

Aspects of Modern Electronic Publishing Systems

Dissertation

for the Award of the Academic Degree

Doctor of Technical Sciences

at

Graz University of Technology

submitted by

Harald Krottmaier

Institute for Information Processing and

Computer Supported new Media (IICM)

Graz University of Technology

hkrott@iicm.edu

Date: December 2002

First Reader: o.Univ.-Prof. Dr.phil. Dr.hc.mult. Hermann Maurer

Second Reader: Univ.-Doz. Dipl.-Ing. Dr.techn. Frank Kappe

Aspekte Moderner Electronic Publishing Systeme

Dissertation

zur Verleihung des akademischen Grades

Doktor der Technischen Wissenschaften

an der

Technischen Universität Graz

vorgelegt von

Harald Krottmaier

Institut für Informationsverarbeitung und

Computergestützte neue Medien (IICM)

Technische Universität Graz

hkrott@iicm.edu

Datum: Dezember 2002

Erster Begutachter: o.Univ.-Prof. Dr.phil. Dr.hc.mult. Hermann Maurer

Zweiter Begutachter: Univ.-Doz. Dipl.-Ing. Dr.techn. Frank Kappe

To those who supported me.

Preface

The results of my research work on digital libraries and journals done in the last 4 years is published in this dissertation.

I worked on many aspects of this broad research area. Some parts of the work were implemented in prototypes or modules. The *Journal of Universal Computer Science* (J.UCS), hosted at the Institute of Information Processing and Computer Supported new Media since 1994, served as research and test environment. The implemented prototypes and modules were used in J.UCS and partly in other contexts such as in an intranet environment of a bank.

At the beginning of this dissertation an introduction gives an overview of existing systems as well as the topic of electronic publishing and digital libraries. Thereafter key issues such as the reuse of parts of documents without copying the content (Part “Transclusions”), interactivity in the usage of digital libraries (Part “Working with Digital Journals”), and aspects of the usage and integration of digital libraries (Part “Further Applications of Digital Libraries”) are discussed in detail. Finally, in the last chapter, conclusions and further prospects of digital libraries and electronic publishing systems are discussed.

Some chapters were presented by me or co-authors at conferences in Europe, Asia and America. One chapter was also published in a peer-reviewed journal.

Vorwort

Die vorliegende Dissertation spiegelt meine Arbeit der letzten 4 Jahre im Bereich Digitale Bibliotheken und Journale wider.

Viele Aspekte dieses umfassenden Themengebiets wurden von mir während dieser Zeit untersucht und einige davon auch praktisch in Form von Prototypen bzw. Modulen implementiert. Als Testumgebung und Forschungsobjekt diente primär das *Journal of Universal Computer Science* (J.UCS), das seit 1994 am Institut für Informationssysteme und Computergestützte neue Medien betreut wird. Teile der Ergebnisse wurden auch in anderen Kontexten (z.B. in der Intranet-Umgebung einer Bank) verwendet.

Eine Einführung und ein Überblick über einige bestehende Systeme sollen dem Leser zu Beginn der Arbeit einen Einblick in *Electronic Publishing Systeme* geben. Danach werden drei Schwerpunkte meiner Arbeit, nämlich Arbeiten zu Dokumentformaten und Wiederverwendung von Teilen eines Dokuments (Part “Transclusions”), Interaktivität in digitalen Bibliotheken und Journalen (Part “Working with Digital Journals”) und die Verwendung und Integration unterschiedlicher Systeme (Part “Further Applications of Digital Libraries”) erläutert. Ein Kaptitel mit einer Zusammenfassung der Ergebnisse und einem Ausblick auf absehbare, zukünftige Entwicklungen und Verbesserungen schließt die Arbeit ab.

Die vorliegenden Arbeiten wurden vom mir oder von Co-Autoren auf Konferenzen in Europa, Asien und Amerika präsentiert und mit Fachleuten diskutiert. Eine Arbeit wurde in einem wissenschaftlichen Journal publiziert.

I hereby certify that the work reported in this thesis is my own and that work performed by others is appropriately cited.

Signature of the author:

Ich versichere hiermit wahrheitsgemäß, die Arbeit bis auf die dem Aufgabengesteller bereits bekannte Hilfe selbständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten Anderer unverändert oder mit Abänderungen entnommen wurde.

Acknowledgments

Without the help of many people this dissertation would not have been possible. If I could set all names to the first position, I certainly would do so, since everyone is an essential part of this mosaic. Nevertheless, I had to create a sequence and here it is:

First, I'd like to thank Prof. Hermann Maurer, who is not just the supervisor of this dissertation, but who has also helped me and inspired me over the last 4 years in many aspects. He spent uncountable hours supporting and encouraging me. He enabled thoughts I could not think of before! His widespread mind still surprises me after all these years.

I appreciate the work done by the second reader, Frank Kappe. Beside his job as Chief Technology Officer at Hyperwave Research and Development Inc. he found time for reading and improving the dissertation.

I also thank Klaus Schmaranz, who strongly influenced me. He enjoyed sharing his knowledge with me and was a constant source of deep inspiration. It was always a pleasure to work, talk and listen to him. Klaus is a co-author of two contributions.

Other co-authors of some papers (in alphabetical order): Christof Dallermassl (who is a great juggler by the way), Wolfgang Götzinger, Christian Gütl, Heimo Haub, Denis Helic, Philipp Zambelli and Bernhard Zwantschko: it was nice to talk and work with you and I hope that our cooperation will continue.

Dana Kaiser spent a lot of hours correcting the papers I had written. Her suggestions made the papers more comprehensible and easier to read. Christian Hinteregger also helped me to express my thoughts correctly.

The whole team at the IICM supported me either directly or indirectly! Marie-Luise Lampl organized a lot of conference travels. Michaela Gmeindl, Peter Grundner and Dana Kaiser kept the J.UCS server running. Karl Trummer, Helmut Leitner and his team helped me to set up the server software and hardware for J.UCS. Didi Freismuth and Karl Blümlinger have joined me very often for lunch and coffee breaks. They have helped me to recharge my batteries during these short breaks. Roswitha Hammer supported me in the library and helped me to find the right books. Thanks to all of you and all members of the IICM!

Last, but not least, I want to thank my family and all my friends for being there, whenever I needed them! Andrea and Michael, I love you more than anything else in this world! My parents Liane and Siegfried Krottmaier supported and believed in me since I started to think (probably even before!). Thanks to my sister Michaela, her family and my (prospective) parents and brother-in-law, Christine, Josef and Christian Hinteregger, for their support.

Harald Krottmaier, December 2002.

Contents

Preface	i
Acknowledgments	vii
I Overview and Scope of this Dissertation	1
1 Introduction	3
1.1 History and Notations	3
1.2 Challenges in Digital Libraries	9
1.2.1 Contents Component	11
1.2.2 Management Component	15
1.2.3 Usage Component	18
1.3 Structure of the Dissertation	20
2 Current and Future Features	27
2.1 Abstract	28
2.2 Introduction and Overview	28
2.3 Overall System Structure and Features	30
2.3.1 Browsing	31
2.3.2 Searching	34
2.3.3 Refinding and Organizing Content	37

2.3.4	Alert Service	39
2.4	Content Related Features	40
2.4.1	Format of an Article	40
2.4.2	Features Using Content	42
2.5	Conclusion and Future Work	44
II	Transclusions	47
3	Transclusions in the 21st Century	49
3.1	Abstract	50
3.2	Introduction	51
3.2.1	Intellectual Property	52
3.2.2	Disk Space	53
3.2.3	Update	54
3.2.4	Two Way Reading	55
3.3	Document Formats	55
3.3.1	HyperText Markup Language	56
3.3.2	Extensible Markup Language	58
3.4	Server Systems	60
3.5	Implementing Transclusions	62
3.6	Applications for Transclusions	65
3.7	Conclusion and Future Work	66
4	Further Aspects of Transclusions	71
4.1	Creation of Transclusions	72
4.2	Visualization of Transclusions	75
4.3	Managing Changes: Conditional Text	77
4.4	Concrete Applications of Transclusions	80

<i>CONTENTS</i>	xi
4.4.1 Scientific Electronic Publishing	80
4.4.2 Discussion Forum	81
4.4.3 Course Material	81
4.5 Conclusion	82
III Working with Digital Journals	87
5 Improving Usability	89
5.1 Abstract	90
5.2 Introduction	90
5.3 Requirements for a Workspace	92
5.4 More than just Reading...	95
5.5 Conclusion and Future Work	100
6 Interactive Features	103
6.1 Abstract	104
6.2 Introduction	104
6.3 Annotations	105
6.4 Rearrange Documents and Document Fragments	109
6.5 Conclusion and Future Work	112
7 Automatic References	115
7.1 Abstract	116
7.2 Introduction, Problems and a Possible Solution	116
7.3 Conclusion and Future Work	118
8 Enhanced Annotations	121
8.1 Abstract	121
8.2 Introduction	122

8.3	Motivation	123
8.4	Improvements	124
8.5	Conclusions and Future Work	126
9	Support in the Review Process	129
9.1	Abstract	130
9.2	Introduction	130
9.3	Current Situation and Improvements	131
9.4	Improvements in the Paper Submission Process	132
9.5	Search for the Right Referee	135
9.5.1	Currently Available Information	135
9.5.2	Search for Publications of Referees	137
9.5.3	Similarity Search	139
9.6	Further Applications	140
9.7	Conclusion and Future Work	142
IV	Further Applications of Digital Libraries	147
10	Managing Digital Audio Data	149
10.1	Abstract	151
10.2	Introduction	152
10.3	Internet and Digital Audio Formats	154
10.4	Classification and the Need of Meta Information	156
10.5	Multimedia and Hyperwave Information Server	157
10.6	Implementation and Features	159
10.7	Some Notes on Data-Reuse	163
10.8	Current and Future Works	165

<i>CONTENTS</i>	xiii
-----------------	------

11 Adaptive Learning Environment	171
---	------------

11.1 Abstract	174
11.2 Motivation	174
11.3 System Requirements	175
11.4 Concept	178
11.5 Views	180
11.6 Conclusion	184

12 Distributed Teaching Environments	187
---	------------

12.1 Abstract	188
12.2 Motivation	188
12.3 A Distributed Environment	192
12.4 Data and Course Flexibility	193
12.5 An Extendable and Exchangeable Environment	195
12.6 A Highly Customizable Environment	196
12.7 User Access Management and Security	198
12.8 Conclusion	199

V Summary and Outlook	203
------------------------------	------------

13 Conclusions	205
-----------------------	------------

13.1 Summary, Results and Conclusions	205
13.2 The Future of Digital Libraries	210
13.2.1 Future Creation of Digital Content	210
13.2.2 Future Dissemination of Information	215
13.2.3 Future Usage of Digital Libraries	218

14 References Overview	223
14.1 Contributions of the Author	223
14.2 Further Readings in Digital Libraries	225

Part I

Overview and Scope of this Dissertation

Chapter 1

Introduction

In this chapter a brief overview of electronic publishing and digital libraries is given. Thereafter current research interests and challenges related to digital libraries are enumerated. An overview of the structure of this dissertation is given in the last section of this chapter.

1.1 History and Notations

Before the discussion about electronic publishing and digital libraries is started, a brief review of historical ideas related to the topic is given.

One of the first *non-traditional* library systems mentioned in history was *memex*. In 1945 Vannevar Bush published a paper where the system is described ([Bush, 1945]). “*As We May Think*” is one of the most popular contributions to information retrieval, therefore it is impossible to write about information processing and digital library systems without citing ideas from Bush’s famous paper. A few key ideas are now cited and reviewed. Bush documented the situation in 1945 in detail and described the term “records”:

A record if it is to be useful to science, must be continuously extended, it must be stored, and above all it must be consulted. Today we make the record conventionally by writing and photography, followed by printing; but we also record on film, on wax disks, and on magnetic wires.

This quote reflects on the *non-static* requirements of records. They have to be adapted and extended by the users of the system to stay up-to-date. In the following discussion the term “information entity” is used instead of “record” because many database-experts use the term “record” for one simple entry in a database. Obviously, more than one database-entry is necessary to express a constantly growing piece of information.

Bush anticipated very small (“in the size of a walnut”) and effective cameras which are able to take full-color pictures – even stereoscopic ones. Hundreds of photographs may be taken in this way. The pictures are sufficiently small so that the entire *Britannica* may be stored on a single film about the size of a sheet of paper. Costs of duplication are very low. Therefore scientists may have a personal copy of encyclopedias and can access them. In addition to pictures and typed text, voice may be recorded by the system. Access to content stored in the system is described as follows ([Bush, 1945]):

Consider a future device for individual use, which is a sort of mechanized private file and library. It needs a name, and, to coin one at random, “memex” will do. [...] There is, of course, provision for consultation of the record by the usual scheme of indexing. If the user wishes to consult a certain book, he taps its code [...] and the title page of the book promptly appears before him [...]. Frequently-used codes are mnemonic, so that he seldom consults his code book [...]. On deflecting one of these levers [...] each page

in turn being projected at a speed which just allows a recognizing glance at each. If he deflects it further to the right, he steps through the book 10 pages at a time; still further at 100 pages at a time.

Every *information entity* must be stored *once* in the system. An electronic book or article may be accessed via indices in much faster speed than in traditional environments. Nowadays the term *code book* is commonly replaced by the term “personalized index”. Bush described the system as a desk with a keyboard, and sets of buttons and levers. A projection display for microfilms is designed to be the output device.

Bush pictured not only tools for information retrieval but also tools for *interactive knowledge creation*. Marginal notes and other types of comments are described. Even interactive features still not widely available in current web technology implementations, such as building trails and interlinking different types of content, were already mentioned and discussed in 1945.

The idea of *memex* is based on analog technologies. In the late 1960’s some ideas about information processing based on digital technologies were described. J.C.R. Licklider and T. Nelson are the most famous scholars who worked in the early days of digital information processing.

Licklider studied how digital computing could transform libraries. In 1965 he published a book titled “Libraries of the Future” where he foresaw many developments. Unfortunately, the book is hard to find and less well known than the mentioned article published by Bush. “Libraries of the Future” is not available on the Internet ([Arms, 2000]).

The second famous scholar is T. Nelson. He coined the term *Hypertext* ([Nelson, 1965]):

Let me introduce the word “hypertext” to mean a body of written or pictorial material interconnected in such a complex way that it could not conveniently be presented or represented on paper. It may contain summaries, or maps of its contents and their interrelations; it may contain annotations, additions and footnotes from scholars who have examined it. Let me suggest that such an object and system, properly designed and administered, could have great potential for education, increasing the student’s range of choices, his sense of freedom, his motivation, and his intellectual grasp.

Many years later (in 1990) the web was invented at CERN by Tim Berners-Lee. Robert Cailliau and his team wrote the first WWW-client called “Mosaic”. Unfortunately, the developers simplified the hypertext-model described in [Nelson, 1965]. Therefore it is not possible for users to annotate documents stored on a web-server or implement other interactive features without extending the standard implementations. Ideas of the original hypertext model such as bidirectional links, link management, integrated metadata management, information clusters etc. were implemented in the Hyper-G system (see e.g. [Maurer, 1996]). Since it is necessary to use the mentioned features in a highly sophisticated digital library environment it was decided to use Hyper-G technology instead of simple standard web-server technology for hosting a digital journal. The electronic journal (Journal of Universal Computer Science, J.UCS) is described in detail in chapter 2.

There is no common term for the items stored in a digital library. [Arms, 2000] suggests *material* or *item*, but many other authors use *electronic document*, *entity* or *object*. As already mentioned above the term *information entity* is used for an item stored in the digital library.

In the following some definitions relating to the term “digital library” are given. *Electronic* and *virtual library* are often used synonymously. People working with digital libraries come from many different disciplines and therefore different terminologies are used for the same meaning. As the following quote shows, it is not easy to define the term *digital library*. In the early 1990’s many initiatives worked on the development of digital libraries ([Fox, 1993]):

One group was supposed to define the library. It came back with a statement that a digital library is a *distributed technology environment* which dramatically reduces barriers to the *creation, dissemination, manipulation, storage, integration and reuse of information* by individuals and groups. They suggested the national initiative should contain some specific testbed projects, but gave no guidance on what these should be. In other words, they not only failed to define the collection, they didn’t really even describe the system that would hold it.

A few years later the Association of Research Library summarized the definitions of a digital library as follows ([Association of Research Library, 1995]):

- A digital library is not a single entity
- A digital library requires technology to link the resources of many digital libraries
- Linkages between the many digital libraries and information services are transparent to the end users
- Universal access to digital libraries and information services is a goal

- Digital library collections are not limited to document surrogates: they extend to digital artifacts that cannot be represented or distributed in printed formats

The Association for Computer Machinery classifies *Digital Libraries* (H.3.7) as a specialization of *Information Storage and Retrieval* (H.3) in *Information Systems* (H) ([ACM-CCS, 1998]).

In [Endres and Fellner, 2000] a digital library is defined as “electronic and virtual library with system-intelligence”. An electronic library is an institution where the content (images, text-, sound-, video-documents etc.) is stored in electronic form. “Virtual” in this context reflects the independence of user’s location when providing services. Only if the services provided by different institutions are integrated *and* accessible via a uniform interface the term “digital library” should be used.

Digital libraries are one topic in this dissertation. The other main topic is related to electronic publishing. According to [ATIS, 2000], *electronic publishing* is “the process of creating messages, distributing them, and reproducing them entirely online, often with a capability for feedback. Note: Unlike desktop publishing, electronic publishing does not usually generate hard copy.”

[Schmaranz, 1998] also discusses briefly the term *electronic publishing*. Most people see electronic publishing as a kind of “electronic paperware”, i.e. content of the documents is mostly text-based and non-interactive. Advantages over traditional publishing arises because of the electronic nature of the documents: searching and hyperlinking are just two of them. Obviously, electronic publishing is more than electronic paperware. Any kind of an electronic document may be published and modern hypermedia systems add additional navigation capabilities. Schmaranz called this approach *dynamic interactive hypermedia publishing*.

In this dissertation the term *electronic publishing* is used as a term for “the act of publishing any type of electronic document on an electronic information system”. *Interactive* electronic publishing adds interactive features to the published documents whereas the software used for interaction with the document is either provided at the server- or the client-side.

In the next section current research issues and challenges in the field of digital libraries are discussed.

1.2 Challenges in Digital Libraries

A study ([Lyman and Varian, 2000]) produced by faculty and students at the School of Information Management and Systems at the University of California at Berkeley shows the current information overload. The team attempted to measure how much information is produced in the world each year.

