumer model, where the publisher, for example Dorling Kindersley, would do all of the authoring and the consumer reads and browses the preauthored information. The design of the corresponding cross-media information spaces proved to be a time-consuming and expensive process. Hence it was an important goal of the generalised iServer framework that we could somehow facilitate the link authoring process. In some more recent iServer applications, such as the Lost Cosmonaut or the EdFest guide, the users create new information and links in interacting with the system.

This leads to alternative authoring paradigms where more power is given to the user. In addition to being a consumer of preauthored links, a user becomes a potential author of new link knowledge. In the peer-to-peer version of iServer presented earlier each user has their personal link information space which can be dynamically enriched with link information from other users. By applying such an adaptive community authoring process, each user can contribute information according to their domain of expertise and, on the other hand, profit by the knowledge of experts in other domains.

A similar trend of open authoring can be found on the Web where, until recently, there was a clear distinction of the author of a web page and the consumer of the information provided on a specific page. New technologies such as the XLink standard support external links, which means that a user no longer has to be the owner of a web page to add new

links. The idea of storing link information separately from a resource's actual content is not at all new and has always been the goal of distributed link services. Furthermore, latest results in information retrieval research show that link knowledge plays a crucial role in classifying Web documents [30].

Such an open authoring approach can, of course, always be combined with the more controlled publisher and consumer model by treating well-known publishers as authors with a high confidence value. However, the community authoring process is much more flexible in adapting quickly to evolving or new information and, furthermore, enables parts of the time-consuming and expensive authoring process to be sourced out to the community of users.

7.2 Interactive Paper Authoring

Various tools have been implemented for authoring the interactive paper applications presented in the previous chapter. Link authoring as well as content authoring components have been developed and used in various applications. However, the presented authoring tools are initial prototypes and there is potential for new authoring features and future extensions.

7.2.1 Digital Link Authoring

Our link authoring tool for interactive paper [42] is displayed in Figure 7.7. The `Document` view shows the digital version of a document page together with the active areas represented by shapes that have been defined for the specific page. For example, there are shapes defined around the four main components of the figure that forms part of the document but also shapes for specific words within the text such as 'OMS Java' or 'XIMA'. Note that it is not necessary to provide a digital version of a document but it significantly simplifies the authoring process. In the case that a digital version of the document is available in the form of a PDF file, the authoring tool will show an image of the appropriate document page as the background of the document view. However, if no digital document version is available, a white background image with the appropriate page dimensions is shown.

The `Navigator` view on the left hand side of the `Document` view shows all interactive documents available in the iServer database. By clicking

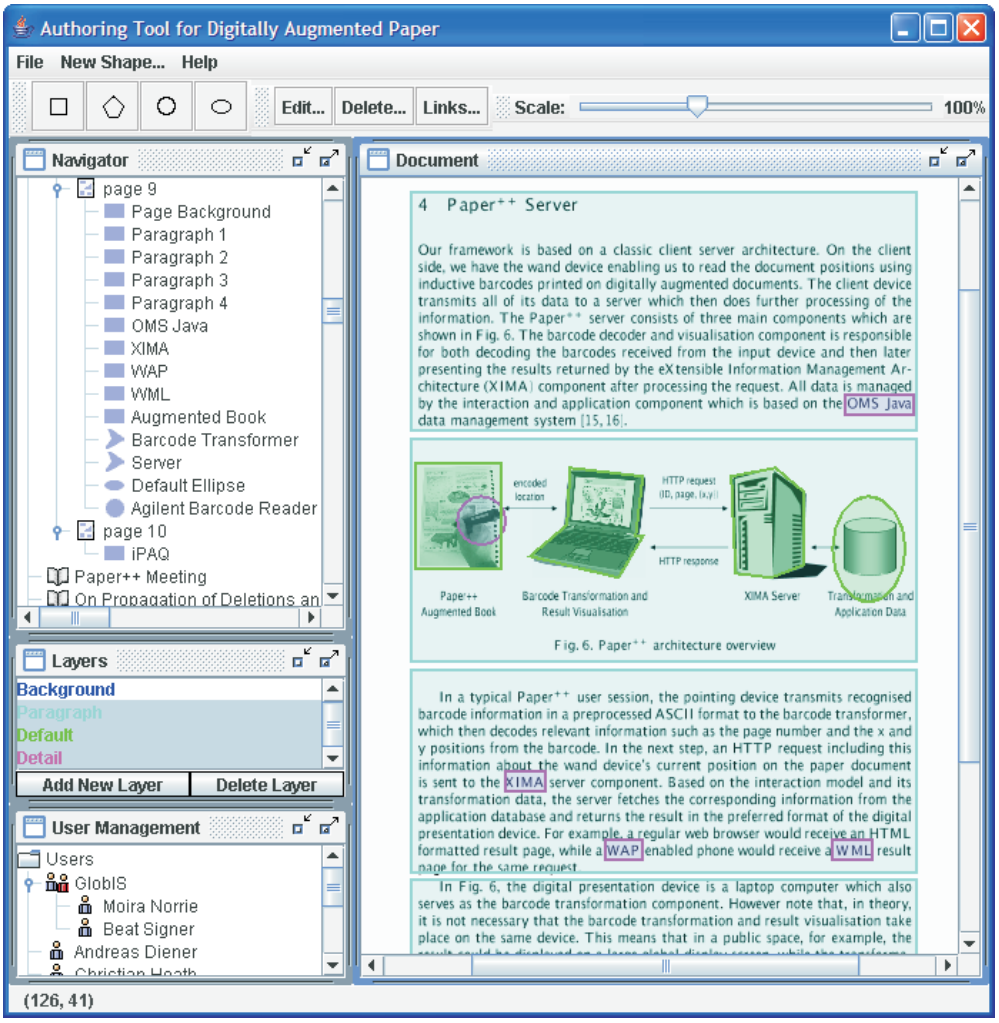


Figure 7.7: Interactive paper authoring tool

on a single document, it gets expanded in the tree view and the different document pages together with all shapes defined for a single page are shown in the `Navigator` view. For example, the `Navigator` view of the authoring tool shown in Figure 7.7 presents 'page 9', which happens to be the page currently outlined in the `Document` view, together with the corresponding shapes such as 'Paragraph 1'.

As introduced earlier, each iServer selector has to be associated with a specific layer. The `Layer` view provides access to the iServer's layer functionality. The ordering of the layers in the `Layer` view represents the ordering within iServer defining the priority in the case of overlapping selectors. A layer can be added to or removed from the list presented in the `Layer` view and the relative order of the layers can be changed by moving a single layer up or down in the list. Finally, by selecting specific layers in the `Layer` view, only the shapes lying on one of the selected layers are shown in the `Document` view.

The `User Management` view visualises the users in the form of individuals and groups defined by the current iServer instance. New individuals or groups can be added and an individual group membership can be changed. Other user management operations include the deletion or update of individuals and groups, respectively. Furthermore, the visibility of single entities can be restricted to single users or groups of users.

To define a new link with the authoring tool, a link source or target anchor has to be defined on a document page by defining a new shape which can either be a simple shape including rectangles, circles, ellipses and polygons or a complex shape which is a composition of multiple simple or complex shapes. Of course we do not always have to create new selectors but can also use existing shapes as the source or target anchor of a new link. Furthermore, the user is requested to select a layer that is then associated with the new shape. Last but not least, the user will be registered as the creator of the new shape and has the exclusive right to define access permission for the newly created selector.

After a new shape has been defined or an existing one has been selected, a new link can be created with the shape as a source or target object and another physical or digital object as the second link parameter. In the same way that the user has been associated as a creator of the shape, they also become the owner of the new link. The creator of any iServer entity, which can be a resource, a selector or a link, can define the access rights for other users and groups of users by adding them to the `AccessibleTo` and `InaccessibleTo` associations of the iServer model. As described in Section 3.3.3, this information is then applied to control access rights at the entity level.

The digital authoring of links based on the presented link authoring tool is appropriate if a digital version of a document's content is available, thereby enabling the authoring tool to provide this information as a page template in the authoring process. However, the definition of the link source and target anchors becomes more difficult if no digital document version is available since a white default background will be shown by the authoring tool and the author has no reference points to help define the appropriate selector shapes.

7.2.2 Paper-Based Link Authoring

For those cases where no digital document version is available, we provide an alternative approach for authoring interactive paper where the user does not have to define the active areas by marking them in a digital

on-screen representation of the document. Rather, the active regions can be generated directly on the physical paper by using the digital pen to draw the outline of a link's source or target anchor. With the Nokia Digital Pen that we used in many of our applications, it is possible to perform both link and content authoring on paper and similar physical objects. *Paper-based link authoring* uses the digital pen to define the boundaries of active areas directly on paper whereas the digital content to be linked is provided by another application. For example, a user could draw a new active area on a paper document page which then gets linked to the web page which is currently active in the web browser. Note that the web browser providing the URL of the current web page is just one of many possibilities for getting the digital information that has to be linked with the active paper area. In addition to the definition of a new selector by drawing its outline on a paper document, *paper-based content authoring* enables us to create new content based on the information captured by the pen.

In the case that a digital pen is used to author both links and content at the same time, the paper-based authoring tool requires a mechanism to distinguish between these two modes. There are several possibilities as to how this can be done. One is that the link source or target anchors and the captured annotations are made on two separate pages. Which anchor belongs to which annotation is then determined by the sequence of the entries. This makes it quite easy for the application, but the users have to be very disciplined in the authoring process. We have investigated other input modes to enable us to compare different approaches. For example, we also implemented a version with the anchor definitions and annotations on the same page, using predefined checkboxes to switch between source anchor definition and capture mode. In addition, a variety of other input modes with non-sequential authoring of anchors and annotations are possible and should be investigated further. However, it is worth noting that all of these solutions tend to have similar advantages and disadvantages.

In authoring activities, it is necessary to give feedback to the user. In the case of combined pen-based anchor and content authoring, the user should know whether the pen is currently in the state for selector authoring or content authoring. The digital pens that we have used in more recent applications can give feedback through vibrations. One possibility would be to use this functionality to provide information about state changes. In addition, the bodies of the digital pens have some LEDs which could provide another possibility for feedback. However, these

seem to be less useful for continuous feedback, as users tend to look at the pen nib when writing or sketching.

While some applications demand the combined pen-based authoring of links and content, others cannot manage with pen-based input only. We have already provided an example where the current web page is linked with a new paper selector. An alternative solution is to combine the digital link authoring tool that has been presented earlier in this section with a component for pen-based authoring. In such an authoring tool, some of the link source and target anchors can be defined with the digital tool while others can be defined directly by pen-based interaction on the paper document. The same holds for the resources to be linked which can either be generated with the digital authoring tool or result from the pen-based capture of information.

When defining a new active area we have to decide whether we are linking a specific document instance or all instances of the document. For example, if a user adds a link to an interactive book the question is whether the active area should only be available in their book or if the information should be in any other instance of the book. In the case that links should be managed at instance level, we need a mechanism to distinguish different instances of the same document by either using distinctive position encodings for each document or applying other document identification technologies (e.g. barcode or RFID document identifiers). However, if links are going to be defined at document rather than instance level, we have to be aware that other users need some way to find the new active regions that have been defined in other instances of the same document.

7.2.3 Pen-Based Writing Capture

We now describe in more detail the technical solution that we have developed to capture information written on paper with a digital pen, store it in a database and later visualise it on demand. As we have outlined when describing the interactive paper architecture, the client device sends single location objects to the iServer paper plug-in. However, as soon as many locations are acquired as part of a capture process, it no longer makes sense to send them as single requests to iServer. We decided to buffer them on the client device and send a single request to iServer only when a capture process has finished.

Figure 7.8 shows the required `BufferedInputDevice` interface together with the `BufferedInputDeviceEventListener`. Note that to

improve the readability of the UML diagram, we do not show all of the input device classes already introduced in Figure 5.3. The `BufferedInputDevice` interface has to be implemented by any input device that is going to be used in a paper-based capture process. It defines four methods to access an input device's buffer. The `getLocation` method just returns the last position the pen has captured together with its timestamp, whereas the `getLocations` method returns all locations available in the input device's buffer. The two other methods return all locations starting from a specific timestamp and all locations within a specific time range defined by its start and end time.

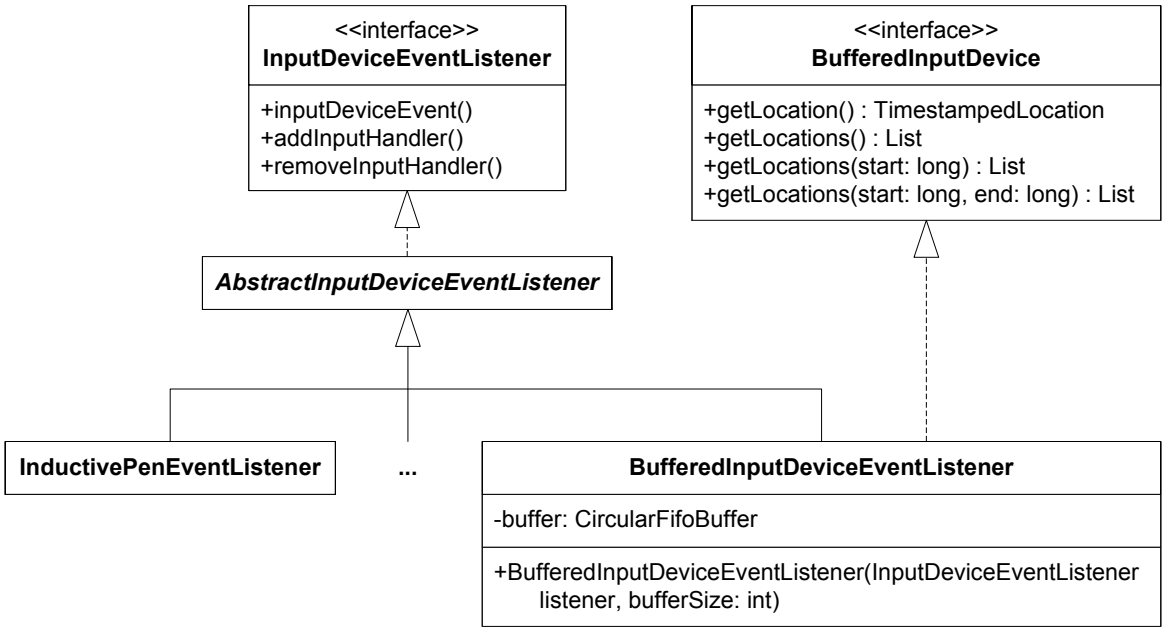
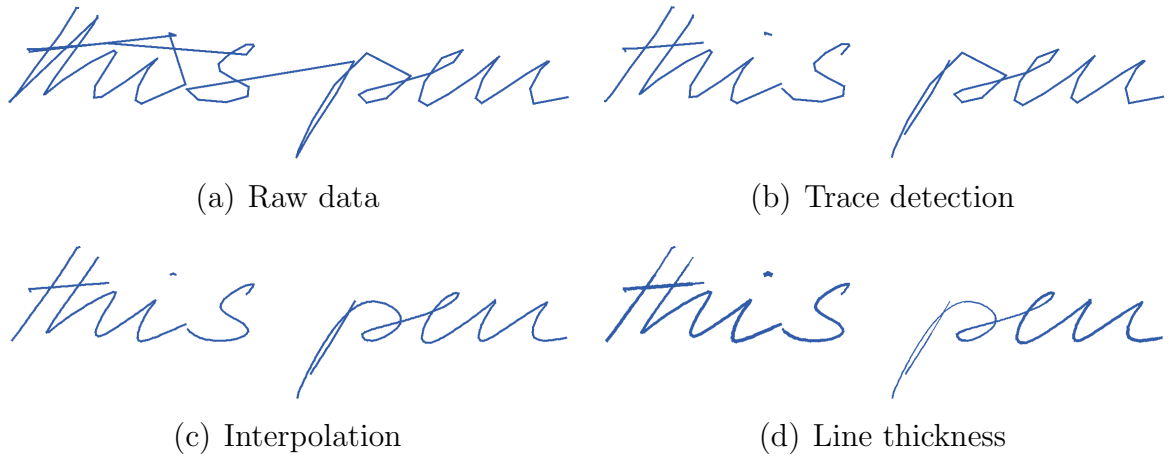


Figure 7.8: Buffered input device interface

Instead of changing the code of all existing input devices, we decided to go for a more flexible solution where the buffer is applied on top of existing input devices. This has the advantage that the buffer functionality has to be implemented only once and from then on can be applied to all existing and future implementations for the various input devices. The `BufferedInputDeviceEventListener` is an implementation of the `InputDeviceEventListener` interface that also conforms to the `BufferedInputDevice` interface. A new `BufferedInputDeviceEventListener` is constructed by providing an existing `InputDeviceEventListener` and defining the preferred size of the buffer. The `BufferedInputDeviceEventListener` class implements the methods defined in the `InputDeviceEventListener` by delegation to the listener instance

provided in its constructor. The `BufferedInputDeviceEventListener` further contains a circular FIFO (First In, First Out) buffer to store the `TimestampedLocation` object each time its `inputDeviceEvent` method gets invoked.