The world produces between 1 and 2 exabytes of unique information per year, which is roughly 250 megabytes for every man, woman, and child on earth. An exabyte is a billion gigabytes, or 10^{18} bytes.

More than 90% of this enormous annual output is stored digitally. Only 0.003% of this output is stored in printed form. Little of this information is made available through Digital Library collections ([DELOS, 2001]).

In 2001 *DELOS Network of Excellence*, an initiative founded by the European Commission’s Information Society Technologies 5th Framework Program (IST-FP5), organized a workshop with leaders in the field of digital library research. In a brainstorming session future directions of the European research program were discussed ([DELOS, 2001]).

The participants of this meeting agreed on the following vision statement:

Digital Libraries should enable any citizen to access all human knowledge anytime and anywhere, in a friendly, multi-modal, efficient, and effective way, by overcoming barriers of distance, language, and culture and by using multiple Internet-connected devices.

From a user's point of view this single sentence is an evidence to the broadness of the field of digital libraries. It also summarizes different aspects of digital library systems. Challenges of digital library system are divided into three system components ([DELOS, 2001]). These components are:

Contents Component: The content of a digital library is stored in this component. Contents may be stored in different electronic formats and metadata may be attached to them. This makes it easier for the users, authors and editors of the digital library to work with the contents.

Management Component: Metadata and content are managed in this layer of the model. It also includes the implementation of functionality provided for users, authors and editors.

Usage Component: This component deals with all aspects of the interface between user and system. Note that functionality may be provided via different interfaces.

In the next sections each of these components is discussed in detail.

1.2.1 Contents Component

Preservation of content is one large research area in the contents component. Different document formats and publishing systems are used in digital libraries to represent contents. This content must be available to future generations. However, there are many different document formats available even in text based systems. It is very important to choose the *right document format*. The electronic format must:

- be accessible and usable even in many years
- allow high quality publications
- be platform independent

Since there is no recommendation which document formats will be supported in the future, it was decided to support different document formats for single information entities in the research environment. In the Journal of Universal Computer Science three different document formats of articles (information entities) are stored to satisfy different user's needs and achieve greatest flexibility: PostScript (PS, [Adobe, 1989]), Portable Document Format (PDF, [Adobe, 1993]) and documents formatted in Hypertext Markup Language (HTML, [Raggett et al., 1999]).

In 2002 the Association for Information and Image Management (AIIM) announced to examine PDF for long term storage and preservation. PDF allows high quality publications by storing font information and layout. Viewers for different platforms are available and the specification of the format is freely available to everybody. Problems related to usability of this format regarding readability on screen are discussed in section 2.4.1. Unfortunately, PDF is not

very user-friendly: it does not support automatic adaption of column sizes, is fixed in layout and inadequate in an adaptive environment.

Users who prefer reading of documents on paper will choose PDF or PostScript formatted content where layout is the main issue. If the user prefers reading on screen, it is very likely that HTML will be the first choice. Unfortunately, it is impossible to create high quality publications using HTML as document format. Mathematical formulae and high quality graphics are still difficult to create using this kind of markup language. Extensions of HTML like MathML allow to express high quality mathematical formulas but up-to-date browsers must be used to view such formulas. Other extensions of HTML like Stylesheets can be used to express highly sophisticated document layout in HTML. Nevertheless, not every browser supports Stylesheets in the same way and therefore the use of different browsers often results in different renderings of a document.

In the last years XML (Extensible Markup Language, [W3C, 2000]) has become more and more popular. Many designers of documents (usually professionals) are using XML because of many advantages over other document formats. Compared to SGML (Standard Generalized Markup Language), XML is much easier to understand, extend and use. It is possible to use XML as a “single source publishing” format. An XML document can be transformed into different electronic document formats. Therefore XML is an ideal solution for the publishing industry.

Unfortunately, it is unusual in the scientific community to format scholarly papers in XML. Scholars want to use their preferred word processor (Word, StarWriter, \LaTeX etc.) to express ideas. A conversion from arbitrary document formats is possible. However, with a small editorial team it is impossible to transform submitted articles into an XML-confirming style, even though au-

automatic conversion tools are available for some document formats. Only large digital libraries (such as [ACM Digital Library, 2002]) have the manpower to do this additional editorial work.

Recently established document formats (e.g. [eBook, 2002]) add more functionality to the document format and even take distribution of documents into account. They combine high quality publications with adaptability regarding user-preferences and device dependent specifications (e.g. screen size).

Digital libraries offer many features not available in traditional library systems: it is possible to publish even non-printable documents. Videos, 3D-scenes, interactive contents etc. may be published in digital libraries. Multimedia libraries offer more features than traditional text based electronic libraries. Content representing video-, audio-, geological-data etc. must be stored in the system. However, content is worthless without proper software on the consumer's side. Therefore such systems must provide appropriate tools to view and work with the content.

To guarantee accessibility to electronic content in the future, the content as well as all tools used to provide *and* consume it must be archived. This step might be difficult because of the complexity of currently available systems. Note that even the system-based hardware must be archived in some way. If archiving is not possible for any reason, the content and publishing system must be migrated to a new environment. Obviously, no features must be lost by this migration. Archiving and preservation is a broad issue in the field of digital libraries and document management systems.

Building information collections is a difficult task and tools to make this process easier must be developed. Information acquisition has to be as easy as possible and an automatic acquisition of primary content should be possible. Especially in multimedia information collections this acquisition process is very

expensive. Once data is stored in an unedited manner, information about this data must be extracted out of the available content to enrich it. Consider the following example of an image: when someone creates and stores the image in any format, just “technical data” about this image is available. This data includes color depth, size of the image, resolution etc. Nevertheless, this information is almost useless for further (intelligent) processing of the image. Information like “this image is a construction plan of house X” is *useful* additional information about the image. Therefore several tools must be developed making it possible for the creator of the image to add this kind of information. Tools which automatically extract information like “this picture shows *Mr. X* talking to *Mrs. Y*” must be created and used. It is essential to create a corresponding relation between metadata and contents. Most important, the metadata must be stored in an appropriate format. Over the last decades many different formats and interpretations of metadata were developed, including Marc, Dublin Core and MPEG-7 ([MPEG-7, 2002]). At the time of writing MPEG-7 is state of the art in metadata storage.

An automatic extraction of metadata would be very helpful for the creators of information collections. Automatic link creation, summarization and classification will (usually) enhance the value of information stored in these collections. If data is edited by an editorial board, value would definitely be added to the raw data.

Information means different things to different people, especially people from different cultures and different countries. Therefore an information collection should be created with all these varieties in mind. However, there are still many problems with translating tools (e.g. when homographs are used in the text).

This discussion shows the variety of aspects in the contents component. Preservation of contents, collection building, metadata extraction and tools to support these tasks are not widely available.

1.2.2 Management Component

While the previously discussed component is more related to the creation, access and storage of contents, the management component is responsible for the management of a system, including scalability, access management, interoperability, quality and administration.

A digital library system must be scalable to be flexible in terms of users and stored content. Therefore a decentralized system architecture must be provided by the used system. Particular effort should be put to the investigation of peer-to-peer architectures. Digital libraries should be implemented as “24 x 7”-service. Dynamic reconfiguration, high availability, replication etc. are keywords in this area.

In our research environment we use a Hyperwave Information Server ([Hyperwave, 2001]) where many management features are already available. User- and group management is available without any further implementation work as well as highly sophisticated document management features such as fulltext and metadata search, similarity search, link management, automatic topic clustering etc.

Interoperability in the exchange and reuse of content *and* metadata is a main issue in the management of digital libraries. It must be possible to transform documents on the fly to other formats depending on factors like used device, connection speed, but also depending on the preferences of the users. Interoperability between different digital libraries is a very interesting topic. Two

chapters discuss a middleware system used for seamless integration of different systems (see chapter 11 and 12). Using this kind of technology will make libraries without walls possible.

The Open Archive Initiative ([OAI, 2002]) introduces an architecture for the exchange of metadata. A simple client server architecture with service- and data providers exchanges metadata. Simplicity is a key issue of this framework. It allows every publisher to implement necessary protocols quickly and to participate in the framework. “Data providers” (i.e. publishers) are harvested by service providers. Service providers are either publishers as well or arbitrary distributors. Simple extensions to the hypertext transfer protocol (HTTP, [Fielding et al., 1999]) defined in “The Open Archives Initiative Protocol for Metadata Harvesting” (see [OAI, 2002]) must be implemented by each data provider. The metadata is basically formatted in XML using elements similar to “Dublin Core”-elements.

Security is another issue in digital libraries. Privacy, anonymity and authorization are three typical aspects of security. When top-secret documents are stored in the system, or when users must pay for information, security issues must be taken into account. Other keywords such as integrity and confidentiality should just be mentioned in this context and are key issues in current research.

While the contents component is concerned with the technical side of the contents, the management component’s competence is to secure the quality of contents as well. In scholarly publication quality is very often guaranteed by a peer-review process. A rigorous review process by experts on the related topic guarantees steady quality of publications. Several other quality control mechanisms are described in [Arms, 2002], including editorial control, reputation, volunteer reviews, and automatic methods like citation indices etc.

Chapter 9 describes another method which can increase the quality of a scientific peer-reviewed journal. By selecting the most appropriate referee, the referees' workload is reduced and quality of the publication is ensured. Nevertheless, metrics to express quality must be developed and established. It must be immediately obvious for the reader which metric is used and why content is considered as "good" or "bad".

Many payment-methods for electronic resources were introduced in the last 10 years. Questions like "How to borrow a book from a digital library?" still cannot be answered generally. When looking at the eBook initiative, very funny situations arise: Adobe for example limits the usage of electronic books by adding different "rights-attributes". An electronic version of "Alice in Wonderland" prohibited "reading aloud". Most children cannot read when they are interested in the story. Therefore there is – legally – no chance for them to get the contents ([Sperberg, 2001]).

Several kinds of statistics are necessary to improve quality of service. Access-statistics about a certain paper may reflect the quality of this specific paper. However, there is no established standard for logging activities in digital library systems. In chapter 13 several activities regarding evaluation of log-files to increase usability and quality of a digital library are discussed.

The management component is responsible for content and user management, billing, accounting, and exchange of contents and metadata. A digital library must be available seven days a week as a "24x7"-service. Distributed quality control ensures necessary quality of contents and full user satisfaction.

1.2.3 Usage Component

The highest level of a digital library system is visible to the user: the interface. The main issue in this component is to ensure optimal user experience in digital library interactions so that the user may obtain the desired information in the best and fastest way.

A user interface (UI) must therefore consider latest findings in usability research. Access to information stored in digital libraries must be task-oriented. Since there are many different tasks, different interfaces must be provided for users.

Personalization of UIs as well as adaptive UIs must be provided for users. Different users act differently in many aspects. The following aspects must be considered when developing personalization concepts:

- functions or features offered by systems
- structure of the systems and
- contents stored in the systems

These aspects must be adaptable to personal models of the user ([Tochtermann, 2002]). The field of personalization is very broad. Personalization in the context of digital libraries is covered in great detail in [Tochtermann, 2002]. Tochtermann considered digital libraries *and* knowledge-management systems at the same time. Accordingly to his ideas of personalization concepts it is possible to personalize content stored in different systems as well as their features simultaneously. An essential point in this discussion is the possibility to *hide* features *and* content to make usage of information (or “knowledge-objects”) stored in the systems as efficient as possible.

Explicit and implicit profiling of users must be implemented by the systems. A famous example of an adaptive UI is implemented by Amazon. Additional information about user behaviors is displayed. The semantic of log-file evaluation is added and displayed to the user (like “readers of this page also read the following...” or “people who bought this book also bought the following...” etc.). Personalization also includes annotation facilities. The concept of annotations and an implementation is explained in section 6.3.

Usage of annotations may be improved by adding additional meta-information to an annotation object. In our research environment we have implemented several types of annotations, such as question, answer, hint etc. This information is implemented via attributes to link-objects and increases efficiency of the readers when working with the content. Intensive use of these attributes in combination with highly sophisticated knowledge management tools makes it possible to implement the concept of active documents (see e.g. [Heinrich and Maurer, 2000]).

Personalization aspects can be expanded from a single user level to a group level. Information sharing within a community is a powerful technique and must be available within the UI of a digital library.

Features of a digital library as well as contents must be reusable. On the one hand functions like “find related articles” (available in the management component) must be provided via forms or hyperlinks for the user in the usage component. Functions are reused in different contexts. Content and parts of content on the other hand should also be reused in different contexts without being copied. Issues concerning the usage component are discussed in several upcoming chapters.

1.3 Structure of the Dissertation

As the previous sections have shown, the field of digital libraries is broad. It is obviously impossible to explore every challenge listed in the previous section in a single dissertation. Therefore a selection of them was chosen to be explored in detail. These results were presented and discussed at different international conferences. Most of the following chapters were published in conference proceedings and journals. The contributions may be clustered into five parts:

The first part, including this and the next chapter, is an introduction to available electronic publishing systems. Before starting research on publishing systems, an exploration of already existing systems is essential. For this purpose a few popular systems were selected, analyzed and compared to our research environment (J.UCS). The selected systems do not just differ in size, content *and* concurrent users, but also in the *concept* of published content. Nevertheless, they all provide the user with primarily text based content. This common issue allows in-depth comparison of these systems. Problems of systems were detected and possible solutions to these problems were proposed. An insight in the features and the used software platform of the Journal of Universal Computer Science is given and future developments are briefly mentioned.

The second part of the dissertation deals with transclusions. In several different digital journals it is impossible to *reuse* content without copying it. The idea of transclusions is about reusing content and content fragments stored on a system. The document representing the content may be stored in any format on any server. Questions concerning performance, source and destination format, server systems etc. are answered in this part. There are many problems but there are also technical solutions to these problems. We propose a solution and introduce a “transclusion aware publishing system” based on the

Hyperwave Information Server. A general chapter shows problems and implementation of transclusions (see chapter 3). Thereafter issues of transclusions and possible applications are discussed. It is shown that transclusions are a solution to many problems. Transclusions will additionally save time for the reader when reading and exploring content.

The third part of this dissertation is a collection of papers related to interactive features and improvements in the usability of digital library systems. Many systems do support users with a simple and static navigation paradigm. However, all users are different. Users differ in educational background, preferences and many other aspects. It is a badly designed electronic publishing system which provides just one single navigation for all users. First generation publishing systems were not able to *adapt* to different users. Publishing systems have to provide users with information stored on the system, at the same time as they have to allow users to work actively with the systems. Users work together and may share ideas via the publishing system. In chapter 5 a server-side workspace is introduced. Several features, such as annotating material and working with document fragments are described. With a large digital library and many users very sophisticated features are possible, including the implementation of active documents, automatic intelligent summary, automatic references etc. Users, both readers and creators of content, profit from the introduced features. Applications of features available in modern knowledge-management systems allow a much faster review-process in peer-reviewed journals.

Usage of digital libraries as well as aspects and problems of integration of digital libraries into other systems are discussed in the fourth part of the dissertation. An adaptive learning environment is introduced and the need for adaptive systems is explained in detail. The link between different distributed systems is a middle-ware software (Dinopolis) developed at the IICM. Adding

new and adapting existing features is easily possible using such a middle-ware software layer.

The dissertation closes with conclusions and an outlook to the future of electronic publishing systems. Further developments and issues related to improvements of service to all users of the Journal of Universal Computer Science are mentioned.

References

- [ACM Digital Library, 2002] ACM Digital Library (2002). <http://www.acm.org/dl>.
- [ACM-CCS, 1998] ACM (1998). ACM Computing Classification System.
- [Adobe, 1989] Adobe (1989). PostScript Programmieretechniken. Addison-Wesley Longman, Inc.
- [Adobe, 1993] Adobe (1993). Portable Document Format Reference Manual. Addison-Wesley Longman, Inc.
- [Arms, 2000] Arms, W. (2000). *Digital Libraries*. The MIT Press.
- [Arms, 2002] Arms, W. (2002). What Are the Alternatives to Peer Review? Quality Control in Scholarly Publishing on the Web. *The Journal of Electronic Publishing*, Vol. 8, Issue 1
- [Association of Research Library, 1995] Definition and Purposes of a Digital Library Association of Research Libraries, 1995 <http://sunsite.berkeley.edu/ARL/definition.html>
- [ATIS, 2000] Alliance for Telecommunications Industry Solutions. Telecom Glossary 2000
- [Bootstrap, 2002] Engelbart, D., Engelbart Ch., Rush P., Dye M., and Yim P. <http://www.bootstrap.org>
- [Bush, 1945] Bush, V. (1945). As We May Think. *The Atlantic Monthly*, 176(1):101–108. <http://www.theatlantic.com/unbound/flashbks/computer/bushf.htm>.

- [DELOS, 2001] DELOS Brainstorming Report. Digital Libraries: Future Directions for a European Research Programme. San Cassiano, Alta Badia, Italy. 13-15 June 2001.
- [eBook, 2002] Introduction to eBook. <http://www.ebooks.at>
- [Endres and Fellner, 2000] Endres, A. and Fellner, D. (2000). *Digitale Bibliotheken*. dpunkt.verlag GmbH, Heidelberg.
- [Fielding et al., 1999] Fielding, R., Gettys, J., and Mogul, J. (1999). Hypertext Transfer Protocol – HTTP/1.1.
- [Fox, 1993] Fox, E. (Ed.). Source Book on Digital Libraries, Version 1.0 December 6, 1993 Covering a series of NSF Invitational Workshops and Related Information <ftp://fox.cs.vt.edu/pub/DigitalLibrary/DLSB.pdf>
- [Heinrich and Maurer, 2000] Heinrich, E. and Maurer, H. (2000). Active Documents: Concept, Implementation and Applications. *Journal of Universal Computer Science*, 6(12):1197–1202. http://www.jucs.org/jucs_6_12/active_documents_concept_implementation
- [Hyperwave, 2001] Hyperwave Information Server. <http://www.hyperwave.com>.
- [Lyman and Varian, 2000] Lyman P. and Varian H. R. (2000) How Much Information <http://www.sims.berkeley.edu/how-much-info>
- [Maurer, 1996] Maurer, H., Editor (1996). Hyper-G now Hyperwave — The Next Generation Web Solution. Addison-Wesley

- [MPEG-7, 2002] Martinez, J. (Ed.). MPEG-7 Overview ISO/IEC JTC1/SC29/WG11 <http://mpeg.telecomitalia.com/standards/mpeg-7/mpeg-7.htm>
- [Nelson, 1965] Nelson T. Complex information processing: a file structure for the complex, the changing and the indeterminate in Proceedings of the twentieth national conference, 1965, p84–100, Cleveland, Ohio, United States
- [OAI, 2002] Open Archive Initiative (OAI) <http://www.openarchives.org/>
- [Raggett et al., 1999] Raggett, D., Hors, A. L., and Jacobs, I. HTML 4.01 Specification, W3C Recommendation. <http://www.w3.org/TR/html4>.
- [Schmaranz, 1998] Schmaranz, K. Aspects of Electronic Publishing in Hypermedia Systems. Dissertation at Graz, University of Technology. June 1998.
- [Sperberg, 2001] Sperberg, R. Removing Those Pesky Passwords <http://www.ebookweb.org/opinion/roger.sperberg.20010712.aebpr.htm>
- [Tochtermann, 2002] Tochtermann, K. Personalisierung im Kontext von Digitalen Bibliotheken und Wissensmanagement Habilitationsschrift, Graz, University of Technology. March 2002.
- [W3C, 2000] Extensible Markup Language (XML) 1.0 (Second Edition). <http://www.w3.org/TR/2000/REC-xml-20001006>.

Chapter 2

Current and Future Features of Digital Journals

This chapter was presented at “The First EurAsian Conference on Advances in Information and Communication Technology” (EurAsia 2002) in Shiraz, Iran.

The paper published in the proceedings shows the current state of the Journal of Universal Computer Science (J.UCS) and gives an overview of features available in some other very popular digital library systems. Available features are divided into two categories to reflect the structure of the databases used and implementation of the user-interfaces. It also shows that a few features available in the Journal of Universal Computer Science are still not available in most of the systems, like multi-categorized information entities, sophisticated fulltext-search, typed annotations, etc. An outlook to some features currently under development and still not available in *any* of the other systems, such as expert search, is also given in the corresponding sections.

The original contribution was published in the conference proceedings and can be found in [Krottmaier, 2002c].

2.1 Abstract

Features in currently available systems are often restricted to passive “consumption” of articles stored in the corresponding digital journal. This paper classifies features into two categories (overall system structure, and content based features) and gives an outlook of planned implementations in the Journal of Universal Computer Science (J.UCS). It also shows that using the powerful Hyperwave Information Server (HIS) makes it quite easy to *implement* features and make knowledge management features (such as “find an expert on a topic”) available to users of a digital journal.

2.2 Introduction and Overview

First generation publishing systems were very static and inflexible. Simple pages encoded in some digital format were prepared by some editorial team, stored on a server system, and then transferred to the user on request. Systems were based on ordinary web-servers without any interactive features. Just “get document XXX” was very often used the only available request. With ongoing development of technologies user supporting tools like “search through the content stored on a server” etc. were implemented. This paper is about features available in digital journals. The following systems were explored in detail:

ACM-DL: Digital library of the ACM (Association for Computing Machinery) [ACM Digital Library, 2002].

LINK: Information service published by Springer [LINK, 2002].

Xplore: Service published by IEEE (Institute of Electrical and Electronics Engineer) [IEEE-xplore, 2002].

ScienceDirect (SD): published by Elsevier [Science Direct, 2002]

JoDI: The Journal of Digital Information maintained by the IAM Research Group, University of Southampton [JODI, 2002].

J.UCS: The Journal of Universal Computer Science. A publication of the Know-Center in cooperation with Springer Co.Pub., JOANNEUM RESEARCH and the IICM, Graz University of Technology [J.UCS, 2002].

To simplify the following discussion, we use abbreviations (printed in *italic*) to refer to the systems, i.e. we write *Xplore* when we talk about the digital library system of the IEEE. *ACM-DL*, *LINK*, *Xplore* and *ScienceDirect* are serving several journals, conference proceedings and books. *JoDI* and *J.UCS* are serving *one* journal each. The selected libraries are very diverse in terms of amount of data provided for the user, features offered, licensing etc. Nevertheless, the selected systems are all electronic publishing systems trying to support the user in information and knowledge gathering, therefore the demands on the features are very similar.

In the following sections we take a critical look at selected qualities of systems, categorized in two categories:

Overall System Structure and Features: like exploring the content stored in the systems via searching and browsing as well as finding articles again in the system.

Content Related Features: like document formats and interactive and active features using the content or parts of the content.

At the end of each subsection we take a look at the differences between the selected features and features available in *J.UCS*. We also present some ideas of future improvements related to the corresponding topic.

2.3 Overall System Structure and Features

Organization and structure of *information entities* are a main issue in information systems. The digital world consists of servers, index- and content pages. Disadvantages of traditional libraries (like limited space in shelves, borrowed books not available to other users etc.) are easily “repaired” in the digital equivalent. But there are problems to solve: we all are familiar with the *lost in hyperspace syndrome* (e.g. [Theng and Thimbleby, 1998]) and try to avoid it in the design of an information system’s user interface.

The content of almost every investigated digital libraries is available not only in electronic form but also in printed form. Articles also appear on paper but the electronic edition appears usually earlier (e.g. *LINK*s “Online First”-service). Since every page includes a page number, an additional navigation feature is added to the content.

In a printed environment space is always an issue and articles must not exceed some predefined number of pages. Electronic editions are easier to handle in this respect: They offer (nearly) unlimited storage space, therefore there is (usually) no limit to the size of an information entity. It is easy for an electronic system to add additional material to an entity, e.g. information about the author (link to the homepage of the author, curriculum vitae, email address etc.), additional software, tutorials etc.

While all these features are “nice to have”, they also have massive drawbacks: just consider archiving the material. Problems related to the electronic format of the content and reliability of storage mediums arise.

Let us take a look at navigation-features of a digital publishing system. Finding the right content starting from the entry page of the system may be accomplished in two ways: by browsing (see section 2.3.1) or by searching for a title, author’s name etc. (see section 2.3.2). If the right article is explored, functions like “Remember this article” (see section 2.3.3) and “Find related articles to the current one” are very useful (see section 2.4.2).

2.3.1 Browsing

Large collections of articles are organized in issues of a journal, issues are collected in volumes, and volumes are again collected into some category. Since entities of publishing systems are accessible via different index pages, all thinkable kinds of sorting (by author, title, category,...) are possible. Figure 2.1 shows a very common implementation of browsing through the information stored on the system. The volume- and issue-listing differ in the level of detail displayed to the user.

In *ACM-DL* it is possible to sort the content by the type of publication (journals, magazines, proceedings etc.), after following e.g. the *journal*-link, a list of all journals published in the *ACM-DL* is presented to the user. All other investigated digital libraries offer similar browsing mechanisms to the user. *LINK* and *Xplore* offer listings by title of the journal, *ScienceDirect* also generates listings clustered by the subject of the journal (e.g. Computer Science, Chemistry etc.).

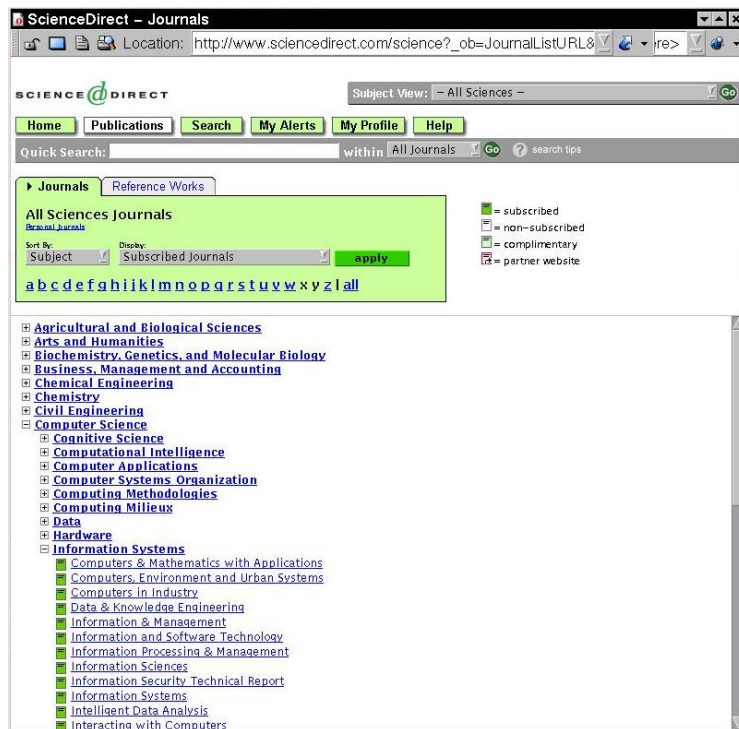


Figure 2.1: List of Journals in Science Direct

Since *JoDI* and *J.UCS* are serving one journal, there is no need for an index at that level. However, articles are organized in issues, volumes and in ACM-CCS to make it easier for users to navigate to them. Figure 2.2 shows an article with appropriate links back to the issue and volume where it was published. Issues published in the same volume, and other volumes are directly reachable.

ScienceDirect is the only examined system where several publishers offer content. Table 2.1 shows the types of indexpages provided by the different systems.

Very often articles are also categorized by some classification system (e.g. the ACM computing classification system, ACM-CCS, [ACM-Classification,1998] for computer science related material). This classification of articles makes it possible to browse through categories of similar documents (see also section 2.4.2). Thereby it is easier to gain an overview of available documents

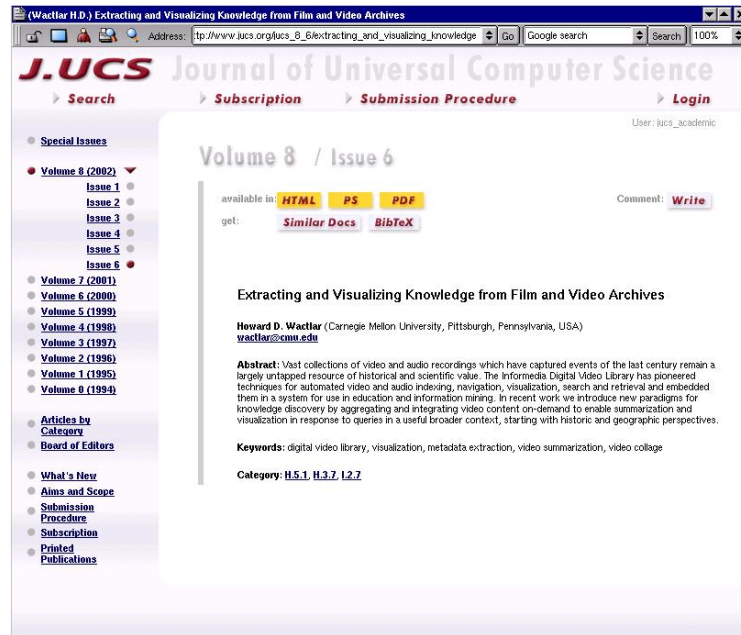


Figure 2.2: Browsing in J.UCS

about a specific topic. Nevertheless, if the amount of articles published in one category increases, a “flat” listing is inappropriate. Additional features like filtering of selections and searching through collections are required. All of the investigated server systems have the browsing feature implemented by generating HTML documents.

In *J.UCS* articles are categorized via the ACM-CCS and may be linked to several categories. At the moment all categories are equally important so there is no *rating* of a category. We think of an additional classification of these categories like “most related category” and “related category”. In the future much more sophisticated (graphically visualized) and easier to use graphical browsing techniques will be implemented too because the number of articles published in *J.UCS* increases.

Indexpage	<i>ACM-DL</i>	<i>LINK</i>	<i>Xplore</i>	<i>SD</i>	<i>JoDI</i>	<i>J.UCS</i>
Journallisting	x	x	x	x		
by type of pub.	x	x	x	x		
by title	x	x	x	x		
by publisher				x		
by subject		x		x		
Volumelisting	x	x	x	x	x	x
incl. issues	x	x	x	x	x	x
incl. articles					x	x
Issuelisting	x	x	x	x	x	x
Articlelisting	x	x	x	x	x	x

Table 2.1: Browsing the Digital Library: Overview of Indexpages

2.3.2 Searching

An expert uses a digital library different from an ordinary user. It is very likely that an expert prefers the search-feature for exploring the library over browsing. Searching is ideal if someone knows exactly what to look for. Publishing systems usually offer search features for bibliographic data and/or for fulltext data. More sophisticated systems also allow scoping of the search query. Fulltext search is an obvious issue in systems where text-information is stored.

Bibliographic data (also know as “metadata”, like author, title of the item, publishing date etc.) is typically stored in some database or in a separate file. There are many different formats for storing bibliographic data. Dublin Core (DC, see e.g. [Dublin Core Metadata Initiative, 2001]) is one of the most popular standards for metadata-entries. DC classifies metadata (called “Core Elements”) into three categories:

Content: title, subject and keywords, description, source, type

Intellectual Properties: author/creator, publisher, rights management and contributor

Instantiation of the Document: date, format, identifier and language

It is also possible to qualify elements. To give an example: the “Date”-element (sub-element of the instantiation core-element) may be qualified by *created*, *issued*, *modified* etc. The concept of elements and qualifiers makes DC very powerful. Elements are clustered into the following categories:

Many systems provide the metadata of the content in that format. If one needs metadata in some other format (e.g. BibTeX), it is a straightforward task to convert the data. Metadata in *J.UCS* are stored in an HIS-proprietary format, however, they may be transformed into arbitrary different formats. At the time of writing conversion of metadata into the well-known BibTeX format is supported by the system.

A highly sophisticated search interface for searching in bibliographic data (as an example see figure 2.3), like searching in abstract, author, affiliation of the author, keywords, publication name, and title is provided by *LINK*. It is notable that different interfaces are provided to the user depending on user’s skills. For novice users a simple form is available. Experts will use the high sophisticated and very powerful search interface. Scoping of the search area by date of publication, category and language is possible. The other reviewed systems also offer some similar technique: a simple search form for novice users and another form for experts.

Aside from the search in metadata, fulltext search is available in most systems. Several different search engines are used and the input forms presented to the user are very different. Depending on the search engine implementation,

LINK Bibliographic Search

Search for

within Abstract And

within Abstract And

within Abstract And

case-sensitive CrossSearch in PubMed/Medline

add line start search clear form

Command Line Search

Your search terms are copied to and displayed in the Command Line text box. If you use different Boolean operators to connect the fields please use brackets in this box to define the hierarchy.

Published

Since Month Year Before Month Year

Classification (PACS)

(e.g. 79.20.Ds; 68.10.Jy; 68.35.Rh)

Libraries

All libraries Chemical Sciences Computer Sciences

Economics Engineering Environmental Sciences

Geosciences Law Life Sciences

Mathematics Medicine Physics and Astronomy

Languages

English German French Italian

start search clear form

Figure 2.3: Bibliographic Search in Springer-Link

operations like “these words should appear in the same sentence” (e.g. implemented in *J.UCS*) are possible. Some operations also use thesaurus search, i.e. if someone searches for the term “magnetic tapes” also other “computer storage devices” may be found.

ACM-DL, *ScienceDirect* and *J.UCS* offer a combined search form for meta-data- and fulltext search, whereas *LINK* offers different forms for each kind of search. At the time of writing *Xplore* did *not* support fulltext search – but this feature will be included in the future.

If some entries are found, they are presented to the user. The listing of matching articles may be sorted by users preferences like date of publication, author, relevance etc. In some systems evaluated it is possible to store search queries and work with the results of those queries.

Search interfaces are either powerful but difficult to use, or weak but easy to use. Upcoming features includes integrated support of thesauri, scoping, searching in several server systems and merging of related documents, as well as ranking of results by different criteria including rating of the article.

In *J.UCS* we are going to improve the search-interface on the one hand and on the other hand we are going to implement different interfaces for experts and new users. Some special queries (like all publications from a specific research institute) are under consideration but therefore a few more metadata must be added to the objects.

2.3.3 Refinding and Organizing Content

Once an article is explored, it is very likely that a user wants to read it again for reference purpose without storing the article on the local computer system. Stability related to storage location (e.g. via Uniform Resource Name, URN, [Moats, 1997]) and content of the information entity is therefore an issue. Some journals (like *ACM-DL* and *LINK*) provide Digital Object Identifiers (DOI, [DOI, 2002]) which are a kind of URN.

In an electronic system electronic bookmarks are the right solution for most of the users to the problem of refinding articles ([Krottmaier, 2001]). They can be created either on the client-, the server side, or on some intermediate or proxy system. The most important feature of server side bookmarking is the possibility to share bookmarks with other colleagues. Therefore one user may

collect and share a selection of articles about a specific topic. Other users may use these articles via the given bookmarks and rate or comment on the articles via the bookmarks. A proper user- and group management on the server side is a prerequisite for bookmarks on the server side.

Beside static bookmarks, dynamic bookmarks should be implemented, i.e. bookmark listings are generated on the fly by performing some database search query. Therefore the list is always up to date. Users may be informed by some alert email mechanism if new articles arrive in the listing. So called “active” bookmark listings may be arbitrary scheduled by the user.

At the time of writing *ACM-DL* was the only examined system which supports server side bookmarking (called *binders*). Binders are private by default but it is possible to share the bookmarks with other users or groups of users. It is also possible to create *active binders*, which performs some *saved search query*. Notifications via email, if the listing changes, are optional.

LINK and *JoDI* do not offer any structuring feature for users. Users of *ScienceDirect* have the option to collect journals in some *Personal Journal* region on the server. Unfortunately, it is not possible to share this collection with other users.

In *J.UCS* a personal restructuring feature is under development. It is planned that each registered user will have the possibility to create arbitrary collections of articles, reuse existing collections (like articles structured by category), or store some active *search query*. These collections may be shared with other registered users. To provide flexible sort orders of the articles (like sorting by date, title, author of the article etc.) surrogates for the original articles will be created in the database. These surrogates will store context specific attributes of the original article. With surrogate objects it will also be possible to represent articles not stored in the database and well defined parts of articles.

The screenshot shows a web browser window titled "Bookshelf" with a URL containing "al&d=ACM&row=0&CFID=1686905&CFTOKEN=97049428". The page header includes the ACM logo and "PORTAL THE ACM DIGITAL LIBRARY". The main content area is titled "Bookshelf - Create a New Binder" and features a form with the following sections:

- Binder Definition:** Includes a "Name" field (pre-filled with "hkrott@iicm.edu"), an "Update" section with radio buttons for "Manually" (selected) and "Automatically", and a "Saved Search" section with a text input and a "Test Search" button.
- Notification:** A section with radio buttons for "None" and "Notify me:".
- Sharing:** A section with radio buttons for "Not Shared" and "Share as:".
- Users:** A section with a text input and a "Create Binder" button.
- Groups:** A section with a "Choose a Group ..." dropdown menu and a "Create Binder" button.