Based on the `BufferedInputDevice`, we get access to all location objects that have been accumulated during a capture process. A visualisation of the raw pen data of a single capture process is shown in Figure 7.9(a). To generate this output, connecting lines have been drawn between neighbouring location objects.



(a) Raw data (b) Trace detection

(c) Interpolation (d) Line thickness

Figure 7.9: Capture processing

A first problem that can be observed in the visualisation of the raw pen data is the fact that a user does not usually write a text or draw an image in a single stroke and this is not reflected by the image shown in Figure 7.9(a). Therefore, the pen data has to be further processed and *trace detection* has to be applied. If the pen is in contact with the paper, the pen continuously transmits positional information at a more or less constant rate. We can heuristically determine single traces by analysing the time elapsed between two successive location events. If the time difference exceeds a specific threshold, we assume that the pen has been lifted between the two events and therefore start with a new trace. The result of applying the trace detection algorithm to the raw pen data is shown in Figure 7.9(b). In addition to the time-based trace detection, the distance between two neighbouring points could be applied as another parameter.

The created list of traces can be serialised in an XML document as shown in Figure 7.10. Each `note` element consists of a list of `trace` elements, whereas a single `trace` element is built from one or more `points`.

Finally, each point is associated with its (x,y) position, a timestamp of the capture event and the force that has been applied to the pen nib when the point has been captured. This data then can be stored in the database in the form of an XML document.

```
<?xml version="1.0" encoding="UTF-8"?>
<ipaper>
  <note>
    <trace>
      <point>
        <x>104.43</x>
        <y>65.86</y>
        <timestamp>1103135662075</timestamp>
        <force>111.0</force>
      <point>
      ...
    </trace>
    ...
  </note>
</ipaper>
```

Figure 7.10: Captured note in XML format

A last problem occurs when the XML representation of a note has to be visualised. Since a pen has a limited sampling rate, single traces may appear a bit rough by just painting lines between the single locations stored in the XML document. Therefore, at visualisation time, we apply a cubic spline interpolation to get the appropriate amount of interpolated values in between two points of the original trace. After scaling the note to the desired size and applying the spline interpolation, the note, shown in Figure 7.9(c), can be output in different formats.

A first output format supported by the note visualisation component is the creation of a still image in the form of a JPEG or GIF document which can be stored somewhere in the file system. In addition to storing captured information in our own XML note format, it can be exported as Scalable Vector Graphics (SVG) [170], an XML format for vector graphics. Finally, the note can always be left in our own XML note format for further processing by other external processes. Such an XML note could then, for example, be transformed by XSLT to the Ink Markup Language (InkML) [75] which has been especially designed for pen-based interfaces.

In addition to these note export formats, we have implemented a Java visualisation component for captured pen strokes, which can, not only generate a static image from the data stored in an XML note, but also replay an animated version of the captured information. This note replay functionality has, for example, been used in the Lost Cosmonaut installation to enable visitors to watch the pen drawings and comments of previous users. The default real-time replay functionality is based on the timestamps available for each captured location but many other variants of dynamically animated drawings, such as a replay in reverse or random order, are possible. Both the replay, as well as the static visualisation components, can also make use of the pen force information which is stored for each captured position to vary the line thickness when rendering a note. This variation of the line thickness results in images as the one shown in Figure 7.9(d).

Based on the paper-based authoring approach, source anchors for the annotations can be defined by marking a specific area on the paper document with the digital pen. *Unstructured annotations* in the form of handwritten text or sketches can then be captured and digitised by the pen. The captured content is transmitted and added to the annotation server. Such an approach of pen- and paper-based annotation has, for example, been applied in the mammography screening application where abnormal areas on a mammogram can be marked and annotated with the digital pen.

In the case of the mammography application, the captured information is stored as images which can later be retrieved and analysed by other doctors. However, sometimes it is not sufficient to store the captured information in a human-readable format but information has to be further processed by a computer. In these cases, special optical character recognition software can be used to translate the pen strokes into machine-processable characters. As described earlier, in the Edinburgh festivals guide, no display functionality was available for tourists on the move. Nevertheless, to provide access to comments written by other visitors, the notes stored in XML format were processed by an OCR engine to produce a textual representation of the comment. The textual information was then handled by a text-to-speech engine to enable audio access to other user's comments.

The digital pens can also be used for more *structured annotations* such as form filling. The less interactive process of filling in entire paper forms is also the main application domain of commercially available applications for digital pen and paper applications based on Anoto functionality.

To support the form filling process, the Anoto pens are able to store information about form fields or specific paper buttons within the pen. Unfortunately, this information cannot be modified by the user. However, the form semantics can also be implemented in the server. Then instead of just storing the captured content, it is analysed and matched against the definition of a form. Checkboxes, sliders, text fields and similar components can easily be implemented. For some of the fields, it might even be feasible to apply handwriting recognition. The limited amount of content and prior knowledge of the content type, e.g. that the content will be a number or a date, can significantly improve the accuracy and performance of the handwriting recognition. The information extracted from the forms can then be used to create structured objects as the content of the annotation.

7.2.4 Content Authoring

So far we have discussed interactive paper authoring tools for link authoring where the documents to be augmented already exist and specific regions of these documents are cross-linked or annotated with captured information. However, if the documents to be made interactive do not already exist or are subject to frequent changes, a content authoring approach may be more appropriate.

In content authoring, the paper-based links are no longer authored manually by the publisher or a user of the system, but automatically generated on demand by the publishing framework. This has the major advantage that changes in document content can be handled in a more flexible way since the repositioning of the paper-based links can be done automatically. We now describe the changes that had to be made to the OMSwe publishing framework to support content authoring for the EdFest interactive festivals guide, an application where information is published and accessed through interactive paper as well as possibly through normal web channels such as desktop or voice browsers [127].

Information about the EdFest application is stored in an OMSwe database and can be deployed on different output channels. It is not only possible to generate web pages or voice output based on the information managed by the database but also active paper documents based on a PDF output channel. The content which later should be active on the paper documents is semantically defined in the publishing framework. Each event listed in the EdFest brochure has a fixed structure in terms of title, venue, category, description, ratings etc. which enabled the OMSwe

publishing 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,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. The iServer layering further has been used to define overlapping active areas such as the writing capture area for an event and information which is linked to the same event.

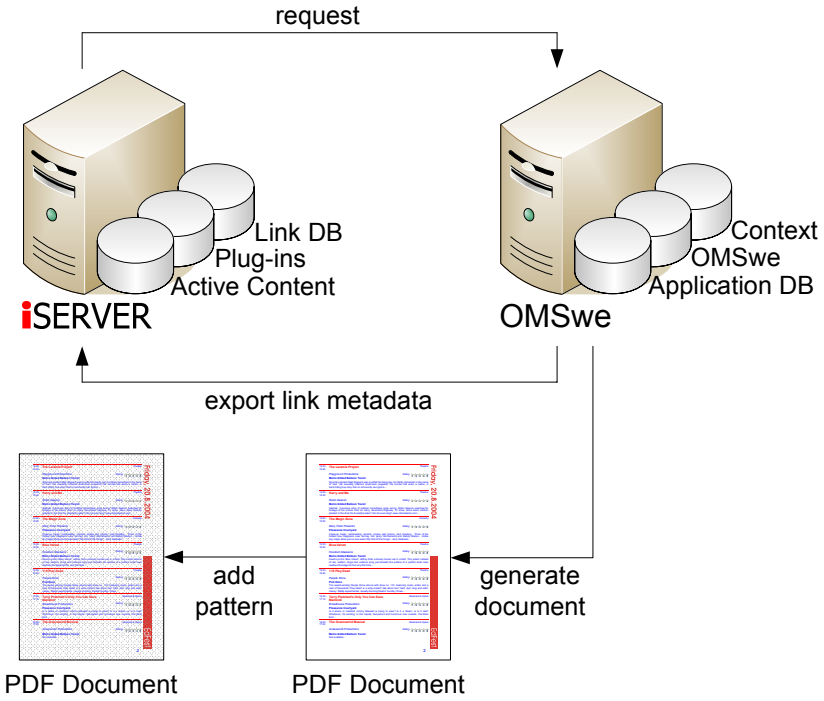


Figure 7.11: Content authoring components

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. The XSL:FO transformations were accomplished by the Apache FOP engine [45]. In a second step, the automatically generated PDF documents could then be covered with the Anoto pattern as shown in Figure 7.11. This two-stage process was necessary since there is restricted access to Anoto publishing software and the pattern has to be generated using an Adobe Acrobat plug-in. However, in the future it should become possible to get access to specific application programming interfaces for generating the pattern which will permit us to generate the document and pattern in a single step. This is particularly important for the next iteration of the EdFest demonstrator where users will be able to print personalised interactive paper 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 and link it to information managed by the OMSwe publishing framework. We therefore had to implement a special output channel in the content publishing framework that generates an XML description of the link information conforming to the iServer XML format introduced in Section 4.3. A simplified version of such an XML export document, defining a single EdFest document that contains just one exemplary link is shown in Figure 7.12. Once this XML file has been imported into the iServer, the link information is stored in its database.

The automatic export from OMSwe to the iServer platform significantly simplifies the rapid prototyping of interactive paper applications as it allows the design of paper documents to be refined in an iterative manner. This raises the issue that different versions of the same printed document may be available and have to be managed by iServer. Such a content-driven authoring and information deployment process for interactive paper would even support varying printed documents for the same digital information printed in different formats.

The definition of paper-based links can not only be based on explicit link metadata stored in the database. Link information could also be derived from meta information such as concept naming so that keywords in a text could be linked automatically, again assuming that we have a mapping from physical location to the semantic concepts in the database. For example, in addition to the content in the form of a document, a publisher could provide a list of keywords together with the link information to be generated for each of these keywords. The system would then automatically check for any occurrences of the keywords within the document and define an active area for each occurrence of the keyword in the published document. Of course these ideas for automatic link generation are not new in that they are, for example, already used in web publishing tools for the generation of websites and as part of general hypermedia systems.

Developing technologies that enable invisible, or nearly invisible, encodings brings us back to the issue of how users know where links are located within a page. When the publishing framework generates a new PDF document, link metadata not only has to be exported to iServer but the links also have to be made visible in the printed document. We investigated various ideas on how users can be made aware of paper-based link sources and targets.

```
<?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 7.12: EdFest content authoring

A first method for link highlighting is to fill out active areas with a specific colour. Different colours may be used for shapes located on different layers. This approach for highlighting active paper areas was applied in the publication annotation application to augment specific parts of research papers. If the active area to be highlighted consists of text only, a different text colour for link visualisation could be used instead of printing coloured shapes. Not only different colours, but also different text styles, such as underlined text, can be used for link visualisation. The problems of paper-based link visualisation are somehow similar to issues of link visualisation on the Web. There, the lack of strict conventions often results in web pages where users are not aware of available links. However, users have come to expect where links might be such as headings and images. The same approach could be used for paper documents by defining link anchors or targets bound to structural components such as titles and images. In some cases it may even make sense not to explicitly mark active regions because it is part of the application that a user has to find the "hidden" information.

Another form of paper-based link highlighting is to use specific pictograms or paper-based buttons. Again, specific colour codings may indicate that these paper buttons are active. The pictogram and paper button approach has been used in the PaperPoint and EdFest applications. We have also experimented with the use of audio feedback to inform users that they have activated links. With the further development of input devices, other forms of audio or even force feedback from the input device itself could be used for link awareness.

Links that are generated dynamically by the system or other users cannot be highlighted in advance. Again, one possibility is that the user points at specific locations without knowing if information is linked to that position. Another possibility that we are investigating is the use of flexible display technologies, such as the ones developed by Acreo, which can be integrated with the paper documents and provide dynamic link feedback by highlighting any active regions. Note that the design of interactive paper documents is still a very new research area and our interactive paper platform provides a flexible framework for experimentation and rapid prototyping of alternative paper-based user interfaces.

While a cross-media content authoring tool especially designed for the OMSwe publishing platform exploits the potential of iServer in an optimal way, we are aware that major content publishers are very reluctant to change their authoring and publishing tools. However, nowadays all major publishing platforms such as Adobe's InDesign or QuarkXpress

offer XML export functionality and therefore the same approach used to export data from OMSwe and import the link metadata into iServer could also be applied to these systems. Nevertheless, for those publishers willing to change their publishing process, in the last section of this chapter we discuss how they could profit from a switch from the traditional publishing process to our content-driven cross-media authoring approach.

7.3 Cross-Media Authoring

After presenting different tools for interactive paper authoring, we now discuss cross-media link authoring for other resource types and present a general iServer authoring tool. For each of the resource plug-ins presented in Section 4.1, we also have to provide the necessary authoring component allowing us to define link source and target anchors within a specific type of media.

The design of an authoring tool for a new resource type involves different steps which we now describe for the example of the iWeb authoring tool for web pages. The first step in building an authoring tool for a new resource type comprises the definition and implementation of a Java application program interface (API) for creating and accessing instances of the new resource type. Parts of the resulting iWeb API for creating new `XHTMLDocument` and `XHTMLSelector` instances are shown in Figure 7.13.

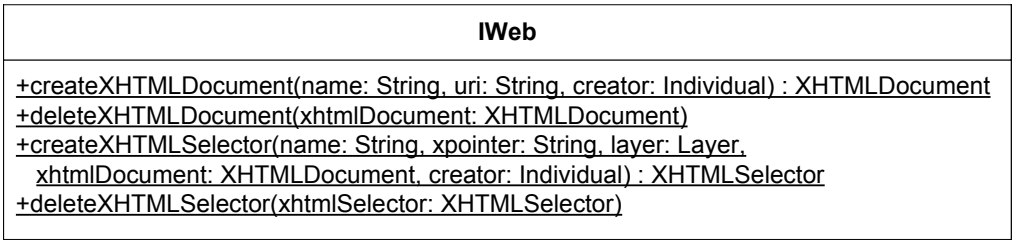


Figure 7.13: iWeb API

While the `IWeb` class provides the functionality to create different iWeb instances, implementations for the corresponding `XHTMLDocument` and `XHTMLSelector` classes have to be provided too. The Java application program interface is a first, even though very low level, authoring tool for the iWeb plug-in. However, this authoring functionality is not intended to be used by publishers or end users in authoring new iWeb links. Rather, the Java API builds the appropriate tool for developers to

implement more sophisticated and user-friendly authoring tools for linking iWeb resources. Note that the iWeb API extends the functionality already provided by iServer and therefore has to be used in combination with the iServer API presented earlier in Section 4.

In a next step, the iServer XML import and export format is extended with XML elements for handling the new iWeb resource and selector data. The resulting `<xhtmlDocument>` and `<xhtmlSelector>` elements are shown in Figure 7.14. Again, the new iWeb XML elements are used in combination with the already existing iServer XML elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<iserver>
  ...
  <xhtmlDocument>
    <name>GlobIS Website</name>
    <uri>http://www.globis.ethz.ch/</uri>
  </xhtmlDocument>
  <xhtmlSelector>
    <name>Mixed-media information</name>
    <xpointer>
      xpointer(string-range(/html[1]/body[1]/div[4], \"\", 1, 9))
    </xpointer>
  </xhtmlSelector>
</iserver>
```