To the right of the form, a "Working with the Binder Definition" section provides instructions and tips:

- You may define a New Binder, or update an Existing Binder.
- Fill in all the fields and click Create Binder or Update Binder appropriately.
- You may also delete a Binder by clicking Delete Binder.
- Name:** The name of this Binder as it will appear on your Bookshelf.
- Update:** Manually updated Binders are maintained by you, as you come across citations of interest you may save them in this Binder. Automatically updated Binders are associated with a Saved Search (see below).
- Saved Search:** Enter the desired search criteria, the collection of citations that result from this search will be stored in this Binder.
- Notification:** You can be notified when new content has been added to this Binder. This can occur when a saved search matches any new materials, and when you are sharing another Subscribers Binder.
- Sharing:** You may share a Binder with an Individual, (you must know their Username) or with an entire Group.
- After making any changes you must click the appropriate button at the bottom of the form.**

The footer of the page states: "The ACM Portal is published by the Association for Computing Machinery, Copyright © 2001 ACM, Inc."

Figure 2.4: ACM-DL: GUI for the creation of a personal binder

2.3.4 Alert Service

As noted in the previous section, alert services are a powerful and *active* system feature. Many journals use different alert mechanisms to notify registered users. A user may be interested in all new arrived articles, articles matching some query in the fulltext or title, or by new articles written by some author.

Unfortunately, this feature is different to use on different publishing systems. Some systems (e.g. *LINK* and *JoDI*) do not support the user with a pre-filtering of the alert email, therefore an email is send to the user even if the user is not interested in the specific topic of the articles published in some journal.

Hermes, a system described in [Faensen et al., 2001], is a solution to this problem. This service filters notifications of published articles before they are sent to the user. A userprofile – like email address, search queries for title and/or abstract – must be defined by the user once for many publishing systems. If there is a match, a notification email will be scheduled.

2.4 Content Related Features

The features described in the previous section are system wide features supporting the user by searching and exploring the *structure* of a publishing system. In this section we explore features which are available as soon as the right *content* is displayed on the screen.

2.4.1 Format of an Article

The content of an article is very often available in some “printer friendly” format like the Portable Document Format (PDF, [Adobe, 1993]) or PostScript (PS). Although PDF is a very popular document format and viewers are available for different operating systems, there are enormous problems with PDF:

Interacting with PDF documents requires additional software:

Interaction with a PDF document is limited to the functionality implemented in the major viewer: Adobe Acrobat Reader. Taking a closer look at the implemented features of that software uncovers the big problems when trying to work *actively* with PDF: It is not possible to change any content of the PDF document. There are some navigation and zooming features available (like goto next/previous/first/last page and zoom-in/out etc.), as well as searching, but no *interaction* with the document

is possible. It is not possible in Acrobat Reader (up to version 5) to add bookmarks or annotate/highlight parts of a document. Other implementations of viewers (e.g. Xpdf, [Noonburg, 2002]) are also very limited in functionality. Buying the full version of Adobes PDF-aware software where it is possible to interact with the document as described above is not acceptable for most of the users because of the price. Nevertheless, PDF was not designed to be heavily modified after creation.

PDF documents are not suitable for reading on screen: A study by Nielsen (see [Nielsen, 2001]) showed that PDF is not the preferred document format for reading content online.

“Forcing users to browse PDF files makes usability approximately 300% worse compared to HTML pages. Only use PDF for documents that users are likely to print. [...] PDF is great for distributing documents that need to be printed. But that is all it is good for. No matter how tempting it might be, you should never use PDF for content that you expect users to read online.”

Unfortunately, many digital journals provide the content of documents exclusively in PDF. This forces the majority of users to print out the articles and read these articles offline. When reading the article offline it is not possible to use *interactive features* of the digital journal *while* exploring the content.

These two problems are solvable: In the first place if someone does not have access to full PDF-aware software (i.e. to Adobe Acrobat) and wants to interact with the document, the document *itself* must provide additional hyperlinks for *interactive web-applications*.

Existing PDF documents may be modified on the server side by some PDF manipulation software. This application may add hyperlinks to interactive web applications. Unfortunately, no information about the structure of a PDF document (like sections, headings etc.) is available. Granularity of the added hyperlinks is therefore limited to pages. To give an example: The PDF manipulation software may take a single PDF document requested by the user, add some header with a special link to each page on the fly and deliver then the modified PDF document to the user. If the user wants to annotate the paper, it is possible to add annotations to a specific page or the whole document. Bookmarks can also be evaluated to enhance the granularity of annotations. Please note that this feature of adding interactivity to *existing* PDF documents stored in *J.UCS* is currently under development and therefore there are no usability studies and user acceptance tests available at the time of writing.

If content is available in XML (Extensible Markup Language) and the PDF document is generated on the fly, the document structure may be extracted out of the XML document.

Secondly it is possible to provide content in different document formats (e.g. in *J.UCS* content is currently provided in HTML, PostScript and PDF). The user may select the appropriate document format depending on the device used. If the content is stored in some generic format (like XML) transformation to arbitrary document formats is possible using the Extensible Stylesheet Language (XSL).

2.4.2 Features Using Content

The content of an article is obviously used in fulltext search. Most of the available digital journals index the content into a fulltext index regardless of the document format. The usage of a thesaurus and additional information of the

words (e.g. position in the document) allows high sophisticated search queries like “search for words within a sentence or paragraph” (see section 2.3.2). Finding similar articles is a feature often available in KM systems. In *J.UCS* and *ACM-DL* this feature is available and may be applied to articles stored on the respective system.

Disassembling the content into the different parts of an article (like abstract, sections, conclusion, bibliography etc.) is a necessary task to make additional features work. Using the abstract (or all abstracts of published articles) makes it possible to support the user with an overview of the available material. The user may then decide which article to read and which article not to read before downloading.

Sections in the document may be interlinked automatically (intra-document links) and it is very common (e.g. in *ACM-DL* and *LINK*) to link even bibliography entries to the electronic versions of the cited material (extra-document links). Section headings may be used to give an additional article navigation frame to the user. Single words may be interlinked to a dictionary or thesaurus to support the user with additional information. Especially in systems supporting teaching it is very common to enable links to dictionaries.

In digital libraries it is also possible to put additional material of the content on the publishing server, including multimedia attachments, demonstration packages, glossaries, programs, additional references etc. Although this is very interesting for online-publications, it is not possible to archive some of these additional contents on paper.

An active document concept is introduced in [Heinrich and Maurer, 2000]. With this concept it is possible to ask questions at any time and any position in the document. The system will then either search for an appropriate answer in the system or will forward the question to the author of the document.

In *J.UCS* we are going to work much more with parts of documents, especially with the reference section. We want to extract information out of this very special section to support the reader with real important information like “Which other resources should be read to understand this article?” and “Who are the *real* experts on this topic?”.

2.5 Conclusion and Future Work

Features of digital publishing systems may be separated in overall features and content related features. This paper has described some currently available features of different systems. It has been shown that (inter-)active features already available for most knowledge management systems are not widely available in digital journals.

Future work is addressed to bring some of those KM features into the journal environment of *J.UCS*. An interactive typed-annotation feature is currently available for HTML documents only. We are going to add this feature also to PDF formatted documents.

References

- [ACM-Classification,1998] ACM (1998). ACM Computing Classification System <http://www.acm.org/class>
- [ACM Digital Library, 2002] ACM (2002). ACM Digital Library <http://www.acm.org/dl>.
- [Adobe, 1993] Adobe (1993). *Portable Document Format Reference Manual*. Addison-Wesley Longman, Inc.
- [DOI, 2002] DOI (2002). Digital Object Identifier. <http://www.doi.org>.
- [Dublin Core Metadata Initiative, 2001] Dublin Core Metadata Initiative (2001). Dublin Core Metadata Initiative. <http://www.dublincore.org>.
- [Faensen et al., 2001] Faensen, Faulstich, Schweppe, Hinze, and Steidinger (2001). Hermes - A Notification Service for Digital Libraries. *Proceedings of First ACM/IEEE-CS Joint Conference on Digital Libraries*, Roanoke Virginia, USA.
- [Heinrich and Maurer, 2000] Heinrich, E. and Maurer, H. (2000). Active Documents: Concept, Implementation and Applications. *Journal of Universal Computer Science*, 6(12):1197–1202. http://www.jucs.org/jucs_6_12/active_documents_concept_implementation.
- [IEEE-xplore, 2002] IEEE (2002). Digital Library of the IEEE. <http://ieeexplore.ieee.org>.
- [JODI, 2002] JODI (2002). Journal of Digital Information. <http://jodi.ecs.soton.ac.uk>.

- [J.UCS, 2002] J.UCS (2002). Journal of Universal Computer Science. <http://www.jucs.org>.
- [Krottmaier, 2001] Krottmaier, H. (2001). Improving the usability of a digital library. In Hübler, A., Linde, P., and Smith, J. W., editors, *Electronic Publishing*, pages 178–182, Canterbury, Kent, United Kingdom.
- [Krottmaier, 2002c] Krottmaier, H. (2002c). Current and Future Features of Digital Journals. In Shafazand, H. and Tjoa, A. M., editors, *Proceedings of “The First Eurasian Conference on Advances in Information and Communication Technology” (EurAsia-ICT 2002)*, LNCS 2510. Springer Verlag.
- [LINK, 2002] Springer (2002). Link Service. <http://link.springer.de>.
- [Moats, 1997] Moats, R. (1997). *URN Syntax*. RFC 2141. <ftp://ftp.isi.edu/in-notes/rfc2141.txt> .
- [Nielsen, 2001] Nielsen, J. (2001). Avoid PDF for On-Screen Reading. <http://www.useit.com/alertbox/20010610.html>.
- [Noonburg, 2002] Noonburg, D. (2002). Xpdf. <http://www.foolabs.com/xpdf/>.
- [Science Direct, 2002] Elsevier (2002). Science Direct. <http://www.sciencedirect.com>.
- [Theng and Thimbleby, 1998] Theng, Y. and Thimbleby, H. (1998). Addressing Design and Usability Issues in Hypertext and on the World Wide Web by Re-Examining the “Lost in Hyperspace” Problem. *Journal of Universal Computer Science*, 4(11):839–855.

Part II

Transclusions

“So Xanadu basically has been my name for an evolving but essentially centrally the same system for the supply and presentation of material with two basic relationships: what we would call the link, which is an unchanging connection between objects, or parts which are different, and the transclusion, which is a maintained connection between parts which are the same.” *Ted Nelson*

Chapter 3

Transclusions in the 21st Century

The first results of the explorations about transclusions were published in the Journal of Universal Computer Science in late 2001. Thereafter two further articles related to transclusions were presented at different conferences. Important advantages and basic concepts of transclusions are summarized in this chapter, further aspects are discussed in the next chapter.

Transclusions are a central aspect in electronic publishing. They would make life much easier for publishers, authors, and readers of electronically published articles. Interestingly transclusions are still not available, neither to the Internet community nor to word-processor users.

Main issues of transclusions (such as update and two way reading) are discussed in this chapter. Advantages over ordinary electronic publishing paradigms when reusing fragments of documents are discussed. Ideas on how to implement transclusions using HTML- and XML-formatted documents are presented. This paper shows that bidirectional links are an absolute requirement

for the implementation of transclusions. Therefore, a Hyperwave Information Server – the platform of our running research and development environment – is a good starting point for the implementation of the prototype.

The original paper can be found in [Krottmaier and Maurer, 2001].

3.1 Abstract

When quoting some part of a document authors usually cut-and-paste the relevant content into the new document. Thereby the connection between this selected part and the original document is lost. Transclusions – first mentioned in 1960 by Ted Nelson – address this problem of “lost context”. With transclusions it is possible to store information about the original document and the exact position of the quote in the newly created document and provide the reader with additional navigational features. Document formats and information systems matured over the last 40 years. This paper gives an overview of some document formats available today in the WWW environment and points to some requirements for server systems providing transclusions. Thereafter we present some ideas on how to implement transclusions based on a Hyperwave Information Server (HIS).

3.2 Introduction

More than 40 years ago, Ted Nelson dreamed about a universal information system called Xanadu. One key feature of that system was the idea of transclusions ([Nelson, 1995]):

The central idea has always been what I now call transclusion, or reuse with original contexts available, through embedded shared instances (rather than duplicate bytes).

The following aspects must be considered when talking about transclusions: the original piece of text to be used as quote, the newly created document using a quote, the system where the documents are stored and the environment of the reader. In the late 1960's, when defacto no information systems were available, Nelson based his ideas on one large and homogeneous system. As we all know there are now many different systems available. This makes it much more difficult to provide a solution for transclusions.

Nowadays HTML (HyperText Markup Language) is the most common format in distributing text documents on the web. Hence we discuss it in section 3.3. Usually not a whole document is quoted, but parts of it. The selected part must therefore be defined in some way.

When using some data to be “included” in a document, this data must be accessible to the author via some protocol. That means in the context of today's available IT infrastructure that the reused data must be stored on some server system on the Internet. The systems at issue are mostly accessible via HTTP (HyperText Transfer Protocol, [Berners-Lee et al., 1996], [Fielding et al., 1999]) and have a restricted command set. See section 3.4 for details about this aspect.

To enable reuse with the original context available, both of the involved server-systems or the client program must provide navigational features to the reader. It must be visible to the reader which part of the document is written by the author and which part is quoted. The original context of the quote is also important to the reader, especially in scientific publications.

Using a special information server (Hyperwave Information Server, HIS) with sophisticated features as publishing server system, makes it easier to solve some of the problems. We introduce a concept on how to implement simple data inclusions with documents stored on an HIS in section 3.5.

Before we start with the more technical parts, let us summarize the need of inclusions of text data and point out some problems.

3.2.1 Intellectual Property

As noted above, using some text via cut-and-paste removes not just the context of the quote, but also other information (meta data) of it like the original author, date of publication etc. With systems supporting the transclusion idea this problem would simply disappear. Every access to the quote would be redirected to the original server. Therefore even charging for pieces of data could be implemented. Nelson dreamed about automatically paid royalties to the owner of each document. However, we are miles away from a solution like this. Micropayment (the payment of very small amounts of money called “microcents”) is still under development. The W3C E-Commerce/Micropayment Working Group ([W3C, 2001b]) specified a common markup for Micropayment but currently this specification is just available to members of the working group. Most of the content providers finance their activities via advertisements (like banners etc.). These advertisements can be easily removed by “web washer” programs (i.e. programs which block specific URLs or content, matching some

pattern). Implementations of transclusions may extract just the interesting part of some document. Hence the quoted content can be displayed without the advertisement. This is not what an author wants, therefore we propose that the original author must permit the reuse of the whole document or parts of it.

3.2.2 Disk Space

If the same material is used in numerous documents the necessary space to store the material is reduced when using transclusions. In 1960 storage-space was an issue because of the high costs. We are now in the year 2001 and it is well known that disk space of several gigabytes is not a problem neither in terms of costs nor in terms of availability. Hence one might believe that this issue is not a problem any more. This is not true. Think of a very large multi media document where you want to delete or add a single character. If someone wants to preserve previous versions of the document separate files of the document have to be stored. After several changes in the document, disk space is again an issue. Therefore transclusions or version control systems (like cvs, [CVS-HOME, 2001]) should be used. As this paper is not about version control systems, we just include a short discourse on how version control with transclusions may work in section 3.6.

The availability of large local disk space brings a new aspect of transclusions. Locally stored data should also be available for transclusions! If someone e.g. bought and installed the new Brockhaus multimedia DVD ([BMM3, 2000], one of the largest available multimedia collections), why is it not possible for an author to refer to a special part of this – locally installed – multimedia collection? There must be a possibility to address parts of documents stored on the local file-system in an arbitrary format. As another example imagine

a database query. Why cannot a system integrate the *query* rather than the *result* of the query?

3.2.3 Update

The publishing system designed by Nelson (Xanadu) was based on a “write once / read many”-storage system. Therefore a once published document could not be changed and stayed stable forever. Nowadays information systems are more dynamic. This results in several problems especially for authors using quotes. For published documents which are changeable by the author (“privately published”), the advantage of updating documents automatically arise ([Nelson, 1987]):

No copying operations are required among the documents throughout the system and thus we solve the problems of updating documents. We solve this problem simply by windowing to a changing document. Thus the problem of distributed update, . . . , disappears.

In some circumstances automatic update of parts of a document is not what an author wants! Therefore some attributes concerning the update behaviour of the quote window must be introduced to prevent automatic updating. For a detailed discussion about different kinds of quote windows and how a system should handle quotes see section 3.4.

Replication mechanisms for certain systems are available, but these mechanisms mostly work in one homogeneous environment. The web consists of very inhomogeneous systems. It is difficult to keep the copies up to date. Some systems on the web, e.g. news services, implement distributed updates

of material using a common protocol. Nevertheless, it takes time, bandwidth and also disk space to synchronize content.

3.2.4 Two Way Reading

Having the possibility to read a quote in the original context is “added value” to the reader and of interest to the author of the original part of information. I.e. the question: “Who uses (parts of) my article?” can easily be answered if the part is included via transclusions. The answer to that question is invaluable! Bidirectional links are the obvious requirement to enable this feature. Although WWW links have now been around for over ten years, only few systems implement the paradigm of two way links.

As the list above shows there are enormous advantages of using transclusions rather than simply cut-and-paste parts of a document. It is obvious that some additional tools are required when preparing documents. In this paper we introduce some ideas on how to implement the idea of transclusions using a Hyperwave Information Server. A transclusion is specified by the author with some tool and is then converted to some special object in the database. Thereafter the system takes control of the compound document and provides author and reader with additional navigational features to the original document of the quote. Let us now take a look at different document formats on the web and compare some of the features relevant to the transclusion concept.

3.3 Document Formats

Since we propose to use transclusions in an internet environment, we take a closer look at the document formats available there. Documents are usually

described using HyperText Markup Language (HTML, e.g. [W3C, 2001a]) or the Extensible Markup Language (XML, [W3C, 2000]).

3.3.1 HyperText Markup Language

Documents are usually sequences of characters. In HTML special sequences of characters (“tags”) indicate some markup. This is an indication for the rendering program to display parts of the material in a special manner to the user. All content displayed to the user – except images, inline frames and objects – is stored inside the HTML file. Hyperlinks are also stored inside the file and are unidirectional. This makes it very difficult to manage the integrity of links and include sophisticated navigational features. Without the ability to include any type of content stored outside the HTML file in the rendered file and bidirectional links, HTML is not suitable for transclusions. Let us take a look at a very simple tag in HTML: the tag to include images in the document (IMG-tag).

Addressing documents in the world wide web is done via uniform resource identifiers (URI, [Berners-Lee et al., 1998]) and uniform resource locators (URL, [Berners-Lee et al., 1994]). To include an image which is accessible via some URI in HTML formatted documents, simply add the following tag:

```
<IMG SRC="URI" [WIDTH="X"] [HEIGHT="Y"]>
```

The browser program now “knows” to include some image addressed by the specified URI in the document. The user reading that document on the screen does not know the exact storage location of the image (without reading the source code of the document)! The physical location of the image does not have to be the same as the physical location of the document. The render-

ing routines of browsers do not distinguish the rendering of local and remote images.

Inline images are – as links (A-tag) in HTML documents – unidirectional, i.e. the author of the image does not know in which documents the image is used. Therefore changing the content of the image without changing the URI can be misleading. See below for a discussion about links in HTML and see section 3.4 for requirements of a system supporting transclusions (bidirectional links etc.). But there is another restriction with the image tag: It is not possible to only include a part (e.g. “the upper left corner”) of an image. Therefore an image must be used as it is – even in a different size – or a copy of that image must be prepared.

To summarize: Existing images can be reused exactly as they are in different contexts without physically copying them. Including images in HTML is therefore a very restricted version of transclusion.

[Pam, 1996] proposed simple tags for text inclusions similar to the `IMG`-tag. However, these tags do not provide any backlinks from the quote to the document using that quote.

The HTML version 4-Recommendation ([Raggett et al., 1999]) in late 1999 defines some possibilities to include other kinds of documents via the `OBJECT` and `IFRAME`-tag. The `OBJECT`-tag may include single objects of any type and `IFRAME` (inline frame) can be used to include a whole HTML-document. However, the same problems as with inline images remain: addressing parts of a document is not possible and backlinks to the original context are not supported by these tags. Please note, that links are stored *in* the HTML document, i.e. try to link from one HTML document (document A) to some other document (document B) implies a modification of document A! Creating a link from document B back to document A results in a modification of docu-

ment B. Therefore a document management system must aid the user when performing such modifications. It is possible to implement bidirectional links based on HTML files if the links are parsed by the publishing system and are stored in some link-database. As noted above bidirectional links are an absolute requirement for transclusions. Please see section 3.4 and [Maurer, 1996] for a detailed discussion of bidirectional links.

3.3.2 Extensible Markup Language

The Extensible Markup Language (XML) is a very general format for structured documents and data on the Web. XML is derived from Standard Generalized Markup Language (SGML) but is much easier to handle for the user. Several working groups are building recommendations and standards. The most interesting parts in the XML community for our discussion of transclusions are the parts from the XML Path Language (XPath, [Clark and DeRose, 1999]) and XML Linking Language (XLink, [DeRose et al., 2001]).

With XPath it is possible to address “nodes” of documents, e.g. chapters, paragraphs and words. In the transclusion context additional features like “identify the selection made by the user with the mouse” must be available, therefore an extension to XPath, the XML Pointing Language (XPointer, [DeRose et al., 2001]) was specified by the W3C. XPointer is at the time of writing a “Candidate Recommendation” and some implementations already exist. Hence XPointer can be used to identify parts of an XML document and can therefore be used for defining the quote to be inserted in another document. XML media types (`text/xml`, `application/xml`, `text/xml-external-parsed-entity` and `application/xml-external-parsed-entity`) are supported at the outset, therefore using HTML-documents as source of the quote requires additional format conversion or implementation of extension facilities. Internal structures of XML

documents are addressed by element types, attribute values, character content, and relative positions.

Linking in XML (XLink) is much more powerful than in HTML. As mentioned above, in HTML links are stored inside the document. In XML links may be stored outside of the document in some link database and therefore also read only documents (like documents stored on a CD-ROM or remote WWW server) may be interlinked by the user. Not only unidirectional links, but also more complex link structures are provided by XLink. It is possible to interlink more than two resources and to associate meta data with the link element. The destination of a link is defined via XPointer described above.

XML Inclusions (XInclude, [Marsh and Orchard, 2001]) is another specification where a processing model and syntax for general purpose inclusion is discussed. It differs from XLink links with attribute `show=embed` and provides a media type specific (XML into XML) transformation.

This was a very short overview of some related document formats and specifications in our discussion about transclusions. In the last months some W3C workingdrafts of XML specifications became recommendations. We expect some full implementations of XPointer and XLink very soon. The best technology will not survive if the browsers available to users are not supporting it. At the time of writing webbrowsers support XML rudimentarily and information providers have to deal primarily with HTML. Therefore either XML to HTML processing must be performed at the server side or sophisticated server systems must combine HTML-documents and XML-linking in some way. Let us now take a look at some aspects of a server system supporting the transclusion idea.

3.4 Server Systems

Usually at least two server systems are involved when talking about translusions: The system where the original quote is physically stored, and the system where the *compound document* (document using that quote) is stored. Please note that also a “cascading composition” of translusions is allowed, i.e. parts of a document might be included which itself uses quotes from some other document. We will call the representation of the quote in the compound document a *quote window*.

In this section we take a closer look at some requirements for the different server systems involved when processing documents. General requirements like availability, fault tolerance etc. are not discussed here. If the server which holds the quote is not available because of a server break down or any other reason, some message will appear to inform the reader of the *broken information*.

When looking at the compound document, additional navigational tools must be displayed to the readers allowing them to jump to the original context of the quote (“two way reading”). The whole quote must be clearly rendered as quote to the reader.

There are different kinds of quotes: static and dynamic ones. An author of a compound document must clearly define which kind of quote is to be used in the document. Static quotes should change neither the content nor the layout over time. However, if changes occur, the author of the compound document must be informed and it must be clearly indicated to the reader that the displayed content of the quote is not the quote the author originally used. The quote may also completely disappear, depending on the attribute of the quote set by the author. If bidirectional links are supported by the

involved systems, the system providing the quote can inform the author of the compound document about a modification. If they are not supported, the server holding the compound document must do this task. In HTTP a simple “GET” request can check for any changes of the content of a document (“if-modified-since” parameter). Therefore the server system where the compound document is stored must check all quotes on a regular basis and must inform the author accordingly. The author then must react via some webform or via email.

If the quote is dynamic – e.g. “the joke of the day” or some weather forecasts – the structure of the content should not change and the quote must be identified in a uniform manner e.g. by identifier tags. However, if the system cannot find the quote, the author must be informed.

It might happen that the context of the quote changes, but not the quote itself. The discussed procedure to detect changes in documents will therefore notify the author of the compound document needlessly. Implementing a simple checksum of the quote and storing that checksum as attribute of the quote window will prevent such notifications.

Changes in the document where the quote is stored are very dangerous for the author of compound documents, therefore the server systems involved should provide version control facilities to authors of WWW documents. If such facilities are available, users might see a complete timeline of a quote.

While checking the status of the quote it may happen that the document where the quote is stored cannot be accessed. The system must then determine if the document was removed permanently or the service is temporarily unavailable. If the document was removed, the author again must be informed and the quote window will be automatically deactivated by the system. Handling *broken*

information is a key feature of a transclusion system. Therefore an easy to use and understandable interface has to be implemented.

As noted above, bidirectional links are of great help for both reader and author of documents. They add additional information to the documents and can answer quite difficult questions like “Who uses parts of this document?” or “What is the original context of that quote”. However, a very popular document quoted by many other authors may provide the reader with too much unwanted information. Therefore it must be possible to filter incoming links depending on various attributes, e.g. “show just incoming links from specific server systems”.

The separation of document content and linking information is an absolute requirement for both systems. Unfortunately not many systems available provide this paradigm. XML and XLink, respectively, may solve this problem in the future.

3.5 Implementing Transclusions

In this section we discuss some ideas of how to implement transclusions using an existing, very sophisticated information system named Hyperwave Information Server (HIS, e.g. [Maurer, 1996]). Many necessary features for transclusions are already built into the system and therefore the effort of implementation will be minimal.

Since users’ browsers are not predictable, we assume standard WWW browsers, capable of rendering HTML documents. Therefore all of the including and parsing processes must be performed on the server side. In an initial scenario we consider an HIS as server system for compound documents and some HTTP server as server serving the quote to be included.

As the discussion above showed, separation of content and linking information is an absolute requirement when building a system which supports translusions. This implies the use of a link database which is already built into HIS. If the document format itself does not support this separation – like HTML – the systems have to handle this task. To give an example: When a document in HTML-format is uploaded to the system all the links stored *in* the document are parsed and *link objects* are created by the system. These link objects – holding some attributes, like exact position of the link and linktarget etc. – are then controlled by the link database. This means that after operations on document objects – like (re-)moving, renaming etc. – the links are still valid and the common HTTP-error '404 – Requested URL not found' is avoided. This link database enables also arbitrary attributes to be stored with the link object. It is obvious that links to objects which are not controlled by the system (i.e. links to some different server system) may become inconsistent. To have full control over link visualization we suggest to include *remote objects* for every “external” link. *Remote objects* are representations of non-local (i.e. remote) stored data in HIS. Several service programs running regularly may then check for existence or modification of the target of the link.

The same mechanism can be used for implementing translusions on an HIS. The remote object is a representation of the “quote window” in the system. Special attributes enables a distinction between “ordinary” remote objects and special translusion objects. The displaying process have to evaluate the attributes stored with the object and perform the appropriate actions.

Much work has been done to define exactly some part of the HTML-page (see e.g. [Poon and Kontogiannis, 2001] or [Huck et al., 1998]). Graphical tools like the WysiWyg Web Wrapper Factory (W4F, [Sahuguet and Azavant, 1999]) make it quite easy for the user to specify the textregion to quote. Depending

on the tool we build into the system, the quote window must store the definition accordingly.

The remote object is responsible for retrieving the content selected. For HTML documents, the exact definition of the quote must be in a syntax similar to XPointer. Parts in plain text documents must be addressed via a simple string range (like the “string-range” definition of XPointer). Please note that not only text based document formats, but also multimedia documents like images, sound, video etc. can be included with this approach. The resulting HTML page is assembled on the server side, therefore simple HTML-links to the original context of the quote can be inserted automatically to enable “two way reading”.

The obvious drawback with this generic solution is the missing backlink from the quote to the document. Some systems are trying to solve this problem by executing special queries on search engines trying to get all of the documents pointing to the document containing the quote. However, this approach is inappropriate because only documents indexed by that search engine are returned. Furthermore it is not possible to jump immediately to the exact position where the quote was used. If the quote is also stored on some Hyperwave Information Server it is possible to make use of the integrated link database and display a backlink to the compound document. A timeline via different versions of the quote is also possible when the quote itself is stored on some Hyperwave Information Server. Therefore “real” two way reading can be implemented for the benefit of the reader.

For the first implementation we assume as source format of the compound- and inlined document HTML. Even with this simple file format problems may arise when quoting parts of a HTML document. The text might either be quoted without the HTML-tags or the system rejects a transclusion if corrupt

syntax is found in the quote. However, certain tags (like `</HTML>` etc.) must not appear in the quote.

In a universal transclusion system it must be possible to quote non-HTML documents. A user may want to transclude parts of a PDF-file or any other text format. As long as the transcluded document format can be translated in some ways into a common format, transclusions are possible.

3.6 Applications for Transclusions

Especially in scientific publications it is necessary to access the sources of a quote. Therefore our aim is to provide this transclusion feature in the Journal of Universal Computer Science ([J.UCS, 2001]) to make it possible for the authors to use transclusions in the HTML format of the article.

Another application of transclusions is version control. Imagine a document where a word is replaced. The new version of the document will then only contain everything before the word as transclusion, the new replaced word, and the rest of the document as transclusion. Also shifting the position of two paragraphs can be implemented without wasting disk space.

An author of documents with inclusions may also use the standard rights and user management tools available with a Hyperwave Information Server. Using these features enables different representations of one and the same document for different users or user groups. Combined with user preferences, this is a very flexible way for providing content.

3.7 Conclusion and Future Work

This paper shows that the concept of transclusion is still as important today as it was when proposed 40 years ago. Amazingly few implementations exist. The described structure on a Hyperwave Information Server enables inclusions of some text-sources available on the web. In a first implementation we consider intraserver documents to reduce the known problems of server breakdowns and communications failures. Transclusions of locally available sources of text, images and other data is not available but there is a need for it. Therefore, future work will address development of tools to enable inclusions independent of location and format of data.

References

- [Berners-Lee et al., 1996] Berners-Lee, T., Fielding, R., and Frystyk, H. (1996). Hypertext Transfer Protocol – HTTP/1.0. RFC 1945. <ftp://ftp.isi.edu/in-notes/rfc1945.txt>
- [Berners-Lee et al., 1998] Berners-Lee, T., Fielding, R., Irvine, U., and Masinter, L. (1998). Uniform Resource Identifiers (URI). RFC 2396. <ftp://ftp.isi.edu/in-notes/rfc2396.txt>
- [Berners-Lee et al., 1994] Berners-Lee, T., Masinter, L., and McCahill, M. (1994). Uniform Resource Locators. RFC 1738. <ftp://ftp.isi.edu/in-notes/rfc1738.txt>
- [BMM3, 2000] BMM3 (2000). Brockhaus Multimedia. <http://www.brockhaus-multimedia.de>.
- [Clark and DeRose, 1999] Clark, J. and DeRose, S. (1999). XML Path Language (XPath) Version 1.0. <http://www.w3.org/TR/1999/REC-xpath-19991116>.
- [CVS-HOME, 2001] CVS-HOME (2001). Concurrent versions system homepage. <http://www.cvshome.org>.
- [DeRose et al., 2001] DeRose, S., Maler, E., and Orchard, D. (2001). XML Linking Language (xlink) Version 1.0. <http://www.w3.org/TR/xlink>.
- [DeRose et al., 2001] DeRose, S., Maler, E., and Jr., R. D. (2001). XML Pointer Language (XPointer) Version 1.0. <http://www.w3.org/TR/2001/CR-xptr-20010911>.

- [Fielding et al., 1999] Fielding, R., Gettys, J., and Mogul, J. (1999). Hypertext Transfer Protocol – HTTP/1.1. RFC 2616. <ftp://ftp.isi.edu/in-notes/rfc2616.txt>
- [Huck et al., 1998] Huck, G., Fankhauser, P., Aberer, K., and Neuhold, E. (1998). JEDI: Extracting and Synthesizing Information from the Web. In *Cooperative Information Systems (COOPIS)*. pages 32–43.
- [J.UCS, 2001] J.UCS (2001). Journal of Universal Computer Science. <http://www.jucs.org>.
- [Krottmaier and Maurer, 2001] Krottmaier, H. and Maurer, H. (2001). Transclusions in the 21st Century. *Journal of Universal Computer Science*, 7(12):1125–1136. http://www.jucs.org/jucs_7_12/transclusions_in_the_21st.
- [Marsh and Orchard, 2001] Marsh, J. and Orchard, D. (2001). XML Inclusions (XInclude) Version 1.0. <http://www.w3.org/TR/2001/WD-xinclude-20010516>.
- [Maurer, 1996] Maurer, H. (1996). Hyper-G now Hyperwave — The Next Generation Web Solution. Addison-Wesley
- [Nelson, 1987] Nelson, T. (1987). *Literary Machines*. Ted Nelson.
- [Nelson, 1995] Nelson, T. (1995). The Heart of Connection: Hypermedia Unified Transclusion. *Communications of the ACM*, 38:31–33.
- [Pam, 1996] Pam, A. (1996). Methods for implementing transclusion of text into HTML pages. Technical report, Xanadu Australia. <http://xanadu.com.au/archive/transclude.html>

- [Poon and Kontogiannis, 2001] Poon, F. and Kontogiannis, K. (2001). i-Cube: A Tool-Set for the Dynamic Extraction and Integration of Web Data. *LNCS 2040*, pages 98–115.
- [Raggett et al., 1999] Raggett, D., Hors, A. L., and Jacobs, I. (1999). HTML 4.01 Specification, W3C Recommendation. <http://www.w3.org/TR/html4>.
- [Sahuguet and Azavant, 1999] Sahuguet, A. and Azavant, F. (1999). Wysiwyg web wrapper factory. <http://db.cis.upenn.edu/cgi-bin/Person.perl?sahuguet>
- [W3C, 2000] W3C (2000). Extensible Markup Language (XML) 1.0 (Second Edition). <http://www.w3.org/TR/2000/REC-xml-20001006>.
- [W3C, 2001a] W3C (2001a). HyperText Markup Language. <http://www.w3.org/MarkUp>.
- [W3C, 2001b] W3C (2001b). Micropayments Markup Working Group. <http://www.w3.org/ECommerce/Micropayments>.

Chapter 4

Further Aspects of Transclusions

After the first results about transclusions were published in late 2001, discussions about the need of transclusions and advantages of them over the common cut-and-paste approach were published and discussed at different conferences. One article (“Issues of Transclusions”) was presented at the “World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education” (E-Learn 2002) in Montreal, Canada. The other article (“Transcluded Documents: Advantages of Reusing Document Fragments”) was presented at the “6th International Conference on Electronic Publishing” (ELPUB 2002) in Karlovy Vary, Czech Republic. The original contributions can be found in [Krottmaier and Helic, 2002a] and [Krottmaier, 2002d].

In the following sections a summary of the results is given. Aspects of creating, visualizing as well as further applications of transclusions are presented.

4.1 Creation of Transclusions

As mentioned in section 3.4, it must be simple for users to reuse existing parts of documents. The appropriate parts of a document to be transcluded must be selected and defined. Thereafter a set of attributes must be defined to control the behavior of the transclusion in the visualization- (see section 4.2) and update-process (see section 4.3). A representation of the transcluded part must be created at the server-side. In this section we discuss the creation of transclusions.

The following steps must be done by the user and the system to create transclusions:

1. Select the appropriate part to be transcluded
2. Create a representation of the selected part at the server-side and establish a connection between this representation and the original part
3. Use this representation to refer to the original content
4. Define the behavior of transclusion (type of transclusion)

A “transclusion wizard” must support the user in the creation of transclusions. The part to be reused must be selected via this wizard, thereafter the tool must create appropriate objects at the server-side. If the document is not immediately created at the server-side, markups must be created in the document-source. Different types of transclusions will manage changes differently. For a detailed discussion about different types see section 4.3.