Figure 7.14: iWeb XML interface

The XML API simplifies the authoring of new iWeb links since it is no longer necessary to write and compile any Java code for entering new link metadata. While the XML API is convenient to automatically import iServer link metadata from third-party applications, it is still not the best tool for manual authoring of information by a publisher or users of the cross-media information platform. Therefore, in a final step a resource-specific visual authoring component has to be provided which enables individual users, without any programming experience, to carry out the authoring process in an easy and natural way.

For the iWeb authoring tool dealing with web pages, we decided to integrate the authoring functionality directly into the web browser. Figure 7.15 shows the web browser with the additional authoring features on the left hand side in the form of a sidebar. The main browser window shows a web page with incoming and outgoing link sources and targets

highlighted in different colours representing the different layers. The definition of a new iWeb link involves the selection of a link source and a link target. The selected source and target entities are shown in the sidebar. The user has already defined a link source entity and now selected the words 'advanced database technologies' within the current web page using the mouse. After a right click, an operation — in our case the 'Select Target' operation — is chosen from the context menu to calculate an XPointer expression for the selected words. Finally, by providing a name for the link and pressing the 'Submit' button in the sidebar, a new link is generated. This is done by sending a single request containing all the necessary information to the iServer Web Service and remotely invoking the method required to create a new link.

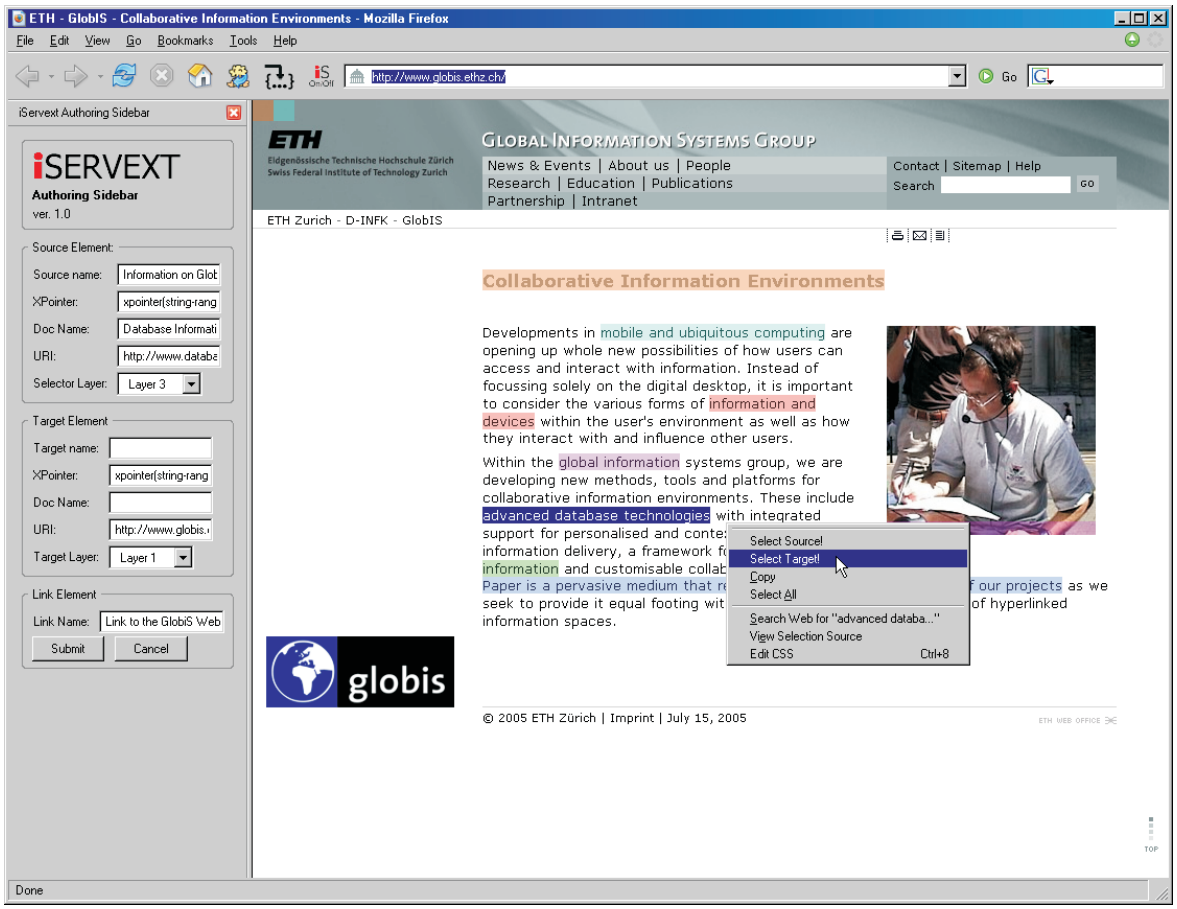


Figure 7.15: iWeb authoring tool

The presented browser extension for iWeb authoring only supports the visual definition of new selectors for web pages. By solely using the browser extension, we cannot define cross-links between arbitrary resources such as a link from parts of a web page to a paper document. Therefore, we decided to implemented a general iServer authoring tool

where the same idea of the resource plug-in mechanism, already used to support new resource types at storage level, is utilised for visual authoring plug-ins. This means we no longer use a single component, such as the iPaper or iWeb links authoring tool, rather than controlling these components via the general iServer authoring tool.

Each new visual authoring component is registered with the iServer authoring tool and is invoked whenever a selector for the corresponding resource type has to be generated. For example, for the iWeb authoring tool this means that the user has to select the type of the desired link source or target entity from a list of available authoring plug-ins. After defining the iWeb selector to be the new link's source, the user may decide to have a paper-based link target. By choosing the iPaper entry from the list of available authoring plug-ins, the iServer authoring tool invokes the corresponding iPaper authoring tool to define the link's target selector. Note that this plug-in mechanism for visual authoring tools can not only be applied to aggregate new link information, but also for browsing existing cross-media links.

The iServer cross-media authoring tool with its main components is shown in Figure 7.16. We have specified the concepts and functionality to be supported by such a cross-media authoring tool and the general iServer functionality has been implemented. This includes the user and layer management and also parts of the link management, such as creating new or deleting existing links. Each resource-specific visual plug-in has to provide two main features based on a well-defined interface: it has to provide a mechanism for defining a selector within the specific resource type and it has to provide a visualisation for this selector.

The user management component shown on the left hand side as well as the layer control at the right hand side are similar to the user and layer management components of the iPaper authoring tool presented earlier in Figure 7.7. However, the difference is that now this functionality is implemented only once and can be used by all visual resource-specific authoring plug-ins.

The `Main View` shows the selected resource with its selectors and can be used for the visualisation of selectors as well as for the authoring of new selectors. The new thing is that the main window view is no longer fixed but will be dynamically loaded from the list of available visual resource plug-ins based on the type of the currently selected resource.

The `Link Browser` shown at the bottom of Figure 7.16 is a new component that helps to browse and analyse the link structure between different resources or selectors. While the `Main View` always presents

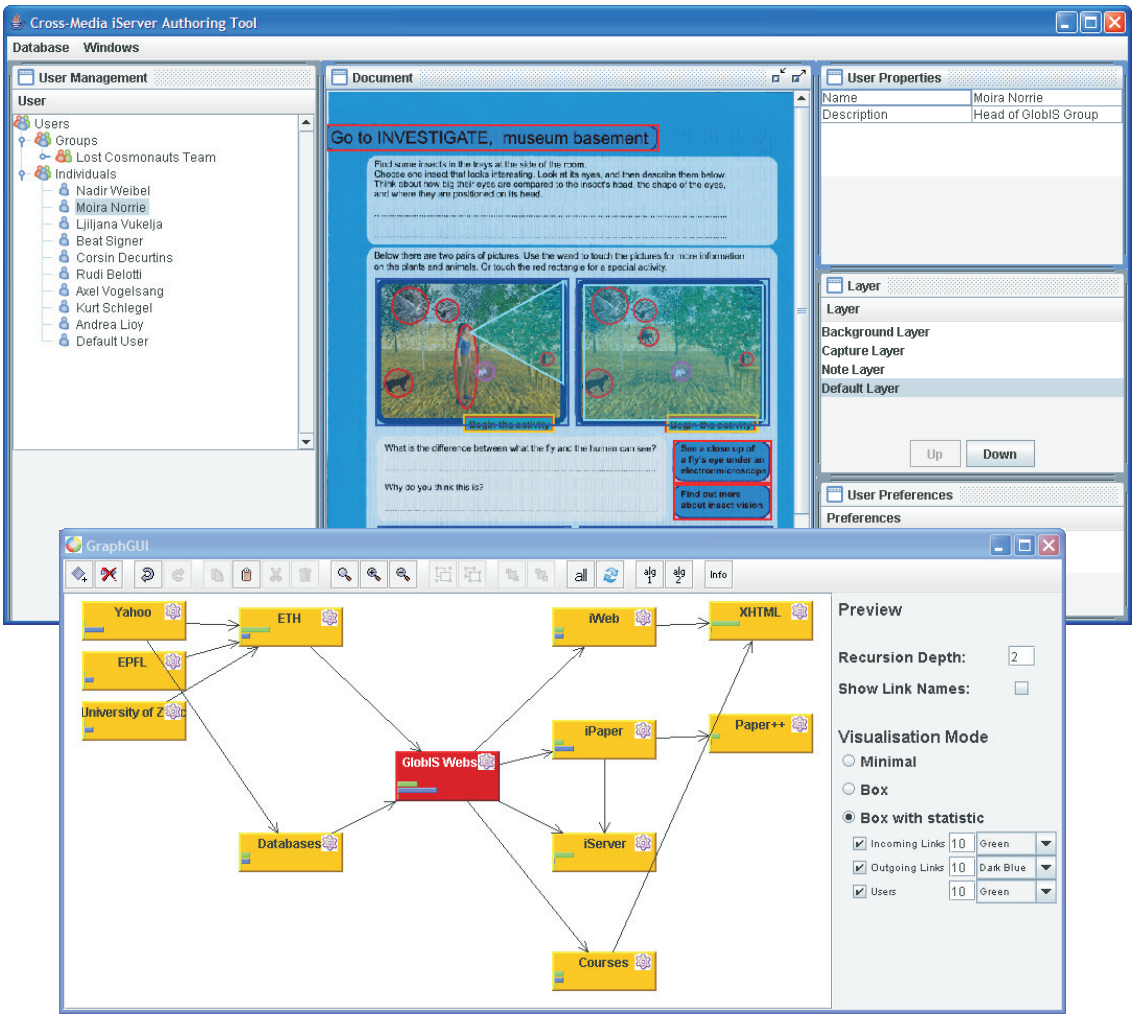


Figure 7.16: iServer authoring tool

a single resource, the `Link Browser` shows the context of the currently selected resource. The selected resource is shown in the centre of the `Link Browser` while entities having incoming or outgoing links to this resource are arranged around the centred resource. By clicking on a neighbouring entity, it becomes the new selection and moves to the centre of the `Link Browser` and the link context is updated. At the same time the `Main View`, after the appropriate visual plug-in has been loaded, shows the new entity.

The `Link Browser` not only allows the iServer link structure to be inspected very easily but it can also be used for changing information. For example, by performing a right-click on a single link and selecting the delete command from the context menu, the selected link is removed from iServer.

The general iServer tool simplifies the implementation of an authoring component for a new resource type since only functionality for the

media-specific authoring, i.e. the definition and visualisation of a selector has to be provided by the plug-in, while the remaining iServer functionality is addressed by the general authoring tool.

7.4 Publisher Case Study

After presenting our interactive paper and cross-media authoring tools let us point out how publishers could profit from such an integrated cross-media approach. It is common for publishers nowadays to produce a package of related materials on different media. For example, the British Broadcasting Corporation (BBC) often produces books and also websites to accompany television documentary series. We chose a specific product of the BBC, a TV series on ocean life called *Blue Planet*, as a case study on cross-media publishing. The series has an associated book and a website with games, quizzes and a fact file on different species [21]. In addition, the series was adopted by the Open University [136] in a course on oceanography and a course textbook was published to be used in conjunction with the TV programmes which are also available on video and DVD.

In the case of the Blue Planet material, the Open University course book has been designed so that there is a close correspondence with the TV material and students can supplement learning by watching clips from the series. At the end of a section of text, there is a recommendation to view all or part of a programme. These recommendations are indicated by a video cassette symbol placed alongside the text as shown in Figure 7.17. In the notes section shown at the bottom of the page, timing information is given for video segments along with a text that either describes what is seen in that part of the video or provides supplementary information. In some cases, the notes may reference other sections of the course text where something mentioned briefly in the video clip is explained in detail.

Currently, the linking between the printed course textbook and the video material is entirely manual in that readers are simply given printed signals to indicate when they should manually activate another media form. By using the interactive paper authoring tool these symbols could be linked to the corresponding video and activated by simply selecting a symbol with a digital pen.