The document to be transcluded must be available via an electronic information system. Therefore it is not immediately possible to transclude parts of a

“traditional resource”, i.e. printed resources such as books, journals etc. However, publishers provide sometimes electronic versions of books and the user may use this electronic representation of the book. If no electronic version exists a scanner or digital camera may be used to create an “electronic surrogate”. By using standard optical character recognition (OCR) technology it is possible to transform a scanned image into text.

It turns out that using an image as a representation of the part to be transcluded solves many problems. Different electronic document formats which are displayed at a screen may be transcluded simply by creating a screen-shot of the related part using the transclusion wizard. Additionally, the wizard must track the exact storage location (via a Uniform Resource Identifier, URI, see [Berners-Lee et al., 1998], with an XPointer-expression, see [DeRose et al., 2001a]) of the original source document. Usage of an image instead of the original text document allows us to include arbitrary text documents without knowing the internal electronic format of the document.

An application such as the one described above creates several representations of the part to be transcluded: (1) there is an image-representation of the part, (2) the wording of the content, and (3) a URI with XPointer-expression to refer to the original document.

These representations of the transcluded part must be stored on the server-side. This information will be stored in a special *collection*. This collection may contain an arbitrary amount of data, therefore no restrictions to future extensions are given. The image-representation and the wording are stored as separate objects in the collection. The URI with the appropriate XPointer-expression is stored as an attribute in the transclusion-collection. If the same part is reused in another document, this collection is simply “linked” to the location of the other transclusion.

In [Krottmaier, 2002d] another workflow for creating translusions is introduced. Using \LaTeX for the creation of a new document allows the implementation of macros, which are then automatically transformed into the described structure at the server-side. In our first discussions we wanted to store the URI with the appropriate XPointer-expression. However, this simple approach has several disadvantages. With this concept it is not possible to reuse arbitrarily different text-based document formats. \LaTeX was chosen as source document format for translusions for various reasons:

- \LaTeX is well known in the scientific community
- \LaTeX is very often the source-format of high-quality publications.
- Different formatted documents can be produced out of \LaTeX -formatted input, including PDF, PostScript, HTML and XML without any user-interaction.
- \LaTeX -formatted documents are easy to parse because they are written and stored in plain-ASCII format.

A macro for using translusions may look like the following:

```
\tranclude[parameters]{URI}{XPointer-Expression}.
```

Using this approach – as well as the previously discussed one – needs additional software for selecting the part to be transluded. Much work has been done to define fragments of an HTML-page (see e.g. [Poon and Kontogiannis, 2001] or [Huck et al., 1998]). Graphical tools (such as the WysiWyg Web Wrapper Factory, W4F, [Sahuguet and Azavant, 1999]) make it simple for the user to specify a text-region to be included in the new document. The quote is identified by an XPointer-expression. However, usage of these tools limits translusions to HTML and XML based documents. It would not easily be

possible to transclude parts of a PDF-document or any other difficult to parse documents (such as Winword-documents etc.). Using an image representation of the transcluded part – as described above – will relieve the burden of different source documents for transclusions.

Another advantage is the possibility to simply include an image representation in the newly created document. Therefore the original document must not be available during the visualization process. Other aspects of visualization of documents is discussed in the following section.

The cut-and-paste approach has many disadvantages over transclusions, but there are also advantages! The best approach is to combine these two concepts! It is necessary to create a local representation of the part to be transcluded. “Broken information” is not what the author or reader of the compound document wants. Using local copies (either via text-representation, image-representation or both kinds) will always represent the document in a proper way. If the content of the original document is not accessible, the concept of “conditional text” must be implemented (see section 4.3).

However, cut-and-paste without referring to the reused part via bidirectional links (see section 3.4) is harmful. The visualization process described in the following section must include links to the original context.

4.2 Visualization of Transclusions

As mentioned in [Nelson, 1987], reused parts must be visible to the user. Visualization in both involved documents – the transcluded part in the document using transclusions *and* the reused part in the original document – must be adaptable by the reader. A user profile stored on the server side must store the default visualization parameters. Transclusions must provide (1) the con-

tent of the transcluded part and (2) the appropriate linking information to the original document.

When displaying a compound document a user must decide whether to display linking information to the original context or not. Depending on user's preferences the link-source should be a well-defined image, the whole URI of the original document. The location of this link should also be defined by the user: one user may prefer this linking-information before and/or after the transclusion or even in a footnote or special section of the document. Every mentioned possibility must be implemented in a rendering engine.

Creating transclusions as described in the previous section (with several representations of the reused part) would make it easy to display the reused part even if the original document is not available at the time of rendering the compound document. Either the image or the wording of the transcluded part must be included in the new document. User-preferences should allow highlighting in different colors. Colors must take the "importance" of the reused document into account. Importance of a document may be measured by counting citations to the document or by evaluating reused parts of the document. An often cited or reused document is probably more important than a never-cited document.

In figure 4.1 different user-preferences are used to render a single page with one transcluded paragraph. In the left two pictures different colors are used to highlight the transcluded paragraph. The right picture also displays a link to the original context of the paragraph at the lower right-hand side of the paragraph.

Visualization of the transcluded document (i.e. the original document) should also be adaptable by the user. A simple button in the user-interface should do a special visualization of reused parts. The implementation should either simply



Figure 4.1: Compound document: different visualizations depending on user-preferences

draw a frame around the reused parts or highlight them. Colors or any other markup should be used to reflect the importance of the parts. Additionally, it must be possible to jump directly to the compound documents, where the highlighted part is reused. Links should be visualized on request.

4.3 Managing Changes: Conditional Text

When a transclusion is created it must be defined by the author of the new document what should happen if the original transcluded part changes. In some applications it is necessary to regard the whole original document. Changes may be of several types:

Changes in Layout: Usually changes in the layout of a quote or the whole transcluded document does not demand any further action. However, the system must detect changes of this type and it must be possible to react via an action on such a change. Mechanisms for detecting changes in HTML formatted content are described in [Francisco-Revilla et al., 2001].

Changes in Content: It must be possible in some applications to react on changes of the content. Consider the example in section 3.4: “joke of the day”. When using dynamic quotes the current contents – in this example the joke – should be displayed in the compound document. As the author of the compound document expects changes in the content, no notification about this change must be sent to the author. In other applications it might be necessary to inform the author of the compound document.

Changes in Structure: In learning environments the structure of a document might be discussed or explained. Therefore any changes in the structure are of great interest to the author of the compound document. The system must therefore detect this type of change, e.g. when new structure elements were added or remove from the contents.

Different types of changes may be handled by different actions: either the author of the compound and/or transcluded document is informed via email or any other notification, or changes are incorporated automatically into the compound document by the system. The “transclusion-wizard” described in section 4.1 should offer options for actions depending on the type of change.

In certain circumstances it is not possible to access the original document. To mention just a few reasons: (1) original document is moved, (2) removed, or (3) the server is down. When transclusions are used in an intra-server or intranet environment, it is likely that transclusions are valid and the original document is accessible. Administrators may prohibit deletion of reused documents. However, on the Internet problems may occur when the original serversystem is unreachable (“broken information”). Therefore the transclusion cannot be checked for changes, contents of the quotation window is in-

accessible. To handle this state, let us introduce the concept of *conditional text*.

Conditional text depends on the existence and availability of the quotation or transclusion. It is similar to an `if`-statement in programming languages. Each transclusion must be identified by a unique identifier. If a transclusion is valid, then the appropriate text is displayed, otherwise an alternative text is displayed to the user. Let us give an example in XML pseudo-code:

```
[...]
<conditionalText transclusion_id="some_id">
  <valid_transclusion>
    In the following section...
  </valid_transclusion>
  <invalid_transclusion>
    To give a short summary...
  </invalid_transclusion>
</conditionalText>
[some more text...]
<transclusion id="some_id">
  <uri>http://www.xyz.org/abc/index.html</uri>
  <xpointer>someParagraph to someOtherParagraph</xpointer>
  <paramter>[types of action]</parameter>
</transclusion>
```

When using a highly sophisticated server system for publishing compound documents, it is possible, to restrict access to certain transclusions. A single document using transclusions will be displayed differently to different users. Therefore a completely new paradigm of publishing will be established. Nevertheless, user acceptance tests must be performed and explored.

As described in this section changes must be handled in different ways. If the original content is not available, authors must be informed by the system to perform further actions. However, the concepts of using the cut-and-paste approach in addition to the transclusion-concept makes this type of publishing very powerful. Combined with the idea of conditional text a very sophisticated publishing environment may be established. In the next section we take a closer look at different applications of transclusions.

4.4 Concrete Applications of Transclusions

With the concept of transclusions in mind it is possible to enhance different existing applications. The situation of every involved participant (readers and authors) of every document used in the concept will be improved.

4.4.1 Scientific Electronic Publishing

Scholarly papers often quote paragraphs or sentences of other already published papers. When using transclusions, readers of a new paper (compound document) are able to jump directly to the context of the quote (i.e. to the original paper). Therefore it is possible for the reader of the compound document to explore the reused quote in the original context. This issue is especially useful for novices in a topic. The author of the compound document may create a special “view” of a topic. On the one hand, readers who are interested in the details may read the original contributions to the topic. Readers of the transcluded document on the other hand are able to see which parts of a document are reused in other documents. This is very important information! If a part of a document is reused in some other document, it is likely that this part is important. This additional “meta-data” (“This part is also used

in some other document(s)...”) is a kind of “implicit rating” of a fragment of a document. In addition to attributes in link-objects (whether the part is used as a positive or negative example) useful information is added to the transcluded document as well!

4.4.2 Discussion Forum

In electronic written communication (especially in electronic discussion forums like bulletin boards or news groups etc.) many new postings refer to previous postings. These postings are (usually) immutable. In electronic journals or libraries this type of communication is also very common (e.g. “letters to the editor” often refer to parts of previous published articles). Referring to parts of an article should be possible by simply marking the fragment to be included in the new document or posting. Readers of the new article can be sure that the referenced part is not modified by the author of the new article. Also readers of the “old” (i.e. transcluded) document are able to see which parts are discussed later in some other posting. It is likely – if the system is used in the right way and users are used to it – that questions will be asked just once. A much more efficient communication will be the result when using translusions in this type of application. This additional structure in discussion threads will save time.

4.4.3 Course Material

Very often existing course material is simply a collection of (unidirectional) links to related material stored on different server systems. Unfortunately, it is not possible with current web technology to link to parts of existing material and to create links back to the collection. A very simple example will clarify this situation: imagine a single very important paragraph in an article

of 100 pages. The author of the course material may add a link exactly to the paragraph, but cannot include a link in the (remote stored) material *back to* the collection! Therefore the author of the collection has to cut-and-paste paragraphs of existing material into new documents. The original contexts of these paragraphs are lost. Using transclusions in this type of application will enhance usability and will add metadata to the reused part! Other readers will benefit from this information.

4.5 Conclusion

This chapter summarizes two conference contributions related to transclusions. Further aspects of the implementation of transclusions are discussed. The introduced concept provides different kinds of representations of the reused parts. Using an image of the quote will solve some of the problems of different document formats. The text representation allows better integration into the type-face of the new document. User-adaptive visualization of transcluded parts will support readers in finding the real important parts of a document. Several applications of transclusions are described to emphasize the need of transclusion-aware systems.

References

- [Berners-Lee et al., 1998] Berners-Lee, T., Fielding, R., Irvine, U.C., and Masinter, L. (1998). Uniform Resource Identifiers (URI). August 1998. RFC 2396. <ftp://ftp.isi.edu/in-notes/rfc2396.txt>
- [DeRose et al., 2001a] DeRose, S., Maler, E., and Daniel Jr., R. (2001a). XML Pointer Language (XPointer) Version 1.0. <http://www.w3.org/TR/2001/CR-xptr-20010911>.
- [DeRose et al., 2001b] DeRose, S., Maler, E., and Orchard, D. (2001b). XML Linking Language (xlink) Version 1.0. <http://www.w3.org/TR/xlink>.
- [Francisco-Revilla et al., 2001] Francisco-Revilla, L., Shipman, F., Furuta, R., Karadkar, U., and Arora, A. (2001). Managing change on the web. In *Proceedings of the first ACM/IEEE-CS joint conference on Digital libraries*, pages 67–76. ACM Press.
- [Huck et al., 1998] Huck, G., Fankhauser, P., Aberer, K., and Neuhold, E. (1998). JEDI: Extracting and Synthesizing Information from the Web. In *Cooperative Information Systems (COOPIS)*. pages 32–43, 1998
- [Hyperwave, 2001] Hyperwave (2001). Hyperwave Information Server. <http://www.hyperwave.com>.
- [J.UCS, 2002] J.UCS (2002). Journal of Universal Computer Science. <http://www.jucs.org>.

- [Krottmaier, 2002d] Krottmaier, H. (2002d). Transcluded Documents: Advantages of Reusing Document Fragments. In Carvalho, J. A., Hübler, A., and Baptista, A. A., editors, *Advances in Media Technology (ELPUB 2002)*, pages 359–367. International Council for Computer Communication (ICCC) and International Federation for Information Processing (IFIP).
- [Krottmaier and Helic, 2002a] Krottmaier, H. and Helic, D. (2002a). Issues of Transclusions. In *Proceedings of E-Learn (E-Learn 2002)*, pages 1730–1733, Montreal, Canada.
- [Krottmaier and Maurer, 2001] Krottmaier, H. and Maurer, H. (2001). Transclusions in the 21st Century. *Journal of Universal Computer Science*, 7(12):1125–1136. http://www.jucs.org/jucs_7_12/transclusions_in_the_21st.
- [Lawrence et al., 1999] Lawrence, S., Lee Giles, C., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71, 1999.
- [Maurer, 1996] Maurer, H. (1996). Hyper-G now Hyperwave — The Next Generation Web Solution. Addison-Wesley
- [NEC Research Institute, 2002] NEC Research Institute (2002). Researchindex. <http://www.researchindex.com>.
- [Nelson, 1987] Nelson, T. (1987) *Literary Machines*. Ted Nelson, 1987.
- [Nelson, 1995] Nelson, T. (1995). The Heart of Connection: Hypermedia Unified Transclusion. *Communications of the ACM*, 38:31–33.
- [Poon and Kontogiannis, 2001] Poon, F. and Kontogiannis, K. (2001). i-Cube: A Tool-Set for the Dynamic Extraction and Integration of Web Data. *LNCS 2040*, pages 98–115, 2001.

- [Sahuguet and Azavant, 1999] Sahuguet, A. and Azavant, F. (1999). Wysi-
Wyg Web Wrapper Factory [http://db.cis.upenn.edu/cgi-bin/
Person.perl?sahuguet](http://db.cis.upenn.edu/cgi-bin/Person.perl?sahuguet)

Part III

Working with Digital Journals

“Everything should be made as simple as possible, but not simpler.” *Albert Einstein*

Chapter 5

Improving the Usability of a Digital Library

This paper was presented at the “5th International Conference on Electronic Publishing” (ELPUB 2001) in Canterbury, United Kingdom.

Working with the material stored in an electronic publishing system is essential to make the usage of the system as easy as possible. This paper introduces upcoming developments in the field of electronic publishing systems and discusses a few features which are essential for users. To make the features available to all users, we propose a server-side implementation using standard web-technology. An implementation of a personal workspace is introduced in detail. Shared and active bookmarks are just two applications of the introduced personal space in a digital library. The power of many users is demonstrated. Please note that at the time of writing these features were not well known in the community.

The original contribution was published in the proceedings of the conference and can be found in [Krottmaier, 2001].

5.1 Abstract

The Journal of Universal Computer Science (J.UCS) was introduced in 1994 and after more than 7 years of operation the service is still up and running. Articles published on the server are categorized in several ways (e.g. in the ACM Classification Scheme) to simplify browsing and finding articles again. Nevertheless users want to create their own view of the material and their own categories. Traditionally, a user creates some categories on the client-side in a bookmark-structure and stores the URLs of the according papers in this structure. But the use of bookmarks binds the user to a specific client on one single machine and so it is not possible to take full advantage of the restructured view. This is just one reason why it is helpful for users to manage their personal view of the articles on the server-side. In this paper we introduce a personal workspace for registered users of the Journal of Universal Computer Science. It is possible to create and modify a personal view, add some personal and/or public comments of the articles and use the workspace as personal repository of published articles. We also show other advantages like personalized search scope, customization of the order of articles and some other useful features of the system.

5.2 Introduction

Typical digital libraries are static and passive systems. They do not adapt to users preferences, like colors and design of the interface, topic of interest, rearrangement of articles etc. Because every user is different in many ways, this issue must be considered in the design of an easy to use and helpful digital library. “Passive” in the context of digital libraries means that the user interacts with the system, but the system does not initiate an interaction with

the user. This behavior can be improved by sophisticated tools described later in this document.

To get rid of the static behavior, user preferences must be stored at some place. Due to the generic accessibility of one system, a standard web-browser is usually used to access and visualize the content digital library. Therefore preferences might be stored on the client-side, proxy-server or server-side.

Storing data on the client-side in some cookies (e.g. [Clarke, 2001]) is a problem when mobility, backup etc. is an issue. Just to address the problem of mobility: If one changes the client-computer due to failure or any other reason, the preferences are not accessible and therefore the server-system cannot take these preferences into account when assembling the content of a page. Also if a user changes the type of browser on one single computer, the cookies stored for browser *a* are not accessible for browser *b*. These are just two reasons why it is not very reasonable to store personalized information in browsers.

A proxy-server is an intermediate server-system between client and server. If such a proxy is located in the intranet within some company, it will not be possible to access this intermediate server-system to get some user defined preferences. Therefore in some cases – e.g. for an exclusively company-wide digital library system this approach might be possible, but for an international, publicly available system only one storage location of user preferences is proper: the server-side. All advantages (browser and client-computer independent access, central backup etc.) and disadvantages (failure of server implies unavailability of the service to all clients etc.) are immanent in this approach. But independent availability is a key-issue in our system [Maurer and Schmaranz, 1994] so the decision was to store all user defined preferences at the server-side. Some requirements on the implementation of the workspace and some tools are described. These tools handle the problem of passivity

of typical digital library systems. The following section describes the requirements for the workspace which will be implemented for the Journal of Universal Computer Science (J.UCS) to provide the full potential of a digital library.