Most of the Blue Planet material is produced as relatively large chunks of information such as 45 minute TV programmes, sections of text, complete quizzes etc. The course textbook links entire sections of

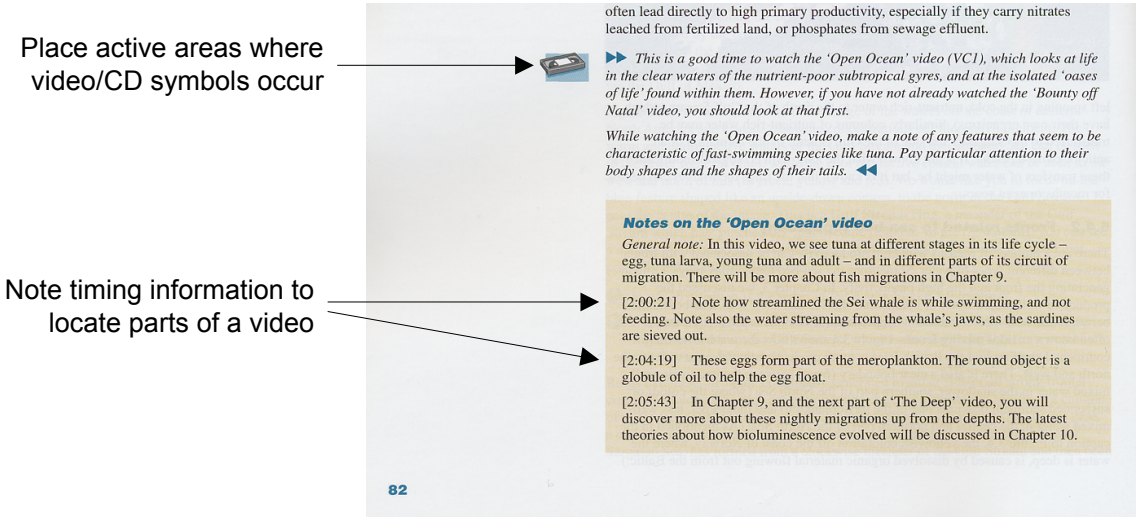


Figure 7.17: Page of Open University course book

text to TV programmes. Since the publishers are aware that this level of granularity for video material is too coarse, they do provide timing information to enable readers to locate parts of a TV programme related to a section of the text or specific themes addressed within the section. To create a richer information environment, a much finer level of granularity must be supported that enables readers to link words, phrases or images to relevant video clips. By applying the iServer movie plug-in, we could not only link to entire movies but address specific parts of the movies, which are, for example, going to be linked from the Open University course book. Furthermore, cross-media links could be defined from the video material to the website or the printed book. During the replay of a video section these links to related material would show up.

Similarly, we note problems with the web-based material in terms of granularity and lack of openness. Quizzes and games are programmed as atomic units and it is not possible to link into or from part of a quiz or game. This might be something useful and worthy of consideration in future publishing scenarios given technologies and tools to support highly-interactive cross media publishing. Also, the fact files providing information on species are quite flat, consisting only of complete sections of text with no links. One can imagine this being replaced by a richer domain-specific application database of information about species, their classifications and geographical locations, linked together and also with links to images, video clips etc.

Since it would be possible for the database to be shared by applications, it might be worth the long-term investment for major publishers

with interests in cross-media publishing to develop databases that would provide a source of material for a number of applications [129, 132]. For example, the BBC are well-known for the high standard of their nature documentary series and over the years there have been many. A well-defined and extensive nature database could be used for all of these programmes and, of course, would provide a rich source of material that the BBC itself could use in the planning and development of their programmes. In fact, according to our sources, they are in the process of developing such a database and this could provide a valuable source of material if integrated into their publishing activities in the future.

The introduction of an application database can potentially provide readers of printed material with access to a wealth of related digital materials. However, like all database development projects, it needs significant investment to do it well. Developing a full application database requires domain expertise and lots of data input. Therefore, this may not be the solution for all forms of cross-media publishing. However, it is important to point out the potential long-term rewards of such an investment, especially when one considers reusability of existing digital information.

In other application domains, it may be the case that databases are already in existence that could support their cross-media publishing. It is therefore also important to note that the application component could be an external database source integrated into the framework, rather than a specially designed component. Various commonly used database integration techniques could be used such as representing concept resources in the iServer framework as active components that request the appropriate data from the external database.

Not only the BBC could profit from an integrated cross-media publishing approach but also their customers would have many more possibilities for working with the available material. For example, while studying the Open University course book, personal paper-based links to other resources could be added. The paper documents and other resources provided by the BBC, including the web pages and the movie material, could be augmented with user-specific information. This only becomes possible because all link information is stored as separate metadata in the iServer database and therefore the original sources do not have to be edited to add new cross-media links.

Finally, based on the distributed iServer version, users could start to share their annotations and links to supplementary resources. Based on iServer's user management, the sharing of information could be realised

in a very controlled manner. For example, a teacher using the material in their class could decide to define a new group for their class and share information within this group only.

Note that such a cross-media publishing solution is not just something of concern to the large commercial content publishers such as the BBC. It is something that most academics do in producing teaching materials such as lecture slides, handouts, manuals, tutorials, exercises, demonstrator applications and course websites. The vision there is that instructors could provide *pre-authored links* between materials that they publish, while students could *dynamically author links* between the materials provided and also to any other materials that they find relevant to a specific course.

7.5 Summary

We have outlined the various forms of authoring activities that need to be supported including link authoring, content authoring, annotations as well as adaptive and dynamic links. Furthermore, collaborative authoring has been introduced as a way of combining a publisher's knowledge with information dynamically acquired by a community of users.

In many cases, the different forms may all be required within a single application domain. For example, in a university learning environment, the instructors may want to generate on-line and printed material which is linked together, but additionally students will want to add and share links both within this set of materials and also to external sources.

After presenting the digital link authoring tool, we discussed different forms of paper-based authoring and potential combinations of digital and paper-based authoring. In addition to the definition of the link source and target anchors in paper documents, pen-based capture of information has been introduced together with the corresponding tools to visualise and replay the captured information. A first prototype for automatic content authoring, where the iServer link metadata is automatically generated by a publishing platform, has been presented.

The general iServer authoring tool integrates existing resource-specific authoring components and thereby simplifies the publishing of cross-media applications by centralising common functionality such as user and layer management. In a case study we have outlined how publishers, but also their customers, could use such a general cross-media authoring approach for new forms of cross-media publishing.

The greatest danger for most of us is not that our aim is too high and we miss it, but that it is too low and we reach it.

Michelangelo

8 Conclusions

After presenting existing interactive paper solutions in the background chapter, we formulated our hypothesis stating that the application of a cross-media information platform can lead to a highly integrated new generation interactive paper framework. In the last five chapters we presented our solution for cross-media information management in the form of the iServer platform and discussed the iPaper plug-in for interactive paper as well as applications of the interactive paper framework. It is now time to provide a critical analysis and evaluation of the achieved work. Following the evaluation of our fundamental concepts for interactive paper and cross-media information spaces, we discuss potential future research directions.

8.1 Evaluation

We started by presenting our model for general cross-media information management. There are several innovative aspects if we compare our model to existing hypermedia and link models. First, our cross-media link model is independent of any specific resource type. The introduction of an abstract `selector` which can be applied to address parts of a `resource` provides maximal flexibility in supporting new types of resources. In contrast, most existing link models already introduce some restrictions at this level by defining a selector in text or XML format.

The concept of layers for controlling overlapping link source and target anchors is the next innovative feature of our link model. It can not only be applied to interactive paper documents, but used by any iServer resource plug-in. Furthermore, the possibility to recursively define links over links and attach arbitrary properties to any `entity`, contributes to the design of a simple but extensible and effective link metamodel.

By providing a Java implementation of the cross-media link model in the form of the iServer platform, we have proven the feasibility of the resource plug-in mechanism and other features such as the concept of multi-layered links. While the main focus has been on the interactive paper plug-in, we implemented three additional resource plug-ins for the iServer platform. Thereby, we already noticed a first advantage of our general cross-media approach since only a few lines of Java code had to be written to support web pages, movies and RFID tags as new resource types. The general iServer functionality for link, layer and user management could be used by all the plug-ins without any modifications.

Various forms of resources such as semantic or active content are provided at the iServer level. Links to semantic resources have proven to be very effective in providing a means to browse digital content and activating links back to paper. On the other hand, active content has been used in all of our recent interactive paper applications. By applying parametrised active components, users can compose new interactive paper applications based on active components without writing a single line of Java code. The only thing to be defined is the behaviour of the active component which can, for example, be configured in an XML file or by a visual authoring tool.

This brings us to the authoring functionality of the iServer cross-media publishing platform. In addition to the Java API, iServer provides XML-based authoring, a Web Service interface and a general graphical authoring tool. The general graphical cross-media authoring tool is another new component not available in most existing hypermedia systems. Existing authoring tools are often designed for one specific media type only. However, within in the iServer authoring tool it is possible to create a link between any two resource types that are supported by a visual authoring plug-in.

Further, the iServer authoring is not limited to a single instance of a cross-media information space but information can be shared across multiple distributed iServer instances based on peer-to-peer (P2P) technologies. To implement this distributed iServer version it was crucial that the underlying data model had a clear notion of data ownership and

access rights could be defined at the entity level. In many other hypermedia systems, user management is not supported as part of the core model and, if necessary, has to be added by specific applications.

Last but not least, to become as flexible as possible in deploying the information on different output channels, the iServer platform uses the eXtensible Information Management Architecture for universal client access based on XML and XSLT technologies. By implementing a set of new default stylesheets, any future client device can be supported with minimal implementation effort. Customised interfaces can then be developed through a refinement of these stylesheets.

The interactive paper framework was realised as an iServer plug-in, which means that it has access to any iServer functionality, including collaborative information sharing or universal client access. All paper-based links are defined on a logical level which means that they are completely independent of any specific hardware solution. The only requirement for an input device to be used within the interactive paper framework is that it can provide its absolute position within a paper document. By defining all necessary input device functionality in separate interfaces, it becomes possible to easily support new input devices. For example, the Natural History application originally was implemented based on the Paper++ inductive pen. However, after adding support for the Nokia Digital Pen to the interactive paper framework, it became possible to control the Natural History application and all other applications with the Nokia Digital Pen. The framework currently supports six types of input devices, including three different types of Paper++ prototype pens. The flexible input device handling was very helpful for rapid interface prototyping within the Paper++ project.

However, the best proof of the flexibility of our interactive paper framework is the fact that more than ten applications have been implemented within a relatively short amount of time. Not only have six different input devices been used in realising these applications, but there are also major differences in the functionality and interaction supported by these applications. The supported functionality includes simple paper-based bookmarks as used in the Nature Encyclopaedia, interactive games as applied in the interactive worksheet for the Natural History Museum as well as writing capture that has, for example, been used in the mammography screening application. A variety of different tasks were supported and the applications were also used in completely different environments. While the Lost Cosmonaut was a fixed art installation, the EdFest guide was used by mobile tourists on the move.

The robustness of our interactive paper framework has been tested in the Generosa Enterprise exhibition where more than 3000 users interacted with the system within two weeks. Another application, the PaperPoint presentation aid has already been integrated into our professional daily work and is used for giving paper-based presentations at our university as well as for presentations at conferences. None of the existing interactive paper solutions presented in the background chapter has the potential to provide such a variety of application domains and forms of interaction with a minimal amount of effort. The interactive paper solution presented in this thesis represents a new form of interactive paper framework that, based on a flexible cross-media information platform, can be used to realise a wide variety of applications and supports the rapid prototyping of new user interfaces and interaction concepts.

A final, very important issue that has been addressed in this thesis is the authoring process of interactive paper applications. We have classified the different forms of authoring and presented different forms of authoring supported by the interactive paper framework. Content authoring has been introduced as a new form of interactive paper authoring where all content is handled by a content management system and the link metadata is generated automatically on demand. Such a content authoring approach has, for example, been used in the EdFest application to generate the booklet automatically. We implemented first prototypes for content authoring but there is an enormous potential for integrating interactive paper authoring functionality into existing applications such as word processing applications.

The field of cross-media publishing for interactive paper is rapidly evolving as the enabling technologies begin to emerge into the marketplace. To date, a study of the literature reveals that the information aspect often tends to be ignored or underplayed in favour of the media technologies. Proposed solutions tend to be based on primitive information models that in many cases perform a simple, one-way mapping from paper to digital media.

The information infrastructure that we have developed has great potential as an experimental platform for the investigation of emerging technologies for interactive paper and its potential uses. By remaining independent of particular hardware solutions and modes of interaction, we are able to easily adapt to both new technologies and applications.

However, the generality and flexibility that we offer has raised even more issues in terms of link visualisation, authoring schemes and the sharing of links among user communities. Further, consideration of various

potential application domains has shown that a wide variety of solutions are required to support different kinds of activities and environments.

The interactive paper framework opens up many new possibilities for forms of information interaction and also radical new approaches to information publishing. Its realisation should therefore not be seen as defining the end of a research project, but rather as defining the beginning of a new research area.

8.2 Vision

The thesis raised a variety of new problems and questions and we will have a look at some of these issues by outlining future research activities. A first new area of research is the design of interactive paper interfaces based on the fact that our framework can support almost any level of granularity and provides features such as the activation and deactivation of specific layers. New forms of design are necessary to make users aware of digital functionality available from interactive paper documents. One of the questions is whether it is possible to define some guidelines for interactive paper links in a similar way to those that exist for link visualisation on web pages. Not only the design of interactive paper interfaces brings up a lot of questions, but also the form and functionality of the digital pens may be modified based on new requirements. For example, until now the digital Anoto pens were almost exclusively used for writing capture. However, we use them in a more interactive manner where the browsing and capture of information are mixed. This brings up the problem that a user has to be aware that the pen now has two different modes for browsing and capturing and may behave differently based on the currently active mode. Specific modifications to the design of the pen, for example additional LEDs or a retractable stylus, could be investigated to provide contextual information about the pen's current mode.

Another research area is the design and development of integrated content authoring tools for interactive paper applications. Here the vision is that any application in the future has some support for interactive paper and can be used to author and print information on interactive paper. From the printouts, users can directly interact with the digital application, in the same way that the PaperPoint application already supports this functionality. Of course there are many potential applications ranging from drawing tools to word processing software for

this kind of paper and digital application integration. Another major application domain, that has already its commercial application, is the digital processing of information written in paper forms.

An obvious possibility for future directions is the development of new iServer resource plug-ins. With each new plug-in, the cross-media platform becomes more powerful because the plug-in can be used with any existing or future application. Since the associative linking provides some very natural functionality for relating information that is not available in today's hierarchical file systems, we could even go a step further and use the iServer cross-media information management platform as a replacement for today's file systems. By using multiple classification, we could still classify information in folders but in addition define cross-media links between arbitrary resources. Based on the distributed iServer version, information stored in this personal cross-media information space could easily be shared with other users based on well controlled access rights.

Last but not least, the concept of active components has great potential for interaction design. By designing a customised authoring tool for active components, we could support users with limited programming skills in designing their own interactive cross-media applications, such as the Lost Cosmonaut or the Generosa Enterprise. The constantly growing number of active components providing specific functionality will enormously simplify the development of future interactive paper and general cross-media applications.