5.3 Requirements for a Workspace

As described above the workspace in our context is some storage place on the server. Another term for a workspace might be “Homecollection” or “Homepage” with some additional tools. Some parts might be accessible for all users or for some groups, other parts are private and therefore invisible for other users. Like in traditional libraries, there is a need for classification of the content. In J.UCS the categorization is done in several standard ways:

ACM Classification System: The first ACM classification system for the computing field was published in 1964. This system was updated several times to consider new topics in computer science [ACM, 1998] and makes it easy for a reader to access immediately the topic of interest without wasting time to find articles on a specific topic. For example, if one is interested in the topic of digital libraries, one will find related articles in category H (H...Information Systems), aspects of Information Interface and Presentation are found in category H.5, whereas the sub aspects of user interfaces in information systems are found in category H.5.2 User Interface.

By Time: Obviously, newer articles are often more interesting to users than old ones. There must be a simple distinction in the time-line of articles. Articles are published in monthly issues and are bound into a volume for each year. This makes it very easy for the user to recognize the time of publication of each article.

By Author: If one is interested in all papers published by one author, the related papers can be found simply by browsing the corresponding collection. Due to the many articles already published in J.UCS, there is a need for further granulation of articles.

Please note that all these listings are generated on the fly and are therefore every time up to date. As a requirement for a private workspace, it must be possible to allow the users to create their own views on the material (i.e. articles, issues or volumes). If the user is interested in e.g. information systems (category H), why should the user see the articles of category F (Theory of Computation)?

To put this idea into practice users must define their topics of interest. This can be done in several ways: Either one user links the collection of category H to his assigned workspace, or the displaying routine for the category listing consider specific variables set by the user via some form. To take full advantage of the parent/children-relations of a Hyperwave Information Server (e.g. [Maurer, 1996], [Hyperwave, 2001]) it is clear that the first approach (link some part – e.g. a paper or any other entity of the content – to the workspace) must be implemented. This can be either done automatically (via some cgi-script or some server-side Java Script) if the user fills out some form, or manually by the user using some tools. If some users decide that their own categories should be available for other users, they simply assign the appropriate rights to their categories.

This approach immanently implements “public bookmarks” and can be used in a background-library for some learning environment or web based training system. Note the big difference of that approach to the discussed possibilities described above. Unlike client-side bookmarks this workspace is not just available to other users (if the owner sets appropriate rights), but the structure

is also available within *any* web-browser. Administrators are able to extract valuable information from the linking structure of the content and can thereby advise the editors about topics often viewed by users. A system explicitly implementing such a public/private bookmark-service is e.g. [MyBookmarks, 2001] but it is obvious that a library system should implement this feature to take full advantage of the system.

One can imagine that due to personal preferences and intention of some collection, the sort order of the linked entities may differ. To give an example: One may collect some articles in one collection "*History of Information Systems*" and therefore wants some sort order of the articles ascending in date of publication. Another user might create a collection with the same articles "*Latest Developments in Information Systems*", but with another sort order (descending in date of publication). It is obvious that no physical copies are allowed but simple transclusions (reference links, see e.g. [Nelson, 1987]) must be implemented.

If other arbitrary sort-orders of articles are desirable, context specific attributes to the articles must be implemented. This can either be achieved via specific container objects or via attributes in the parent collection of the linked articles. Another feature easily implemented with that technology is a global history of what the user has accessed on the server system.

One can see that this kind of workspace has some powerful features and will make the life much easier for users of the digital library. The next step of implementation will allow to rearrange and collect material (i.e. everything addressable by an Uniform Resource Identifier (URI, [Berners-Lee et al., 1998]) not stored on the local Hyperwave Information Server but anywhere on the web.

As pointed out above a personal workspace can improve the usability of the library enormously. But so far, only the navigation through the material of the library and finding articles again is improved in comparison to static libraries. In the following section some improvements in existing functions and new functions provided by the library are described.

5.4 More than just Reading...

As mentioned in section 5.2, a user should work with the library and not just read articles in it. It is obvious that a personal workspace on the library server is just the first step to a useful environment.

Traditional features for adding some kind of value to the articles are annotations or comments to articles. J.UCS was one of the first systems implementing that feature. Nowadays nearly every electronic library system supports annotations in some way. Annotations are not necessarily “Letters to the Editor” about some article but small comments to a specific part of an article. They should either be public, public for a special user-group, or private. Special kinds of annotations may have a priceless value to the reader and the editor of the library. To give some examples:

Rating of Content: Annotations can be used to rate some article. If this feature is easy to use, users will work with it and after some users rated the article, a preselection of articles for other users will automatically occur. Several discussion systems like news server and appropriate clients already implement that kind of preselection of articles. One can see this rating process as a “review” process of the users using the library.

Link to the Future: Comments are not just text based. HTML tags are also allowed in annotations and therefore links to some external sources (identified by some URL) are possible. If the system has access to some citation index these “Links to the Future” can be created automatically by simply looking for documents referring to that specific paper. The question “Who uses this paper?” can be answered by search engines and if the result of that question can be interpreted by the system, links might be inserted automatically. On the other hand if one article is cited in a paper on the same server system, the cited article might be linked to the reference section of the paper automatically and the user have direct access to the cite.

Highlight / Cancel: Several programs (e.g. PC-Bibliothek [PC-Bib, 2001]) are supporting this feature of highlighting some text in some text-document. This is also a kind of annotation (an annotation probably without content but with several attributes like colors, rating, time of annotation etc.). Again, if this feature is easy to use and several persons highlight or cancel some text, an automatic summary can be generated by the system and reading speed will increase enormous! This human based text mining will be a powerful feature. Problems will occur when implementing this feature. One will be the lack of programming support of the different web browsers. When using Java Script as a common basis (supported by most of the browsers) it is possible to extract the value of the selected text part, but it is not possible to get the position information! Some tricks and assumptions must be performed (e.g. assume that the reader is highlighting the article in some “known” sequence) to get the exact highlighted part of the HTML document.

Another problem will be the handling of different formats of one article. In J.UCS one paper is published in different data formats like

PostScript, PDF and HTML. At the moment there are tools available for annotating HTML documents. The annotation feature is not available in PostScript-format documents, but in PDF documents. Therefore it must be possible, if one annotates the HTML-document at some point that this annotation is also available to the user in the PDF document using e.g. Acrobat Reader. Due to the fact that HTML is not a page description language it will be difficult to determine the exact position of the annotation in the PDF document especially if graphics are included in the document.

Questions and Answers: A special type of annotation is a question/answer dialog. As mentioned in [Maurer, 2001] this kind of annotation enables the implementation of “active documents”, documents which are able to automatically answer questions. In [Heinrich and Maurer, 2000] the basic idea of these documents is described: “The basic idea is that in the future, users of documents in any networked system should not just be able to communicate with other users, but also with documents.”

[Maurer, 2001] explains some requirements and shows the power of many users. If hundreds of thousands of persons are working with a document, many semantically equal questions will arrive. To give a description of the work flow: The document is published at the system and facilities are available to ask any question at any position in the document. Experts answers such questions. These answers and the related questions are recorded.

Experimental evidence shows that no new questions (just may be old questions in a new form) are asked after some 500 - 1000 viewers. Thus, if a document is viewed within, say, a year by 200.000 users, experts to answer questions are only required for a short time (until the first 1000 persons

have visited the document). After this, all other 199.000 users get their answer from the system, if only the system can tell whether two similarly looking questions are indeed semantically identical. ... In our example, only 0.5% of users get their questions answered by experts, the remaining 99.5% receive the answer instantaneously from the system [...].

As one can see the old idea of annotations is a powerful feature especially when many users are working with the library. Obviously, the tools must be available for every user on any client system. As discussed above, these tools must be as easy to use as a pencil. Annotations are not the only possible interactive feature. One can imagine links from some very specific technical terms in the text to some dictionary server. Some systems like the one used at Learning Network of “The New York Times” [Learning-NY-Times, 2001] have already implemented this useful feature.

Some large dictionaries like PC-Bibliothek [PC-Bib, 2001], Brockhaus’ Encyclopedia ([BMM, 2000], one of the largest available multimedia collections) are client-side implementations. If one wants to use a dictionary, one simply selects the unknown string, type a defined hot-key combination and the dictionary does its work. Therefore changing the laptop implies also changing of the desktop-environment and the loss of the installed dictionary (like in the traditional world, if one changes the office, all books needed must be transferred too...). The client/server paradigm makes live easier for some users.

Implementing links in the content to these dictionaries shows that user preferences are an essential requirement to realize this feature. An example will clarify this statement: Imagine two users of the library, a novice student of computer science and a professor of computer science. It might be nice, if there is a link from the term “Programming Language” to some basic com-

puter science dictionary for the student, but it makes no sense to create such a link for the professor! On the other hand if there are some newly developed programming languages which are maybe interesting for the professor, then a link will also be suitable for the professor, but the target is then obviously a different document as for the student. This implies, that not just “skill/experience” (e.g. in a range from 1 to 5) but also “interest” must be considered by the link-creation tool. Either the entry of a dictionary or the whole dictionary as such must be classified in a similar manner, i.e. a “Basic Dictionary Of Computer Science Terms” might be declared to be of interest for students (skill/experience ≤ 2) and for very interested people (interest ≥ 4). Note that it is absolutely necessary to evaluate the response of the users and to classify the dictionaries properly.

Currently the Dublin Core Education Working Group (DCEd) proposed a Working Draft to the Dublin Core Usage Committee of the Dublin Core Metadata Initiative (DCMI) to overcome this problem [Mason and Sutton, 2000] of describing the audience for some material. They introduced a element called “dc-ed:audience”. E.g. the Education Network Australia (EdNA, [EDN, 2001]) is in the process of implementing some vocabulary. Various education-specific thesauri contain audience classification descriptors (e.g., the Thesaurus of ERIC Descriptors and the NICEM Thesaurus, to name but a few) [Mason et al., 2000]. A discussion about different kinds of links (reference links, glossary links etc.) can be found in [Schmaranz, 1996].

One interesting topic currently discussed at the World Wide Web Consortium (W3C) is the Semantic Web [W3C, 2001]:

The Semantic Web is a vision: the idea of having data on the Web defined and linked in a way that it can be used by machines not

just for display purposes, but for automation, integration and reuse of data across various applications.

Several techniques are available to describe data. The “Semantic Web” can be used to find related articles or a cluster of similar articles automatically. In J.UCS similar articles are clustered in different collections. As described above, the ACM-classification scheme is used for the clustering of documents stored on the same server system. If the Semantic Web becomes reality also documents stored on different systems can be clustered together and this would be a powerful feature for all users.

5.5 Conclusion and Future Work

There is a need for users to bias the layout and the content of a digital library. Such user dependent preferences must be stored at some place. If these preferences are stored on the server-side there are enormous advantages. A workspace for a registered user of the Journal of Universal Computer Science (J.UCS) was introduced and some functions like rearranging articles, public and private bookmarks, and a global history were described. A digital library should not just be read, but a user should work with it. “Links to the future”, special annotations etc. (see section 5.4) are necessary tools to make full use of the power of many users. Reuse of existing material in terms of reference linking instead of cut-and-pasting is a key issue.

The next step is to redesign some existing annotation tools to make rating of content and other described annotation facilities possible and to implement some of the described functions. Articles are already described by some meta-data but these data are in a proprietary format and must be translated to a common format (like Dublin Core) to make the Semantic Web possible.

References

- [BMM, 2000] Brockhaus Multimedia. <http://www.brockhaus-multimedia.de>.
- [MyBookmarks, 2001] MyBookmarks <http://www.mybookmarks.com>.
- [EDN, 2001] Education Network Australia. <http://www.edna.edu.au>.
- [Hyperwave, 2001] Hyperwave Information Server. <http://www.hyperwave.com>.
- [Learning-NY-Times, 2001] Learning network. <http://www.nytimes.com/learning>.
- [PC-Bib, 2001] PC-Bibliothek. <http://www.iicm.edu/pcbibliothek/home>.
- [ACM, 1998] ACM (1998). Computing Classification System. <http://www.acm.org/class>.
- [Berners-Lee et al., 1998] Berners-Lee, T., Fielding, R., Irvine, U., and Masinter, L. (1998). Uniform Resource Identifiers (URI). RFC 2396. <ftp://ftp.isi.edu/in-notes/rfc2396.txt>
- [Clarke, 2001] Clarke, R. (2001). Cookies. <http://www.anu.edu.au/people/Roger.Clarke/II/Cookies.html>.
- [Heinrich and Maurer, 2000] Heinrich, E. and Maurer, H. (2000). Active Documents: Concept, Implementation and Applications. *Journal of Universal Computer Science*, 6(12):1197–1202. http://www.jucs.org/jucs_6_12/active_documents_concept_implementation
- [Krottmaier, 2001] Krottmaier, H. (2001). Improving the Usability of a Digital Library. In Hübler, A., Linde, P., and Smith, J. W., editors,

Electronic Publishing (ELPUB 2001), pages 178–182, Canterbury, Kent, United Kingdom. International Council for Computer Communication (ICCC) and International Federation for Information Processing (IFIP).

[Mason and Sutton, 2000] Mason, J. and Sutton, S. (2000). Education working group: Draft proposal. <http://dublincore.org/documents/education-namespace/>.

[Mason et al., 2000] Mason, J., Sutton, S., and Co-chairs (2000). Education working group: Report of deliberations. http://www.ischool.washington.edu/sasutton/dc-ed/Dc-ac/DC-Education_Report.html.

[Maurer, 1996] Maurer, H. (1996). *Hyper-G now Hyperwave — The Next Generation Web Solution*. Addison-Wesley

[Maurer, 2001] Maurer, H. (2001). Beyond classical digital libraries. In *Proc. NIT conference*.

[Maurer and Schmaranz, 1994] Maurer, H. and Schmaranz, K. (1994). J.UCS - The Next Generation in Electronic Journal Publishing. *J.UCS*, 0(0):117–125. http://www.jucs.org/jucs_0_0/j_ucs_the_next.

[Nelson, 1987] Nelson, T. (1987). *Literary Machines*. Ted Nelson.

[Schmaranz, 1996] Schmaranz, K. (1996). Professional electronic publishing in hyper-g. In *Proceedings of WebNet 1996*. <http://curry.edschool.virginia.edu/aace/conf/webnet/html/130.htm>.

[W3C, 2001] W3C (2001). Semantic web activity. <http://www.w3.org/2001/sw>.

Chapter 6

More than Passive Reading: Interactive Features in Digital Libraries

This chapter was presented by the co-author of the corresponding paper at the “World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education” (E-Learn 2002) in Montreal, Canada.

It continues the work printed in the previous chapter and clarifies the need for interactive features in digital libraries. Once an information resource is found, the user wants to work with it and explore the content. Especially when working with scientific publications interactive features are necessary. The user’s ability to annotate parts of a document is mentioned and problems arising are described. In our research and development system we provide different formats of the content via objects on the server. Special treatment of those objects must be implemented to make annotations visible in all formats

of the information. Other features, such as content rating and even rating of parts of content, are introduced and the technical problems are discussed.

The original contribution was published in the proceedings of the conference and can be found in [Krottmaier and Helic, 2002b].

6.1 Abstract

At the moment it is not very common to actively work with material stored on an Internet server system. Once the right information resource is found, only passive information consumption of documents is provided by a majority of information systems. While this is appropriate for a kind of reading material, this is a limitation for scientific articles. Scientific articles are often read more than once, are usually annotated while they are explored, and are often the root of further investigations. In this paper we explore two features which will be available in the Journal of Universal Computer Science (J.UCS). These features will support the reader in exploring articles and will make it much easier to turn information into knowledge.

6.2 Introduction

Digital libraries are often the source of information. Three common steps are necessary to turn information into knowledge:

1. Find the right information.
2. Explore this information.
3. Apply the information.

Several tools are available to support scientists in finding the right information: search engines (such as Google) and directory services (such as Opendirectory) are often used as “entry to the web”. While search engines are often used by users *who know exactly what they are searching for* and directory services are used by users *who do not know exactly what they are looking for*, there are also some system specific notification mechanisms available, which “directs” users to sources of information. In the context of digital library systems this kind of service is called “Alert Service”. These tools are currently well known and heavily used in the Internet community.

A few highly sophisticated features (such as “find similar information resources”) are already integrated in search engines and are inherently available in directory services . Other specialized application specific features are available in the server systems providing the information. In this paper we are going to discuss two features: an annotation feature and a personalization feature. We illustrate implementation details of these features. Annotations have been available in J.UCS ([J.UCS, 2002]) since the beginning (1994). The personalization feature is currently under development. In J.UCS we use a Hyperwave Information Server (HIS, [Hyperwave, 2001]) to make the implementation of these features as easy as possible. In the following sections we explore some aspects of these features.

6.3 Annotations

It is very common to write handwritten annotations on paper or mark some regions of text as important or unimportant for understanding, especially in personally owned material ([Marshall, 1997]). “Dog-eared” books are *highly customized books* and the turned-down corner may be of great value to the

“creator” while trying to find an interesting part of the book again! No matter what type of content (text or graphic) on whatever position may be annotated in a traditional library environment. It is obvious that it should be possible to annotate also material stored in a digital library environment.

In the electronic environment the software must support the user with a highlighting tool. Web browsers (such as Amaya), available for different operating systems) provide a client based annotation facility, i.e. the client supports the user in generating annotations. Annotations are stored locally or at a server system. The whole document or a fragment of a document may be annotated. If a fragment is annotated, the source region is described via XPointer and can therefore be cascaded with other annotations. Different web browsers are also supported by “bookmarklets” and plug-ins.

Unfortunately, not all digital library systems support the reader in the creation of electronic annotations although electronic annotations are much more powerful than “analog” annotations in a traditional environment. The following list illustrates features of electronic annotations:

Private Annotations: if the digital library system provides the user with an identification facility, it is very easy to support private annotations. Private annotations are visible only to the author of the annotation.

Shared Annotations: if annotations are stored in an intermediate- or server system, it is possible that users share annotations. Therefore many users may explore the content by using annotations very quickly. There are many applications of this kind, especially if users are working in groups and/or are exploring online-available literature.

Typed Annotations: types like “Question”, “Answer”, “Problem”, “Solution”, “Rating” etc. are needful in selecting whether to read or not to

read the annotation or even the content. Additional attributes such as “User who wrote the annotation” may be very helpful.

Visualization of Annotations: Annotations may be visualized differently and the current user of the system must be able to decide how to display them. To give an example: An author of a document which is annotated very often with “Question”-typed annotations may improve the document to make it easier for other users to understand the content stored in the document. On the other hand a student may let the system display “Answer”-typed annotations.

Rated Content: A document may be rated by special annotations. Users may decide to read just articles with top ratings and may not waste their time in reading (obviously) poor written articles. Actual content (i.e. recently published content) must be treated in a special manner.