A
iServer Schema

```
type entity (
  name        :  string;
  properties  :  set of (string, string);
);

type link subtype of entity (
);

type selector subtype of entity (
);

type resource subtype of entity (
);

type user (
  name        :  string;
  description :  string;
);

type individual subtype of user (
  login       :  string;
  password    :  string;
);
```

```
type group subtype of user (
);

type layer (
  name        :  string;
);

type medium subtype of resource (
  description :  string;
  content     :  mime;
);

type textComp subtype of resource (
  content     :  string;
);

type complexResource subtype of resource (
);

type activeComponent subtype of resource (
  identifier  :  string;
  timeout     :  integer;
);

type parameter (
  key         :  string;
  value       :  string;
);

collection AccessibleTo     :  set of (entity, user);
collection ActiveComponents :  set of activeComponent;
collection ActiveLayers     :  set of layer;
collection Containers       :  set of resource;
collection Contains         :  set of (resource, resource);
collection CreatedBy        :  set of (entity, individual);
collection Entities         :  set of entity;
collection Groups           :  set of group;
collection HasMembers       :  set of (group, user);
collection HasPreferences   :  set of (user, parameter);
```

```
collection HasProperties    :  set of (entity, parameter);
collection HasSource        :  set of (link, entity);
collection HasTarget        :  set of (link, entity);
collection InaccessibleTo   :  set of (entity, user);
collection Individuals      :  set of individual;
collection Layers           :  ranking of layer;
collection Links            :  set of link;
collection Media            :  set of medium;
collection Movies           :  set of medium;
collection OnLayer          :  set of (selector, layer);
collection Preferences      :  set of parameter;
collection Properties       :  set of parameter;
collection RefersTo         :  set of (selector, resource);
collection Resources        :  set of resource;
collection Selectors        :  set of selector;
collection Texts            :  set of textComp;
collection Users            :  set of user;
collection Webpages         :  set of medium;

constraint AccessibleTo
  association from Entities (0:*) to users (0:*);
constraint Contains
  association from Containers (1:*) to Resources (0:*);
constraint CreatedBy
  association from Entities (1:1) to Individuals (0:*);
constraint HasMembers
  association from Groups (0:*) to Users (0:*);
constraint HasPreferences
  association from Users (0:*) to Preferences (0:*);
constraint HasProperties
  association from Entities (0:*) to Properties (0:*);
constraint HasSource
  association from Links (1:*) to Entities (0:*);
constraint HasTarget
  association from Links (1:*) to Entities (0:*);
constraint InaccessibleTo
  association from Entities (0:*) to Users (0:*);
constraint OnLayer
  association from Selectors (1:1) to Layers (0:*);
```

```
constraint RefersTo
  association from Selectors (1:1) to Resources (0:*);

constraint ActiveComponents subcollection of Resources;
constraint ActiveLayers     subcollection of Layers;
constraint Containers       subcollection of Resources;
constraint Groups           subcollection of Users;
constraint Individuals      subcollection of Users;
constraint Links            subcollection of Entities;
constraint Media            subcollection of Resources;
constraint Movies           subcollection of Media;
constraint Resources        subcollection of Entities;
constraint Selectors        subcollection of Entities;
constraint Texts            subcollection of Resources;
constraint Webpages         subcollection of Media;

constraint (Resources and Selectors and Links) partition Entities;
constraint (Groups and Individuals) partition Users;
```

B
iPaper Schema

```
type page subtype of resource (
  number    :  integer;
  size      :  dimension;
);

type shape subtype of selector (
);

type rectangle subtype of shape (
  upperLeft :  point;
  size      :  dimension;
);

type circle subtype of shape (
  centre    :  point;
  radius    :  integer;
);

type ellipse subtype of shape (
  centre    :  point;
  width     :  integer;
  height    :  integer;
);
```

```
type polygon subtype of shape (
  points     :  ranking of point;
);

type complexShape subtype of shape (
);

type document subtype of complexResource (
  id         :  string;
  size       :  dimension;
  content    :  mime;
);

type dimension (
  width      :  integer;
  height     :  integer;
);

type point (
  x          :  integer;
  y          :  integer;
);

collection Circles        :  set of circle;
collection Dimensions     :  set of dimension;
collection ComplexShapes  :  set of complexShape;
collection ContainsPages  :  set of (document, page);
collection Documents      :  set of document;
collection Ellipses       :  set of ellipse;
collection HasShapes      :  set of (complexShape, shape);
collection OnPage         :  set of (shape, page);
collection Pages          :  set of page;
collection Points         :  set of point;
collection Polygons       :  set of polygon;
collection Rectangles     :  set of rectangle;
collection Shapes         :  set of shape;

constraint ContainsPages
  association from Documents (0:*) to Pages (1:1);
```

```
constraint HasShapes
  association from ComplexShapes (2:*) to Shapes (0:*);
constraint OnPage
  association from Shapes (1:1) to Pages (0:*);

constraint Circles       subcollection of Shapes;
constraint ComplexShapes subcollection of Shapes;
constraint ContainsPages subcollection of Contains;
constraint Documents     subcollection of Containers;
constraint Ellipses      subcollection of Shapes;
constraint OnPage        subcollection of RefersTo;
constraint Pages         subcollection of Resources;
constraint Polygons      subcollection of Shapes;
constraint Rectangles    subcollection of Shapes;
constraint Shapes        subcollection of Selectors;

constraint (ComplexShapes and Circles and Ellipses and Polygons
  and Rectangles) partition Shapes;
```

C
iServer XML Format

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
<xs:element name="iserver">
  <xs:complexType>
   <xs:choice maxOccurs="unbounded">
    <xs:element name="link" type="linkType"
      minOccurs="0"maxOccurs="unbounded"/>
    <xs:element name="individual"
      minOccurs="0"maxOccurs="unbounded">
     <xs:complexType>
      <xs:complexContent>
       <xs:extension base="individualType"/>
      </xs:complexContent>
     </xs:complexType>
    </xs:element>
    <xs:element name="group" minOccurs="0" maxOccurs="unbounded">
     <xs:complexType>
      <xs:complexContent>
       <xs:extension base="groupType"/>
      </xs:complexContent>
     </xs:complexType>
    </xs:element>
    <xs:element name="layer" type="layerType"
      minOccurs="0"maxOccurs="unbounded"/>
    <xs:element name="textComp"
      minOccurs="0"maxOccurs="unbounded">
```

```
      <xs:complexType>
       <xs:complexContent>
        <xs:extension base="textCompType"/>
       </xs:complexContent>
      </xs:complexType>
     </xs:element>
     <xs:element name="medium" type="mediumType"
       minOccurs="0"maxOccurs="unbounded"/>
     <xs:element name="activeComponent" type="activeComponentType"
       minOccurs="0"maxOccurs="unbounded"/>
     <xs:element name="parameter" type="parameterType"
       minOccurs="0"maxOccurs="unbounded"/>
     <xs:element name="resource" type="resourceType"
       minOccurs="0"maxOccurs="unbounded"/>
     <xs:element name="selector" type="selectorType"
       minOccurs="0"maxOccurs="unbounded"/>
    </xs:choice>
   </xs:complexType>
</xs:element>

<xs:complexType name="entityType">
  <xs:sequence>
   <xs:element name="name" type="xs:string"/>
   <xs:element name="creator" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="individual" type="individualType"/>
     </xs:sequence>
     <xs:attribute name="individual" type="xs:IDREF" use="optional"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="authorised" minOccurs="0">
    <xs:complexType>
     <xs:choice maxOccurs="unbounded">
      <xs:element name="individual" type="individualType"
        minOccurs="0"maxOccurs="unbounded"/>
      <xs:element name="group" type="groupType"
        minOccurs="0"maxOccurs="unbounded"/>
     </xs:choice>
     <xs:attribute name="users" type="xs:IDREFS" use="optional"/>
    </xs:complexType>
   </xs:element>
   <xs:element name="unauthorised" minOccurs="0">
    <xs:complexType>
     <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element name="individual" type="individualType"
        minOccurs="0"maxOccurs="unbounded"/>
      <xs:element name="group" type="groupType"
```

```
        minOccurs="0"maxOccurs="unbounded"/>
      </xs:choice>
      <xs:attribute name="users" type="xs:IDREFS" use="optional"/>
     </xs:complexType>
    </xs:element>
    <xs:element name="properties" minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="parameter" type="parameterType"
        maxOccurs="unbounded"/>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
   <xs:attribute name="id" type="xs:ID" use="optional"/>
   <xs:attribute name="creator" type="xs:IDREF" use="optional"/>
   <xs:attribute name="properties" type="xs:IDREFS" use="optional"/>
   <xs:attribute name="authorised" type="xs:IDREFS" use="optional"/>
   <xs:attribute name="unauthorised" type="xs:IDREFS" use="optional"/>
</xs:complexType>

<xs:complexType name="individualType">
  <xs:complexContent>
   <xs:extension base="userType">
    <xs:sequence>
     <xs:element name="login" type="xs:string"/>
     <xs:element name="password" type="xs:string"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="userType">
  <xs:sequence>
   <xs:element name="name" type="xs:string"/>
   <xs:element name="description" type="xs:string"/>
   <xs:element name="preferences" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="parameter" type="parameterType"
       maxOccurs="unbounded"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
   <xs:element name="members" minOccurs="0">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="individual" type="individualType"
```

```
        minOccurs="0"maxOccurs="unbounded"/>
      <xs:element name="group" type="groupType"
        minOccurs="0"maxOccurs="unbounded"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
  </xs:sequence>
  <xs:attribute name="id" type="xs:ID" use="optional"/>
  <xs:attribute name="preferences" type="xs:IDREFS" use="optional"/>
</xs:complexType>

<xs:complexType name="groupType">
  <xs:complexContent>
   <xs:extension base="userType">
    <xs:attribute name="members" type="xs:IDREFS" use="optional"/>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="linkType">
  <xs:complexContent>
   <xs:extension base="entityType">
    <xs:sequence>
     <xs:element name="sources" minOccurs="0">
      <xs:complexType>
       <xs:sequence>
        <xs:element name="entity" type="entityType"
          maxOccurs="unbounded"/>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
     <xs:element name="targets" minOccurs="0">
      <xs:complexType>
       <xs:sequence>
        <xs:element name="entity" type="entityType"
          maxOccurs="unbounded"/>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
    <xs:attribute name="sources" type="xs:IDREFS" use="optional"/>
    <xs:attribute name="targets" type="xs:IDREFS" use="optional"/>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="layerType">
  <xs:sequence>
```

```
    <xs:element name="name" type="xs:string"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:ID" use="optional"/>
</xs:complexType>

<xs:complexType name="selectorType">
  <xs:complexContent>
   <xs:extension base="entityType">
    <xs:sequence minOccurs="0">
     <xs:element name="layer" type="layerType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="layer" type="xs:IDREF" use="optional"/>
    <xs:attribute name="resource" type="xs:IDREF" use="required"/>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="resourceType">
  <xs:complexContent>
   <xs:extension base="entityType"/>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="parameterType">
  <xs:sequence>
   <xs:element name="key" type="xs:string"/>
   <xs:element name="value" type="xs:string"/>
  </xs:sequence>
  <xs:attribute name="id" type="xs:ID" use="optional"/>
</xs:complexType>

<xs:complexType name="mediumType">
  <xs:complexContent>
   <xs:extension base="resourceType">
    <xs:sequence>
     <xs:element name="description" type="xs:string"/>
     <xs:element name="content" type="xs:string"/>
     <xs:element name="collection" type="xs:string" minOccurs="0"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="textCompType">
  <xs:complexContent>
   <xs:extension base="resourceType">
    <xs:sequence>
     <xs:element name="content" type="xs:string"/>
```

```
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="activeComponentType">
  <xs:complexContent>
   <xs:extension base="resourceType">
    <xs:sequence>
     <xs:element name="identifier" type="xs:string"/>
     <xs:element name="timeout" type="xs:int" minOccurs="0"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

</xs:schema>
```

D Software

As part of this thesis three major Java software components, the `iserver`, `ipaper` and `sigtec` packages have been developed. Since it is out of the scope of this thesis do discuss all classes that have been implemented in detail, we present an overall package structure.

D.1 iserver Packages

The `iserver` software component, shown in Figure D.1, provides all functionality for cross-media link management. This includes the authoring as well as the distribution of information.

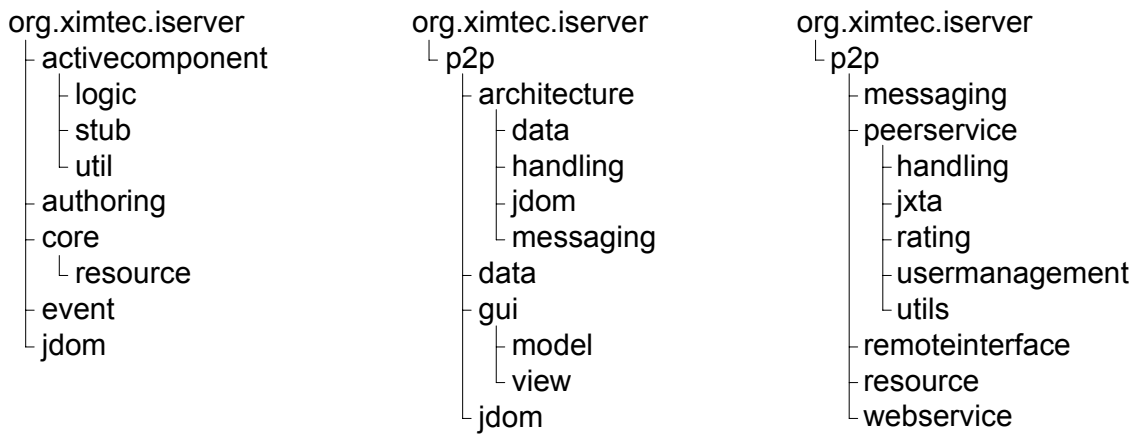


Figure D.1: iserver packages

The core concepts and functionality, such as the representation of an `Entity`, `Link`, `Resource`, `Selector`, `Individual` and `Group`, are implemented in the `core` package. In addition, the `core` package contains the `IServer` class providing the main functionality to create new objects and access existing information. The `activecomponent` package provides the active component framework whereas authoring functionality was implemented in the `authoring` package. Last but not least any distribution-specific functionality, including link rating and the provision of a Web Service interface, is handled by the `p2p` packages.

D.2 ipaper Packages

In Figure D.2 we present the `ipaper` packages providing the interactive paper plug-in functionality for iServer. In addition, the packages contain an authoring tool for interactive paper as well as various demonstrator applications.

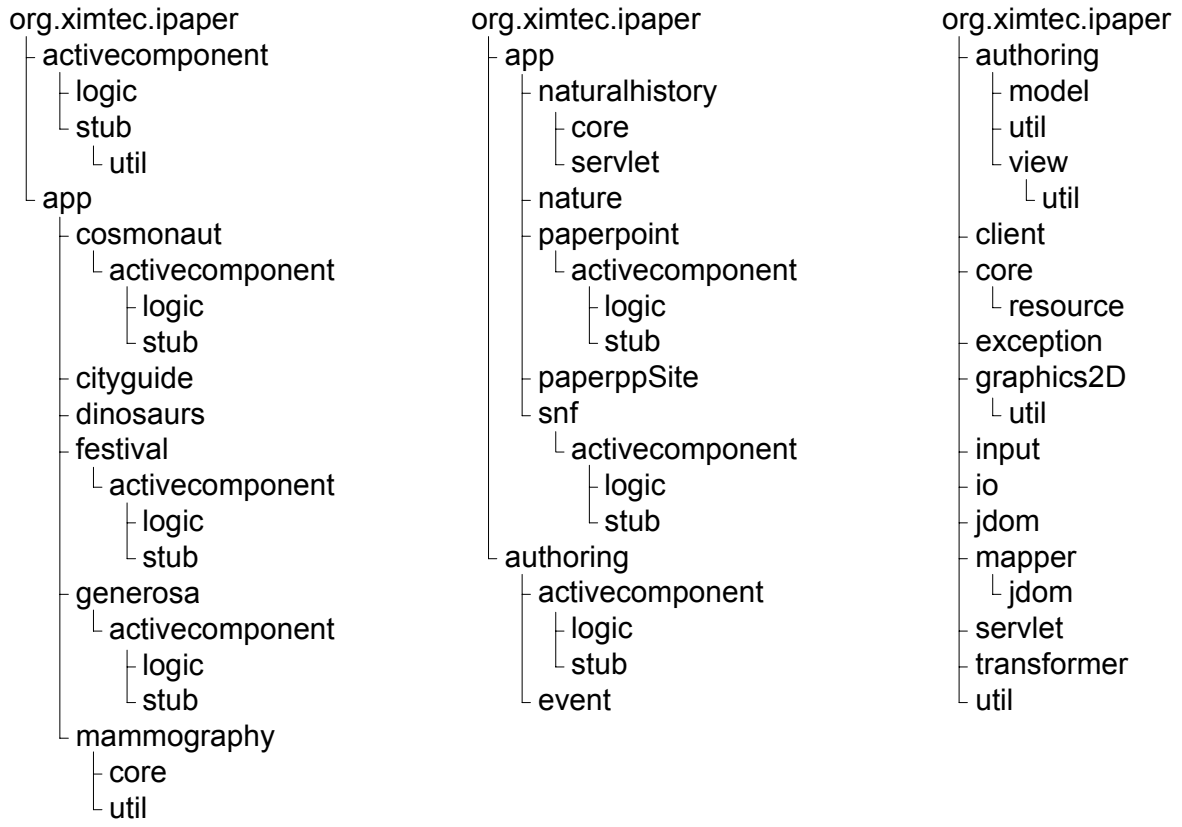


Figure D.2: ipaper packages

The core classes required by the iPaper plug-in are located in the `core` package. This includes the appropriate representations for `Document`

and `Page` instances. The different selector shapes are handled by the `graphics2D` package. Pen-specific implementations of the `InputDevice` interface, including support for the Nokia Digital Pen, multiple Paper++ inductive pen versions and an Agilent barcode reader, can be found in the `io` package. Different document and page handlers, for example the `DocHandlerConnectedPages` based on the algorithm described earlier in Section 5.4, are implemented in the `mapper` package. The interactive paper authoring tool is located in the `authoring` package whereas a variety of paper-based active components, for example for pen-based writing capture, paper-buttons etc., can be found in the `activecomponent` package. Finally, the implementation of the applications described in Chapter 6 can be found in the corresponding `app` subpackages.

D.3 sigtec Packages

Over the past few years we implemented a lot of functionality that can be applied not only in an iServer and interactive paper context, but is applicable for other applications as well. This functionality therefore was implemented in the more general `sigtec` software package shown in Figure D.3.

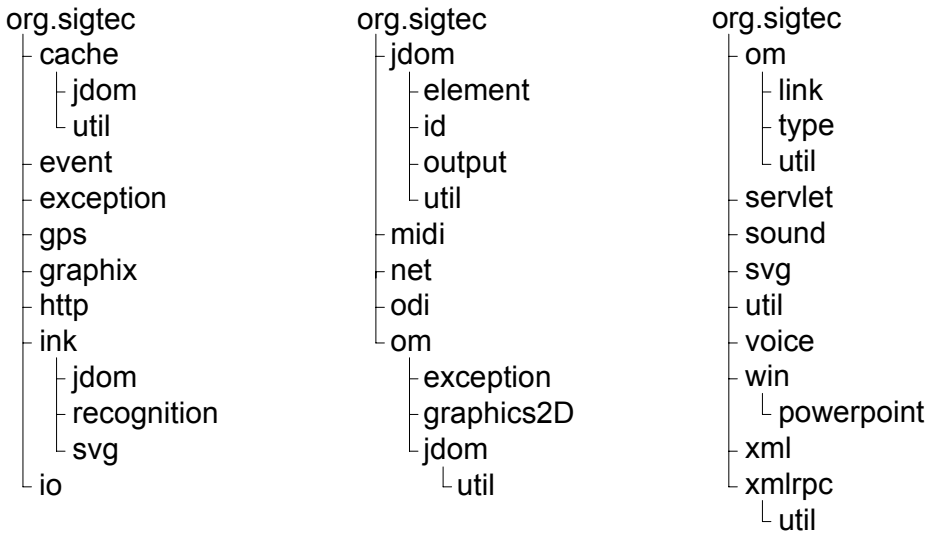


Figure D.3: sigtec packages

Some general functionality for processing pen-based input is located in the `ink` package. This includes trace detection on pen-based input, integration of third-party tools for character recognition as well as rendering functionality for pen-based input and XML export of captured

information. Extensions and tools for the OMS Java platform can be found in the `om` package. Finally, there are various tool classes for handling graphics and sound in the `graphix`, `sound` and `voice` packages.

Bibliography

[1] G. D. Abowd, C. G. Atkeson, A. Feinstein, C. Hmelo, R. Kooper, S. Long, N. Sawhney, and M. Tani. Teaching and Learning as Multimedia Authoring: The Classroom 2000 Project. In *Proceedings of ACM Multimedia 96, 4th International Conference on Multimedia*, pages 187–198, Boston, USA, November 1996.

[2] M. J. Adler and C. Van Doren. *How to Read a Book*. Revised edition, Simon and Schuster, New York, 1972.

[3] D. Aliakseyeu and J.-B. Martens. Physical Paper as the User Interface for an Architectural Design Tool. In *Proceedings of INTERACT 2001, 8th Conference on Human-Computer Interaction*, pages 680–681, Tokyo, Japan, July 2001.

[4] D. Aliakseyeu, J.-B. Martens, S. Subramanian, M. Vroubel, and W. Wesselink. Visual Interaction Platform. In *Proceedings of INTERACT 2001, 8th Conference on Human-Computer Interaction*, pages 232–239, Tokyo, Japan, July 2001.

[5] Amaya, W3C's Editor/Browser, World Wide Web Consortium, http://www.w3.org/Amaya/.

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

[7] T. Arai, D. Aust, and S. E. Hudson. PaperLink: A Technique for Hyperlinking from Real Paper to Electronic Content. In *Proceedings of CHI '97, ACM Conference on Human Factors in Computing Systems*, pages 327–334, Atlanta, USA, March 1997.

[8] T. Arai, K. Machii, and S. Kuzunuki. Retrieving Electronic Documents with Real-World Objects on InteractiveDESK. In *Proceedings of UIST '95, 8th Annual ACM Symposium on User Interface*

Software and Technology, pages 37–38, Pittsburgh, USA, November 1995.

[9] ARToolKit, http://www.hitl.washington.edu/artoolkit/.

[10] M. Baang, A. Larsson, and H. Eriksson. Paper-Based Medical Ubiquitous Computing. In *HMI Workshop*, Söderköpings Brunn, Sweden, May 2003.

[11] M. Back and J. Cohen. Page Detection Using Embedded Tags. In *Proceedings of UIST 2000, 13th Annual ACM Symposium on User Interface Software and Technology*, pages 159–160, San Diego, USA, November 2000.

[12] M. Back, J. Cohen, R. Gold, S. Harrison, and S. Minneman. Listen Reader: An Electronically Augmented Paper-Based Book. In *Proceedings of CHI 2001, ACM Conference on Human Factors in Computing Systems*, pages 23–29, Seattle, USA, March 2001.

[13] C. Bailey, W. Hall, D. E. Millard, and M. J. Weal. Towards Open Adaptive Hypermedia. In *Proceedings of AH 2002, 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems*, pages 36–46, Malaga, Spain, May 2002.

[14] D. Barreau and B. A. Nardi. Finding and Reminding: File Organization from the Desktop. *ACM SIGCHI Bulletin*, 27(3):39–45, July 1995.

[15] R. Belotti, C. Decurtins, M. Grossniklaus, M. C. Norrie, and A. Palinginis. Modelling Context for Information Environments. In *Proceedings of UMICS 2004, 2nd International Workshop on Ubiquitous Mobile Information and Collaboration Systems*, pages 43–56, Riga, Latvia, June 2004.

[16] R. Belotti, C. Decurtins, M. C. Norrie, B. Signer, and L. Vukelja. Experimental Platform for Mobile Information Systems. In *Proceedings of MobiCom 2005, 11th Annual International Conference on Mobile Computing and Networking*, pages 258–269, Cologne, Germany, August 2005.

[17] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, May 2001.

[18] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and Magic Lenses: The See-Through Interface. In *Proceedings of ACM SIGGRAPH 93, International Conference on Computer Graphics and Interactive Techniques*, pages 73–80, Anaheim, USA, August 1993.

[19] M. Billinghurst, H. Kato, and I. Poupyrev. The MagicBook: A Transitional AR Interface. *Computers and Graphics*, 25(5):745–753, October 2001.

[20] M. Billinghurst, H. Kato, and I. Poupyrev. The MagicBook: Moving Seamlessly between Reality and Virtuality. *IEEE Computer Graphics and Applications*, 21(3):6–8, 2001.

[21] The Blue Planet: A Natural History of the Oceans, BBC Science and Nature, http://www.bbc.co.uk/nature/blueplanet/.

[22] P. Bolliger. Active Components for iServer: In Consideration of Paper-Based Authoring. Semester project, Institute for Information Systems, ETH Zurich, 2004.

[23] D. Brookshier, D. Govoni, N. Krishnan, and J. C. Soto. *JXTA: Java P2P Programming*. Sams, March 2002.

[24] B. Brown and M. Chalmers. Tourism and Mobile Technology. In *Proceedings of ECSCW 2003, 8th European Conference on Computer Supported Cooperative Work*, pages 335–355, Helsinki, Finland, September 2003.

[25] H. Brown, R. Harding, S. Lay, P. Robinson, D. Sheppard, and R. Watts. Active Alice: Using Real Paper to Interact with Electronic Text. In *Proceedings of EP 98, 7th International Conference on Electronic Publishing*, pages 407–419, Saint-Malo, France, April 1998.

[26] J. S. Brown and P. Duguid. *The Social Life of Information*. Harvard Business School Press, February 2000.

[27] P. Brusilovsky. Methods and Techniques of Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, 6(2-3):87–129, 1996.

[28] V. Bush. As We May Think. *Atlantic Monthly*, 176(1):101–108, July 1945.

[29] O. Cakmakci, F. Bérard, and J. Coutaz. Back to Paper: A Technique to Facilitate Access and Capture in a Ubicomp Setting. Technical report, LIPS-IMAG, 2002.

[30] P. Calado, M. Cristo, E. Moura, N. Ziviani, B. Ribeiro-Neto, and M. A. Gonçalves. Combining Link-Based and Content-Based Methods for Web Document Classification. In *Proceedings of CIKM 2003, 12th International Conference on Information and Knowledge Management*, pages 394–401, New Orleans, USA, November 2003.

[31] A New Technology for Addressing the Shortcomings of Paper-Based Workflows. CAP Ventures, Inc., December 2003.

[32] C-Channel AG, Switzerland, http://www.c-channel.ch.

[33] WizCom Technologies, Ltd., http://www.wizcomtech.com.

[34] M. Chalmers, K. Rodden, and D. Brodbeck. The Order of Things: Activity-Centred Information Access. In *Proceedings of WWW7, 7th International World Wide Web Conference*, pages 359–368, Brisbane, Australia, April 1998.

[35] B. G. Christensen and F. A. Hansen. XLink - Linking the Web and Open Hypermedia. In *Proceedings of OHS 2002, 8th International Workshop on Open Hypermedia Systems*, pages 9–18, Maryland, USA, June 2002.

[36] E. F. Churchill and L. Nelson. Tangibly Simple, Architecturally Complex: Evaluating a Tangible Presentation Aid. In *Proceedings of CHI 2002, ACM Conference on Human Factors in Computing Systems*, pages 750–551, Minneapolis, USA, April 2002.

[37] Strip-Reader, Datasound GmbH, http://www.datasound.de.

[38] A. de Spindler. Distributed Collaborative Information Environment Based on iServer. Diploma thesis, Institute for Information Systems, ETH Zurich, 2005.

[39] C. Decurtins, M. C. Norrie, and B. Signer. Digital Annotation of Printed Documents. In *Proceedings of CIKM 2003, 12th International Conference on Information and Knowledge Management*, pages 552–555, New Orleans, USA, November 2003.

[40] C. Decurtins, M. C. Norrie, and B. Signer. Putting the Gloss on Paper: A Framework for Cross-Media Annotation. *New Review of Hypermedia and Multimedia*, 9:35–57, 2003.

[41] S. J. DeRose. XML Linking. *ACM Computing Surveys*, 31(4), December 1999.

[42] A. Diener. Authoring Tool for Digitally Augmented Paper. Semester project, Institute for Information Systems, ETH Zurich, 2004.

[43] M. Dymetman and M. Copperman. Intelligent Paper. In *Proceedings of EP 98, 7th International Conference on Electronic Publishing*, pages 392–406, Saint-Malo, France, April 1998.

[44] K. M. Everitt, S. R. Klemmer, R. Lee, and J. A. Landay. Two Worlds Apart: Bridging the Gap Between Physical and Virtual Media for Distributed Design Collaboration. In *Proceedings of CHI 2003, ACM Conference on Human Factors in Computing Systems*, pages 553–560, Fort Lauderdale, USA, April 2003.

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

[46] I. R. Forman and N. Forman. *Java Reflection in Action*. Manning Publications, 2004.

[47] B. Forster. Writing to the Future. *Computerworld*, February 2001.

[48] M. Fraser, D. Stanton, K. H. Ng, S. Benford, C. O'Malley, J. Bowers, G. Taxen, K. Ferris, and J. Hindmarsh. Assembling History: Achieving Coherent Experiences with Diverse Technologies. In *Proceedings of ECSCW 2003, 8th European Conference on Computer Supported Cooperative Work*, pages 179–198, Helsinki, Finland, September 2003.

[49] Fruits ApS, Denmark, http://www.fruits.dk.

[50] D. Gambetta. Can We Trust Trust? In *Trust: Making and Breaking Cooperative Relations*, chapter 13, pages 213–237. Department of Sociology, University of Oxford, 2000.

[51] Generosa Enterprise, http://www.generosa.ethz.ch.

[52] M. Gertz and K.-U. Sattler. Integrating Scientific Data through External, Concept-Based Annotations. In *Proceedings of DiWeb 2002, 2nd International Workshop on Data Integration over the Web*, pages 87–101, Toronto, Canada, May 2002.

[53] M. Gertz, K.-U. Sattler, F. Gorin, M. Hogarth, and J. Stone. Annotating Scientific Images: A Concept-Based Approach. In *Proceedings of SSDBM 2002, 14th International Conference on Scientific Database Management*, pages 59–68, Edinburgh, Scotland, July 2002.

[54] M. Götze, S. Schlechtweg, and T. Strothotte. The Intelligent Pen: Toward a Uniform Treatment of Electronic Documents. In *Proceedings of Smart Graphics 2002, 2nd International Symposium on Smart Graphics*, pages 129–135, Hawthorne, USA, June 2002.

[55] J. Graham and J. J. Hull. Video Paper: A Paper-Based Interface for Skimming and Watching Video. In *Proceedings of ICCE 2002, International Conference on Consumer Electronics*, pages 214–215, Los Angeles, USA, June 2002.

[56] A. Grasso, A. Karsenty, and M. Susani. Augmenting Paper to Enhance Community Information Sharing. In *Proceedings of DARE 2000, Conference on Designing Augmented Reality Environments*, pages 51–62, Elsinore, Denmark, April 2000.

[57] K. Grønbæk, J. Kristensen, P. Ørbæk, and M. A. Eriksen. Physical Hypermedia: Organizing Collections of Mixed Physical and Digital Material. In *Proceedings of Hypertext 2003, 14th ACM Conference on Hypertext and Hypermedia*, pages 10–19, Nottingham, UK, August 2003.

[58] K. Grønbæk, P. P. Vestergaard, and P. Ørbæk. Towards Geo-Spatial Hypermedia: Concepts and Prototype Implementation. In *Proceedings of Hypertext 2002, 13th ACM Conference on Hypertext and Hypermedia*, pages 117–126, College Park, USA, June 2002.

[59] M. Grossniklaus and M. C. Norrie. Information Concepts for Content Management. In *Proceedings of DASWIS 2002, International Workshop on Data Semantics in Web Information Systems*, pages 150–159, Singapore, Republic of Singapore, December 2002.

[60] F. Guimbretière. Paper Augmented Digital Documents. In *Proceedings of UIST 2003, 16th Annual ACM Symposium on User Interface Software and Technology*, pages 51–60, Vancouver, Canada, November 2003.

[61] C. Haddad, K. Bedell, P. Brown, D. Srinivas, and D. Almaer. *Programming Apache Axis*. O'Reilly & Associates, March 2004.

[62] W. Hall, H. Davis, and G. Hutchings. *Rethinking Hypermedia: The Microcosm Approach*. Kluwer Academic Publishers, May 1996.

[63] G. Hardock, G. Kurtenbach, and W. Buxton. A Marking Based Interface For Collaborative Writing. In *Proceedings of UIST '93, 6th Annual ACM Symposium on User Interface Software and Technology*, pages 259–266, Atlanta, USA, November 1993.

[64] M. Hartswood, R. Procter, M. Rouncefield, and R. Slack. Performance Management in Breast Screening: A Case Study of Professional Vision and Ecologies of Practice. *Journal of Cognition, Technology and Work*, 14(2):91–100, 2001.

[65] M. Hartswood, R. Procter, M. Rouncefield, R. Slack, J. Soutter, and A. Voss. 'Repairing' the Machine: A Case Study of the Evaluation of Computer-Aided Detection Tools in Breast Screening. In *Proceedings of ECSCW 2003, 8th European Conference on Computer Supported Collaborative Work*, pages 375–394, Helsinki, Finland, September 2003.

[66] R. A. Hayes and B. J. Feenstra. Video-Speed Electronic Paper based on Electrowetting. *Nature*, 425:383–385, September 2003.

[67] C. Heath and P. Luff. *Technology in Action*. Cambridge University Press, June 2000.

[68] D. L. Hecht. Printed Embedded Data Graphical User Interfaces. *IEEE Computer*, 34(3):47–55, March 2001.

[69] C. Heinzer. iServerP2P: Distributed iServer Architecture Based on Peer-to-Peer Concepts. Diploma thesis, Institute for Information Systems, ETH Zurich, 2004.

[70] L. E. Holmquist, J. Redström, and P. Ljungstrand. Token-Based Access to Digital Information. In *Proceedings of HUC '99,*

1st International Symposium on Handheld and Ubiquitous Computing, pages 234–245, Karlsruhe, Germany, September 1999.

[71] J. Hong, M. N. Price, B. N. Schilit, and G. Golovchinsky. Printertainment: Printing With Interactive Cover Sheets. In *Extended Abstracts of CHI '99, ACM Conference on Human Factors in Computing Systems*, pages 240–241, Pittsburgh, USA, May 1999.

[72] G. T. Huang. Microsoft's Magic Pen. *Technology Review*, 107(4), May 2004.

[73] Dallas Semiconductor, iButton, http://www.ibutton.com.

[74] Seiko Instruments USA Inc, http://www.siibusinessproducts.com.

[75] Ink Markup Language, W3C Working Draft 28 September 2004, http://www.w3.org/TR/InkML/.

[76] H. Ishii and B. Ullmer. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In *Proceedings of CHI '97, ACM Conference on Human Factors in Computing Systems*, pages 234–241, Atlanta, USA, March 1997.

[77] J. Jacobson, B. Comiskey, C. Turner, J. Albert, and P. Tsao. The Last Book. *IBM Systems Journal*, 36(3), 1997.

[78] W. Johnson, H. Jellinek, L. Klotz, Jr., R. Rao, and S. K. Card. Bridging the Paper and Electronic Worlds: The Paper User Interface. In *Proceedings of INTERCHI '93, ACM Conference on Human Factors in Computing Systems*, pages 507–512, Amsterdam, The Netherlands, April 1993.

[79] A. Jøsang, R. Ismail, and C. Boyd. A Survey of Trust and Reputation Systems for Online Service Provision. *Decision Support Systems*, 2005.

[80] C. Kafka. DataGlyphs Bridge Paper & Digital Worlds. *Docu World*, pages 66–67, 1998.

[81] A. Kidd. The Marks are on the Knowledge Worker. In *Proceedings of CHI '94, ACM Conference on Human Factors in Computing Systems*, pages 186–191, Boston, USA, April 1994.

[82] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, J. Schettino, B. Serra, and M. Spasojevic. People, Places, Things: Web Presence for the Real World. In *Proceedings of WMCSA 2000, 3rd IEEE Workshop on Mobile Computing Systems and Applications*, pages 19–28, Monterey, USA, December 2000.

[83] T. Kindberg, E. Tallyn, R. Rajani, and M. Spasojevic. Active Photos. Technical Report HPL-2004-47, HP Laboratories Palo Alto, March 2004.

[84] Dorling Kindersley. *The Dorling Kindersley Nature Encyclopedia*. Dorling Kindersley Limited, London, September 1998.

[85] S. R. Klemmer, J. Graham, G. J. Wolff, and J. A. Landay. Books with Voices: Paper Transcripts as a Tangible Interface to Oral Histories. In *Proceedings of CHI 2003, ACM Conference on Human Factors in Computing Systems*, pages 89–96, Fort Lauderdale, USA, April 2003.

[86] S. R. Klemmer, M. Newman, R. Farrell, R. Meza, and J. A. Landay. A Tangible Evolution: System Architecture and Participatory Design Studies of the Designers' Outpost. Technical Report UCB//CSD-00-1116, Computer Science Division, UC Berkeley, November 2000.

[87] A. Kobler. *The eXtreme Design Approach*. PhD thesis, ETH Zurich, 2001.

[88] A. Kobler and M. C. Norrie. OMS Java: A Persistent Object Management Framework. In *Java and Databases*, pages 46–62. Hermes Penton Science, May 2002.

[89] A. Kobler, M. C. Norrie, B. Signer, and M. Grossniklaus. OMS Java: Providing Information, Storage and Access Abstractions in an Object-Oriented Framework. In *Proceedings of OOIS 2001, 7th International Conference on Object-Oriented Information Systems*, pages 25–35, Calgary, Canada, August 2001.

[90] M. Koch, A. Rancati, A. Grasso, and D. Snowdon. Paper User-Interfaces for Local Community Support. In *Proceedings of HCI International '99, 8th International Conference on Human-Computer Interaction*, pages 417–421, Munich, Germany, August 1999.

[91] N. Kohtake, J. Rekimoto, and Y. Anzai. 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, December 2001.

[92] H. Koike, Y. Sato, and Y. Kobayashi. Integrating Paper and Digital Information on EnhancedDesk: A Method for Realtime Finger Tracking on an Augmented Desk System. *ACM Transactions on Computer-Human Interaction*, 8(4):307–322, December 2001.

[93] H. Koike, Y. Sato, Y. Kobayashi, H. Tobita, and M. Kobayashi. Interactive Textbook and Interactive Venn Diagram: Natural and Intuitive Interfaces on Augmented Desk System. In *Proceedings of CHI 2000, ACM Conference on Human Factors in Computing Systems*, pages 121–128, The Hague, The Netherlands, April 2000.

[94] R. Kurzweil. *The Age of Intelligent Machines*. MIT Press, 1990.

[95] F. Lanfranchi. QuickTime Movie Plug-in for the iServer Architecture. Semester project, Institute for Information Systems, ETH Zurich, 2005.

[96] B. M. Lange, M. A. Jones, and J. L. Meyers. Insight Lab: An Immersive Team Environment Linking Paper, Displays, and Data. In *Proceedings of CHI '98, ACM Conference on Human Factors in Computing Systems*, pages 550–557, Los Angeles, USA, April 1998.

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

[98] D. M. Levy. *Scrolling Forward: Making Sense of Documents in the Digital Age*. Arcade Publishing, October 2001.

[99] J. Lin, M. W. Newman, J. I. Hong, and J. A. Landay. DENIM: An Informal Tool for Early Stage Web Site Design. In *Proceedings of CHI 2001, ACM Conference on Human Factors in Computing Systems*, pages 205–206, Seattle, USA, March 2001.

[100] 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.

[101] P. Luff, C. Heath, and D. Greatbatch. Tasks-in-Interaction: Paper and Screen Based Documentation in Collaborative Activity. In *Proceedings of CSCW '92, ACM Conference on Computer Supported Cooperative Work*, pages 163–170, Toronto, Canada, November 1992.

[102] P. Luff, C. Heath, M. C. Norrie, B. Signer, and P. Herdman. Only Touching the Surface: Creating Affinities Between Digital Content and Paper. In *Proceedings of CSCW 2004, ACM Conference on Computer Supported Cooperative Work*, pages 523–532, Chicago, USA, November 2004.

[103] W. E. Mackay. Is Paper Safer? The Role of Paper Flight Strips in Air Traffic Control. *ACM Transactions on Computer-Human Interaction*, 6(4):311—340, December 1999.

[104] W. E. Mackay, A.-L. Fayard, L. Frobert, and L. Médini. Reinventing the Familiar: Exploring an Augmented Reality Design Space for Air Traffic Control. In *Proceedings of CHI '98, ACM Conference on Human Factors in Computing Systems*, pages 558–565, Los Angeles, USA, April 1998.

[105] W. E. Mackay and D. S. Pagani. Video Mosaic: Laying Out Time in a Physical Space. In *Proceedings of Multimedia '94, 2nd ACM International Conference on Multimedia*, pages 165–172, San Francisco, USA, October 1994.

[106] W. E. Mackay, D. S. Pagani, L. Faber, B. Inwood, P. Launiainen, L. Brenta, and V. Pouzol. Ariel: Augmenting Paper Engineering Drawings. In *Proceedings of CHI '95, ACM Conference on Human Factors in Computing Systems*, pages 421–422, Denver, USA, April 1995.

[107] W. E. Mackay, G. Pothier, C. Letondal, K. Bøegh, and H. E. Sørensen. The Missing Link: Augmenting Biology Laboratory Notebooks. In *Proceedings of UIST 2002, 15th Annual ACM Symposium on User Interface Software and Technology*, pages 41–50, Paris, France, October 2002.

[108] S. Malär. Advanced XHTML Plug-in for iServer. Semester project, Institute for Information Systems, ETH Zurich, 2005.

[109] C. C. Marshall. Annotation: From Paper Books to Digital Library. In *Proceedings of DL '97, 2nd ACM International Conference on Digital Libraries*, pages 131–140, Philadelphia, USA, July 1997.

[110] C. C. Marshall. The Future of Annotation in a Digital (Paper) World. In *Proceedings of the 35th Annual GSLIS Clinic: Successes and Failures of Digital Libraries*, Urbana-Champaign, USA, March 1998.

[111] C. C. Marshall. Toward an Ecology of Hypertext Annotation. In *Proceedings of Hypertext '98, 9th ACM Conference on Hypertext and Hypermedia*, pages 40–49, Pittsburgh, USA, June 1998.

[112] T. Masui and I. Siio. Real-World Graphical User Interfaces. In *Proceedings of HUC 2000, 2nd International Symposium on Handheld and Ubiquitous Computing*, pages 72–84, Bristol, UK, September 2000.

[113] D. R. McGee, P. R. Cohen, R. M. Wesson, and S. Horman. Comparing Paper and Tangible, Multimodal Tools. In *Proceedings of CHI 2002, ACM Conference on Human Factors in Computing Systems*, pages 407–414, Minneapolis, USA, April 2002.

[114] D. R. McGee, P. R. Cohen, and L. Wu. Something from Nothing: Augmenting a Paper-Based Work Practice via Multimodal Interaction. In *Proceedings of DARE 2000, Designing Augmented Reality Environments*, pages 71–80, Elsinore, Denmark, April 2000.

[115] D. H. McKnight and N. L. Chervany. The Meanings of Trust. Technical Report MISRC 96-04, Management Information Systems Research Center, University of Minnesota, 1996.

[116] D. T. Michaelides, D. E. Millard, M. J. Weal, and D. De Roure. Auld Leaky: A Contextual Open Hypermedia Link Server. In *Proceedings of OHS-7, 7th International Workshop on Open Hypermedia Systems*, pages 59–70, Åarhus, Denmark, August 2001.

[117] D. Millard, L. Moreau, H. Davis, and S. Reich. FOHM: A Fundamental Open Hypertext Model for Investigating Interoperability between Hypertext Domains. In *Proceedings of Hypertext 2000, 11th ACM Conference on Hypertext and Hypermedia*, pages 93–102, San Antonio, USA, May 2000.

[118] mimio Xi, Virtual Ink Corp., http://www.mimio.com.

[119] S. Nabeshima, S. Yamamoto, K. Agusa, and T. Taguchi. MemoPen: A New Input Device. In *Proceedings of CHI '95, ACM Conference on Human Factors in Computing Systems*, pages 256–257, Denver, USA, April 1995.

[120] L. Nelson, S. Ichimura, E. R. Pedersen, and L. Adams. Palette: A Paper Interface for Giving Presentations. In *Proceedings of CHI '99, ACM Conference on Human Factors in Computing Systems*, pages 354–361, Pittsburgh, USA, May 1999.

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

[122] W. Newman and P. Wellner. A Desk Supporting Computer-Based Interaction with Paper Documents. In *Proceedings of CHI '92, ACM Conference on Human Factors in Computing Systems*, pages 587–592, Monterey, USA, May 1992.

[123] Nokia Digital Pen, Nokia, http://www.nokia.com.

[124] M. C. Norrie. An Extended Entity-Relationship Approach to Data Management in Object-Oriented Systems. In *Proceedings of ER '93, 12th International Conference on the Entity-Relationship Approach*, pages 390–401, Arlington, USA, December 1993.

[125] M. C. Norrie. Paper on the Move. In *Proceedings of UMICS 2004, 2nd International Workshop on Ubiquitous Mobile Information and Collaboration Systems*, pages 1–12, Riga, Latvia, June 2004.

[126] M. C. Norrie and A. Palinginis. Versions for Context Dependent Information Services. In *Proceedings of COOPIS 2003, 11th International Conference on Cooperative Information Systems*, pages 503–515, Catania, Italy, November 2003.

[127] M. C. Norrie, A. Palinginis, and B. Signer. Content Publishing Framework for Interactive Paper Documents. In *Proceedings of DocEng 2005, ACM Symposium on Document Engineering*, Bristol, UK, November 2005.

[128] M. C. Norrie and B. Signer. Web-Based Integration of Printed and Digital Information. In *Proceedings of DiWeb 2002, 2nd International Workshop on Data Integration over the Web*, pages 71–85, Toronto, Canada, May 2002.

[129] M. C. Norrie and B. Signer. Issues of Information Semantics and Granularity in Cross-Media Publishing. In *Proceedings of CAiSE 2003, 15th International Conference on Advanced Information Systems Engineering*, pages 421–436, Klagenfurt/Velden, Austria, June 2003.

[130] M. C. Norrie and B. Signer. Switching over to Paper: A New Web Channel. In *Proceedings of WISE 2003, 4th International Conference on Web Information Systems Engineering*, pages 209–218, Rome, Italy, December 2003.

[131] M. C. Norrie and B. Signer. Web-Based Integration of Printed and Digital Information. In *Proceedings of the VLDB 2002 Workshop EEXTT and CAiSE 2002 Workshop DiWeb on Efficiency and Effectiveness of XML Tools and Techniques and Data Integration over the Web, Revised Papers*, pages 200–219, 2003.

[132] M. C. Norrie and B. Signer. Information Server for Highly-Connected Cross-Media Publishing. *Information Systems*, pages 526–542, November 2005.

[133] M. C. Norrie and B. Signer. Overlaying Paper Maps with Digital Information Services for Tourists. In *Proceedings of ENTER 2005, 12th International Conference on Information Technology and Travel & Tourism*, pages 23–33, Innsbruck, Austria, January 2005.

[134] K. O'Hara and A. Sellen. A Comparison of Reading Paper and On-Line Documents. In *Proceedings of CHI '97, ACM Conference on Human Factors in Computing Systems*, pages 335–342, Atlanta, USA, March 1997.

[135] The Office of the Future. Business Week, June 1975.

[136] The Open University, http://www.open.ac.uk.

[137] PaperDisk, Cobblestone Software, http://www.paperdisk.com.

[138] Paper++ Project Site, IST-2000-26130, Disappearing Computer Initiative, http://www.paperplusplus.net.

[139] A. Pashtan, R. Blattler, A. Heusser, and P. Scheuermann. CATIS: A Context-Aware Tourist Information System. In *Proceedings of*

IMC 2003, 4th International Workshop of Mobile Computing, Rostock, Germany, June 2003.

[140] A. Pearl. Sun's Link Service: A Protocol for Open Linking. In *Proceedings of Hypertext '89, 2nd ACM Conference on Hypertext and Hypermedia*, pages 137–146, Pittsburgh, USA, November 1989.

[141] E. R. Pedersen, T. Sokoler, and L. Nelson. PaperButtons: Expanding a Tangible User Interface. In *Proceedings of DIS 2000, Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, pages 216–223, New York City, USA, August 2000.

[142] PC Notes Taker, Pegasus Technologies, Ltd., http://www.pc-notetaker.com.

[143] S. Pradhan, C. Brignone, J.-H. Cui, A. McReynolds, and M. T. Smith. Websigns: Hyperlinking Physical Locations to the Web. *IEEE Computer*, 34(8):42–48, August 2001.

[144] M. N. Price, G. Golovchinsky, and B. N. Schilit. Linking by Inking: Trailblazing in a Paper-Like Hypertext. In *Proceedings of Hypertext '98, 9th ACM Conference on Hypertext and Hypermedia*, pages 30–39, Pittsburgh, USA, June 1998.

[145] R. Rao, S. K. Card, W. Johnson, L. Klotz, and R. H. Trigg. Protofoil: Storing and Finding the Information Worker's Paper Documents in an Electronic File Cabinet. In *Proceedings of CHI '94, ACM Conference on Human Factors in Computing Systems*, pages 180–185, Boston, USA, April 1994.

[146] Standard Register, http://www.standardregister.com.

[147] S. Reich, L. Carr, D. De Roure, and W. Hall. Where Have You Been From Here? Trails in Hypertext Systems. *ACM Computing Surveys*, 31(4), December 1999.

[148] J. Rekimoto. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of UIST '97, 10th Annual Symposium on User Interface Software and Technology*, pages 31–39, Banff, Canada, October 1997.

[149] J. Rekimoto. Matrix: A Realtime Object Identification and Registration Method for Augmented Reality. In *Proceedings of*

APCHI '98, 3rd Asia Pacific Conference on Computer Human Interaction, page 63, Hayama-Machi, Japan, July 1998.

[150] J. Rekimoto and Y. Ayatsuka. CyberCode: Designing Augmented Reality Environments with Visual Tags. In *Proceedings of DARE 2000, Designing Augmented Reality Environments*, pages 1–10, Elsinore, Denmark, April 2000.

[151] P. Robinson, D. Sheppard, R. Watts, R. Harding, and S. Lay. A Framework for Interacting with Paper. In *Proceedings of Eurographics '97*, pages 378–385, Budapest, Hungary, September 1997.

[152] P. Robinson, D. Sheppard, R. Watts, R. Harding, and S. Lay. Paper Interfaces to the World-Wide Web. In *Proceedings of WebNet '97, World Conference on the WWW, Internet & Intranet*, Toronto, Canada, November 1997.

[153] M. Rohs and J. Bohn. Entry Points into a Smart Campus Environment Overview of the ETHOC System. In *Proceedings of IWSAWC 2003, 3rd International Workshop on Smart Appliances and Wearable Computing*, pages 260–266, Providence, USA, May 2003.

[154] B. N. Schilit, G. Golovchinsky, and M. N. Price. Beyond Paper: Supporting Active Reading with Free Form Digital Ink Annotations. In *Proceedings of CHI '98, ACM Conference on Human Factors in Computing Systems*, pages 249–256, Los Angeles, USA, April 1998.

[155] S. Schulé. XHTML Plug-in for the iServer Architecture. Diploma thesis, Institute for Information Systems, ETH Zurich, February 2004.

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

[157] B. Signer, A. Erni, and M. C. Norrie. A Personal Assistant for Web Database Caching. In *Proceedings of CAiSE 2000, 12th International Conference on Advanced Information Systems Engineering*, pages 64–78, Stockholm, Sweden, June 2000.

[158] B. Signer, M. Grossniklaus, and M. C. Norrie. Java Framework for Database-Centric Web Engineering. In *Proceedings of WebE 2001, 4th Workshop on Web Engineering*, Hong Kong, May 2001.

[159] B. Signer and M. C. Norrie. Multi-Layered Cross-Media Linking. In *Proceedings of Hypertext 2003, 14th ACM Conference on Hypertext and Hypermedia*, pages 106–107, Nottingham, UK, August 2003.

[160] B. Signer and M. C. Norrie. A Framework for Cross-Media Information Management. In *Proceedings of EuroIMSA 2005, International Conference on Internet and Multimedia Systems and Applications*, pages 318–323, Grindelwald, Switzerland, February 2005.

[161] B. Signer, M. C. Norrie, P. Geissbuehler, and D. Heiniger. Aural Interfaces to Databases based on VoiceXML. In *Proceedings of VDB6, 6th IFIP Workshop on Visual Database Systems*, pages 235–249, Brisbane, Australia, May 2002.

[162] B. Signer, M. C. Norrie, P. Geissbuehler, and D. Heiniger. Telephone Interface for Avalanche Warnings Based on Information Server for Adaptable Content Delivery. In *Proceedings of Pervasive 2002, International Conference on Pervasive Computing*, pages 99–111, Zurich, Switzerland, August 2002.

[163] I. Siio. InfoBinder: A Pointing Device for a Virtual Desktop. In *Proceedings of HCI International '95, 6th International Conference on Human-Computer Interaction*, pages 261–264, Tokyo, Japan, July 1995.

[164] I. Siio, T. Masui, and K. Fukuchi. Real-world Interaction using the FieldMouse. In *Proceedings of UIST '99, 12th Annual ACM Symposium on User Interface Software and Technology*, pages 113–119, Asheville, USA, November 1999.

[165] I. Siio and Y. Mima. IconStickers: Converting Computer Icons into Real Paper Icons. In *Proceedings of HCI International '99, 8th International Conference on Human-Computer Interaction*, pages 271–275, Munich, Germany, August 1999.

[166] J. Smith, T. White, C. Dodge, J. Paradiso, N. Gershenfeld, and D. Allport. Electric Field Sensing for Graphical Interfaces. *IEEE Computer Graphics and Applications*, 18(3):54–60, 1998.

[167] L. J. Stifelman. Augmenting Real-World Objects: A Paper-Based Audio Notebook. In *Proceedings of CHI '96, ACM Conference on Human Factors in Computing Systems*, pages 199–200, Vancouver, Canada, April 1996.

[168] L. J. Stifelman, B. Arons, and C. Schmandt. The Audio Notebook: Paper and Pen Interaction with Structured Speech. In *Proceedings of CHI 2001, ACM Conference on Human Factors in Computing Systems*, pages 182–189, Seattle, USA, March 2001.

[169] Storyspace, http://www.eastgate.com/storyspace/.

[170] Scalable Vector Graphics (SVG) 1.1 Specification, W3C Recommendation, http://www.w3.org/TR/SVG11/.

[171] E. Tallyn, D. Frohlich, N. Linketscher, B. Signer, and G. Adams. Using Paper to Support Collaboration in Educational Activities. In *Proceedings of CSCL 2005, Conference on Computer Supported Collaborative Learning*, Taipei, Taiwan, May 2005.

[172] S. Taylor, C. Dance, W. Newman, A. Taylor, T. Aldhous, and M. Taylor. Augmenting Paper: Using a Video Camera to Support Selective Scanning from Paper to Screen. Technical Report EPC-1998-105, Xerox Research Centre Europe, 1998.

[173] A. A. Terry. Electronic Ink Technologies: Showing the Way to a Brighter Future. *Library Hi Tech*, 19(4):376–389, 2001.

[174] B. Traversat, A. Arora, M. Abdelaziz, M. Duigou, C. Haywood, J.-C. Hugly, E. Pouyoul, and B. Yeager. Project JXTA 2.0 Super-Peer Virtual Network. Technical report, Sun Microsystems, Inc., May 2003.

[175] B. Ullmer and H. Ishii. The metaDESK: Models and Prototypes for Tangible User Interfaces. In *Proceedings of UIST '97, 10th Annual Symposium on User Interface Software and Technology*, pages 223–232, Banff, Canada, October 1997.

[176] B. Ullmer and H. Ishii. mediaBlocks: Tangible Interfaces for Online Media. In *Extended Abstracts of CHI '99, ACM Conference on Human Factors in Computing Systems*, pages 31–32, Pittsburgh, USA, May 1999.

[177] A. Vogelsang and B. Signer. The Lost Cosmonaut: An Interactive Narrative Environment on Basis of Digitally Enhanced Paper. In *Proceedings of VS 2005, 3rd International Conference on Virtual Storytelling*, Strasbourg, France, December 2005.

[178] R. Want, K. P. Fishkin, A. Gujar, and B. L. Harrison. Bridging Physical and Virtual Worlds with Electronic Tags. In *Proceedings of CHI '99, ACM Conference on Human Factors in Computing Systems*, pages 370–377, Pittsburgh, USA, May 1999.

[179] R. Want and D. M. Russel. Ubiquitous Electronic Tagging. *IEEE Distributed Systems Online*, 1(2), September 2000.

[180] K. Weber and A. Poon. Marquee: A Tool for Real-Time Video Logging. In *Proceedings of CHI '94, ACM Conference on Human Factors in Computing Systems*, pages 58–64, Boston, USA, April 1994.

[181] M. Weiser. The Computer for the 21st Century. *Scientific American*, 265(3), September 1991.

[182] P. Wellner. The DigitalDesk Calculator: Tangible Manipulation on a Desk Top Display. In *Proceedings of ACM UIST '91, 4th Annual ACM Symposium on User Interface Software and Technology*, pages 27–33, Hilton Head, USA, November 1991.

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

[184] E. J. Whitehead. An Architectural Model for Application Integration in Open Hypermedia Environments. In *Proceedings of Hypertext '97, 8th ACM Conference on Hypertext and Hypermedia*, pages 1–12, Southampton, UK, April 1997.

[185] S. Whittaker, P. Hyland, and M. Wiley. Filochat: Handwritten Notes Provide Access to Recorded Conversations. In *Proceedings of CHI '94, ACM Conference on Human Factors in Computing Systems*, pages 271–277, Boston, USA, April 1994.

[186] U. K. Wiil and J. J. Leggett. The HyperDisco Approach to Open Hypermedia Systems. In *Proceedings of Hypertext '96, 7th ACM Conference on Hypertext and Hypermedia*, pages 140–148, Washington DC, USA, March 1996.

[187] Wikipedia, The Free Encyclopedia, http://wikipedia.org.

[188] L. D. Wilcox, B. N. Schilit, and N. N. Sawhney. Dynomite: A Dynamically Organized Ink and Audio Notebook. In *Proceedings of CHI '97, ACM Conference on Human Factors in Computing Systems*, pages 186–193, Atlanta, USA, March 1997.

[189] Wiziway SA, http://www.wiziway.com.

[190] A. Würgler. *OMS Development Framework: Rapid Prototyping for Object-Oriented Databases.* PhD thesis, ETH Zurich, 2000.

Index

Curriculum Vitae

Particulars

Name:	Beat Signer
Date of Birth:	August 20, 1973
Birthplace:	Uster, Zurich, Switzerland
Citizenship:	Appenzell, Appenzell Innerrhoden, Switzerland

Education

1980	Primary School in Uster
1981–1985	Primary School in Hornussen
1985–1989	Secondary School in Frick
1989–1993	Matura Type C, Alte Kantonsschule Aarau
1993	Basic Military Education, Swiss Army
1993–1999	Study of Computer Science at the Swiss Federal Institute of Technology, ETH Zurich
1999–2005	Research and teaching assistant supervised by Prof. Moira C. Norrie in the Global Information Systems Research Group, ETH Zurich

Work Experience

1993	PC software and hardware installations and support at teleprint tdc SA
1997	Industry placement, software engineering for real-time systems, Oerlikon Contraves