Nevertheless, some technical problems have to be solved and tools must be created to support users in writing and managing annotations. Let us now discuss some approaches and problems in conjunction with the annotation feature. There are different possibilities for the location where to store such annotations. Annotations may be stored on (1) client-side systems, (2) intermediate systems and (3) server-side systems ([Krottmaier, 2001]).

Annotations in J.UCS are stored on the server-side, therefore they are easily accessible to all users or to a group of users. Annotations in J.UCS are (1) indexed and therefore search able, (2) interlinked via special link-types with the content and (3) physically separated from the content.

We mentioned the term *information source*. This kind of source is a collection of documents which all represents the same content. I.e. if content is available

to users in different formats (e.g. in PDF, PostScript and HTML) we talk about *one* information source but different formats of the content.

This fact introduces another problem: Annotations to the HTML representation of the content should also be presented to users when looking at the PDF-version of the content and vice versa. On the other hand, not every reader of the PDF-version is able to create annotations because the annotation feature is not available in Acrobat Reader. The Adobe Acrobat API-license prohibits development of plug-ins supporting annotations. Therefore another solution without using the PDF-inherent annotation facility must be developed. Currently we explore two approaches to solve this problem: feedback forms for *each page* in the document and hyperlinks from *each section* to an annotation server application.

There are many applications of annotations. One application is the *active documents* concept ([Heinrich and Maurer, 2000]). Active documents introduce a system which may automatically answers questions. Users may ask any question at any position in a document stored on the server system. The system either answers the question automatically (by searching for answers to similar questions), or forwards the question to the author or maintainer of the document. If there are many users of the library and if the tools are easy to use, there is an enormous power in using this idea of active documents! But also other applications are possible: e.g. an *intelligent automatic summary* of a document. This summary can be as easy to implement as collecting and displaying all highlighted document fragments which are created by the readers of the document. Again, many users may increase the quality of these summaries.

There are many studies about how annotations are written in an electronic environment (e.g. [Marshall et al., 2001]). Many web based based environments

(e.g. [Rosenstock and Gertz, 2001], [Kahan et al., 2001]) are using annotations. Annotations are a necessary feature in every system providing information to a large community.

6.4 Rearrange Documents and Document Fragments

Organizing whole content entities or parts of content in a user defined way is essential when improving the usability of digital libraries ([Krottmaier, 2001]). We are currently implementing such a *server-side personal data collection* in the environment of the Journal of Universal Computer Science. This makes it easier for readers *to work with the content* of the library. In the upcoming prototype it will be possible to simply select information sources (in our case articles and article collection) stored on the server and link them to a personal collection. As a result of the integrated link database, the collections will remain up-to-date even if objects are moved to some other location on the server. Therefore the infamous `HTTP-error 404` (“Page not found”) cannot appear when accessing objects stored in the database.

Such a personalized data collection may be static or dynamic. The example above showed a static collection, where each article must be selected and linked to the collection. If there is the request to visualize “all articles by author XXX” or “all articles in category YYY”, this approach is not very effective. Therefore other possibilities must exist: to answer the first question (articles by author), so called *query objects*, i.e. objects, representing a search query in the database, will be used. When these queries are executed, all matching objects will appear in the listing of the personal collection. The second question

(articles by category) can be answered by linking the appropriate category-collection organized by the administrator to the personal collection of the user.

Since we are using an object-oriented database system (Hyperwave Information Server, HIS) which supports inherently users and groups, it is possible to implement *public* and *private* collections of articles simply by assigning the proper “rights”-attribute. Public collections may be used as a kind of *public bookmarks* already implemented by a lot of services (e.g. [MyBookmarks, 2001]). But there are several other advantages, including accessibility, when saving bookmarks on the server-side rather than on the local file system in some browser-specific format.

Organizing objects stored *on* the server system (*local objects*) is obviously not sufficient in the context of an open digital library system. It must also be possible to organize objects stored on any other system (*remote object*) like HTTP-server systems, other internet based systems, and even files stored on the local file system.

To enable structuring of remote objects a surrogate object must be created in the database. This object (in Hyperwave terminology also known as “remote object”) is then used as handle for further operations. Since the control of the referenced part is completely up to the remote server administrator, tools (such as link checkers and content watchers etc.) must be integrated to support the user while working with remote objects.

At least the following problems must be handled by the software system:

Removed Destinations: The referenced part may be deleted at any time *without* notification to the author using that destination as a reference or inclusion. Therefore it is possible that the most common error in the WWW (“404: Page not found”) occurs. According to web usabil-

ity analysis the system should take care of link destination availability, therefore broken links *must not* be displayed to the user. Since there is no notification in ordinary server systems when an object is removed the application must observe link destinations. It is very common that automatically scheduled programs (known as “robots”) are checking the availability of link destinations and perform some actions when an object is removed. In the Journal of Universal Computer Science a note will be attached by the system to the link-region indicating that the link is broken. The link will also be deactivated and the author of the link will be notified by email.

Moved Destinations: Modern publishing systems (like the ACM-DL and Springer-Link provide the users with so called *digital object identifiers* or DOIs. This service is very often used for *stable linkage* to online available information sources and should be preferred when linking to references.

Modified Destinations: A link becomes useless if the content of the destination has changed completely. Therefore it is not enough if the link checking software is looking for broken links, but also for modified destinations. If the destination server supports HTTP/1.0 or later this task can easily be performed by sending a conditional request command (e.g. GET or HEAD) containing a “if-modified-since”-headerline to the server system. Please note that this version of the protocol is available on almost every modern server system.

Structuring a single document as one unit is easy compared to the task of structuring parts of a document. Resources are usually addressed by URLs (Uniform Resource Locators, [Berners-Lee et al., 1994]). Depending on the document format, more granularity is possible.

We should not limit this discussion of fragments of a text based document (such as HTML, XML and PDF). A user may address the upper left part of an image, or the first 5 seconds of a video, etc. There is much work to do, to implement this feature.

6.5 Conclusion and Future Work

In this paper two necessary features were described in the context of the Journal of Universal Computer Science. Problems and issues of the well known annotation feature were presented and ideas on how to personalize documents and document collections were discussed.

In the current implementation it is not possible to add annotations *at any position* in the HTML representation of the information resource. Only whole content objects may be annotated at the time of writing. Although this approach is suitable for small documents, it is inappropriate when working with large documents. Therefore future work is addressed to make it possible to add annotations to parts of documents – simply by selecting a corresponding document fragment. Sharing annotations between different document formats is also an issue in the prototype.

The current working prototype does support rearrangement of locally stored objects, but does not support restructuring of document fragments. The next step is therefore to organize and restructure parts of documents.

References

- [Berners-Lee et al., 1994] Berners-Lee, T., Masinter, L., and McCahill, M. (1994). Uniform Resource Locators. <http://www.w3.org/Addressing/rfc1738.txt>.
- [Heinrich and Maurer, 2000] Heinrich, E. and Maurer, H. (2000). Active Documents: Concept, Implementation and Applications. *Journal of Universal Computer Science*, 6(12):1197–1202. http://www.jucs.org/jucs_6_12/active_documents_concept_implementation.
- [Hyperwave, 2001] Hyperwave (2001). Hyperwave Information Server. <http://www.hyperwave.com>.
- [J.UCS, 2002] J.UCS (2002). Journal of Universal Computer Science. <http://www.jucs.org>.
- [Kahan et al., 2001] Kahan, J., Koivunen, M.-R., Prud’Hommeaux, E., and Swick, R. R. (2001). Annotea: An open rdf infrastructure for shared web annotations. In *Proc. of the WWW10 International Conference*.

- [Krottmaier, 2001] Krottmaier, H. (2001). Improving the Usability of a Digital Library. In Hübler, A., Linde, P., and Smith, J. W., editors, *Electronic Publishing*, pages 178–182, Canterbury, Kent, United Kingdom. International Council for Computer Communication (ICCC) and International Federation for Information Processing (IFIP).
- [Krottmaier and Helic, 2002b] Krottmaier, H. and Helic, D. (2002b). More than Passive Reading: Interactive Features in Digital Libraries. In *Proceedings of E-Learn (E-Learn 2002)*, pages 1734–1737, Montreal, Canada.
- [Marshall, 1997] Marshall, C. C. (1997). Annotation: From paper books to digital library. In *ACM DL*, pages 131–140.
- [Marshall et al., 2001] Marshall, C. C., Price, M. N., Golovchinsky, G., and Schilit, B. N. (2001). Designing e-books for legal research. In *Proceedings of the first ACM/IEEE-CS joint conference on Digital libraries*, pages 41 – 48. ACM Press.
- [MyBookmarks, 2001] MyBookmarks (2001). <http://www.mybookmarks.com>.
- [Rosenstock and Gertz, 2001] Rosenstock, B. and Gertz, M. (2001). Web-based scholarship: annotating the digital library. In *Proceedings of the first ACM/IEEE-CS joint conference on Digital libraries*, pages 104 – 105. ACM Press.

Chapter 7

Automatic References: Active Support for Scientists in Digital Libraries

This chapter was accepted for presentation at the “5th International Conference on Asian Digital Libraries: Digital Libraries: People, Knowledge & Technology” (ICADL 2002) in Singapore.

It is very difficult, especially for young and novice scientists, to find the right references when exploring a new topic. This short paper gives an overview of a system for developing and assembling a list of references when an outline of an article is given. This unique system will be implemented in the Journal of Universal Computer Science. At the time of writing, no other digital library system supports readers actively in this aspect.

The original contribution can be found in [Krottmaier, 2002a].

7.1 Abstract

Scientists need automatic support from digital library systems when writing papers about a new topic. In this paper we introduce a concept to be integrated in the Journal of Universal Computer Science (J.UCS), where scientists will be able to upload an abstract or outline of a paper, and the system will return a list of “must-read” papers related to the topic introduced in the uploaded document.

7.2 Introduction, Problems and a Possible Solution

Traditional libraries have a few very sophisticated features still not available in most digital libraries. While some problems of traditional libraries are solved in many digital library systems, such as searching in fulltext or personalization of the material stored in those systems, the most sophisticated “feature” in traditional libraries is still not available: the librarian!

The role of librarians is often underestimated by most of the users. Especially for young scientists, who are trying to explore a new topic, librarians are very helpful. They are able to answer fuzzy questions or help with formulating “search queries” by asking related questions to the scientists. This interactive discussion is very helpful for a scientist when trying to find the right “starting-point” for a new work, i.e. to form the right list of references. Knowledge management tools are widely available and must be integrated in existing digital library environments.

In the following we list tools to find the right references. Please note the big problem with common search engines: they are just able to index freely available resources. However, high quality publications are often available to *subscribers* of the corresponding journals. Therefore many high quality publications are not accessible via search engines.

Search engine usage is an every day task. It is possible to formulate a search query, but it is very difficult to formulate the *right* query. There are very often thousands of results or zero. It is well known that search engines are very powerful if someone knows *exactly what to search* but this is not true in our application. We want to find the right references where the topic is not clearly stated.

GoogleScout is a very powerful tool implemented in the google search engine ([Google, 2002]). GoogleScout is an implementation of the “search for similar pages” feature without having to worry about selecting the right keywords.

Related conference proceedings and SIGs are very often the right starting point of researching a topic and finding the “big players” in the topic. These proceedings are very often also available in electronic form and are therefore easily searchable. Nevertheless, current publishing systems provide the user with a feature like “find related articles to an *already published* article” but not “find related articles to some *arbitrary abstract or document outline*”!

Digital libraries like ACM/DL or IEEE/Xplore do support registered user with a “find related/similar articles”-feature. They consider already published articles but no arbitrary, user-defined abstracts or outlines.

ResearchIndex also supports the scientists in finding the right reference list with rated resources ([Lawrence et al., 1999]).

We are going to prepare an uploading area in the Journal of Universal Computer Science ([J.UCS, 2002]). Scientists will be able to upload an article abstract or outline in different electronic formats (such as PDF, PostScript, MS-Word, HTML, XML, etc.). The system will find related articles using similarity matching algorithms implemented in the used search engine.

At the moment there are approximately 600 peer-reviewed articles available in the system. These articles were published during the last 8 years. Because of strict peer-reviewing in J.UCS the quality of the articles is very high compared to freely available systems. We are convinced that scientists working in the area of computer science will find this unique feature valuable and other digital library systems will implement this feature in their systems.

7.3 Conclusion and Future Work

An every day problem of information exploration for scientists is to find related or similar documents to some given article outline. Many digital library systems provide registered users with similarity search to already published articles. Currently it is not possible in well known publishing systems to *find related articles to an arbitrary article*. We suggest to extend currently available systems with this feature and are going to implement a prototype in the Journal of Universal Computer Science (J.UCS).

References

- [Google, 2002] Google (2002). <http://www.google.com>.
- [J.UCS, 2002] J.UCS (2002). Journal of Universal Computer Science. <http://www.jucs.org>.
- [Krottmaier, 2002a] Krottmaier, H. (2002a). Automatic References: Active Support for Scientists in Digital Libraries. In Lim, E.-P., Foo, S., and Khoo, C., editors, *Digital Libraries: People, Knowledge & Technology: Proceedings of the 5th International Conference on Asian Digital Libraries (ICADL 2002)*. Springer.
- [Lawrence et al., 1999] Lawrence, S., Giles, C. L., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71.

Chapter 8

Enhanced Annotations

The following chapter was accepted for presentation at “14th International Conference on Society for Information Technology and Teacher Education” (SITE 2003) in New Mexico, USA.

Annotations are well known in the Internet community. However, users do not recognize the power of annotations. Especially when many users work with this feature, new and powerful features may be implemented. In the following sections some of these features are described.

The original contribution can be found in [Krottmaier, 2003].

8.1 Abstract

Usage of annotation features is common and implemented in many information systems such as digital library systems and learning environments. The Journal of Universal Computer Science (running since 1994) was one of the first digital library systems, which offered this interactive feature to the users.

Unfortunately, features already available in modern knowledge management (KM) systems are not integrated in most of these common annotation facilities. KM-features will support users in the creation of annotations and they would make a much more structured discussion about the content of an article possible. In this paper we are going to show why the integration of a very common KM feature (similarity search) is necessary. It will be shown that this simple additional feature will enhance the process of annotating scientific papers dramatically. Learning environments are also an important application for annotations. Learning material will be enhanced automatically by users.

8.2 Introduction

Annotations are usually small text documents attached to a published article. Ideally an annotation is attached to a fragment of an article. Thereby additional information about this specific part of the content is added. An annotation is a kind of *electronic post-its* and may start a discussion about the annotated part.

In a traditional environment with printed material annotations are also very common. Personal owned books, journals and especially learning material is “personalized” with markers, pens and bookmarks ([Marshall, 1997]).

There are much more advantages in the electronic environment than in the traditional one because annotations may be shared. Authors may get additional input from readers of their contributions. If the annotations are typed, i.e. marked with additional attributes such as *question* or *answer*, the author, but also other readers, may benefit from this feature.

Let us now explore some situations where authors as well as readers benefit from annotations attached to an article fragment. Let us assume the following

types of annotations: *comment*, *question*, *answer*, *problem*, *solution* and *advice* as implemented in the Journal of Universal Computer Science ([J.UCS, 2002]) and some other annotation-aware systems like Annotea ([Kahan et al., 2001]).

8.3 Motivation

An article which has been published is often immutable (“dead documents”, [Nürnberg, 2002]). As in many professional electronic publishing systems (especially in the scientific community) this annotation feature is the only way how an author may clarify ideas presented in a previously published article if questions from authors arise.

In a traditional – exclusively printed – environment there is no feature like this available! “Letters to the editor” may be used to clarify some previously published ideas. However, it is not very likely, that *all* users who read the original contribution will also read this specific letter to the editor.

Many publishers offer electronic discussion forums or news groups, where readers are able to discuss a publication. Unfortunately, this discussion is very often separated from the electronic content. There is no hint when reading the content that there is a discussion about the document or a specific part of the document.

On the one hand many annotations of type *question*, *problem* or *advice* to a specific document may indicate that the document is not well-written. This fact should influence the author when writing upcoming articles and should improve the style of future papers. On the other hand readers, who are going to work with an article with such kinds of annotations, would probably decide not to read the article first but start with the questions and problems.

The style of the article and the open questions obviously not answered in the article indicate an inadequate quality of the article. This implicit fine-grade rating of articles is much more sophisticated than the common number-based rating system of articles available in many systems.

The visualization process of an annotated article must be very flexible. In addition to private annotations it should also be possible to switch annotations on and off, list articles not just in alphabetic order but also by the implicit ranking of the articles.

8.4 Improvements

Tools for creating annotations must be easy to use. No additional software should be installed on the client side. Therefore it is quite difficult to annotate arbitrary document types. To give an example: PDF is very often displayed with Adobe Acrobat Reader. Unfortunately, it is not possible in this software to annotate PDF-documents. Although Adobe announced that upcoming versions of the software will support this feature, it will take time till the new version is widely available on every platform. Some ideas on how to overcome this problems are discussed in [Krottmaier and Helic, 2002].

In currently available systems it is possible to write arbitrary comments to articles. No quality-control is implemented in many systems, therefore any kind of annotation – even violations of the “netiquette” – may be published. Technically it is possible to check an annotation against a list of “black words” before it is published. If there is a match, an editor should review the annotation. User management is therefore a requirement for a system supporting annotations.

Before uploading an annotation, the system must check if there is a similar annotation available in the system. Highly sophisticated similarity matching algorithms are already available for many platforms and should be used to support the user in creating annotations. A similar discussion available in the system or in some other part of the document or even in another document should be taken into account before this new annotation is published.

Beside this “simple quality control” many other mechanisms must be implemented to guarantee quality in the system. Users submitting annotations should be rated by other users. This mechanism is well known in discussion forums and it should be available in every annotation aware system, since annotations are a kind of discussion about a document or about document fragments. Highly rated users should be honored by the system.

Many publishing systems or learning environments provide content in different electronic formats. In the Journal of Universal Computer Science content is offered in PostScript, PDF and HTML. Since the contents are equivalent, a discussion in the PDF document must be synchronized with a discussion in the HTML formatted content. Therefore the annotation feature should be independent of the storage location of the documents and must reflect to the *contents* stored in these documents.

Annotations should also be “active” to the participants of a discussion. This may be implemented by sending emails to the authors of an article or readers of the article who subscribed to upcoming discussions or similar discussions. Ideally a document will answer questions automatically as described in [Heinrich and Maurer, 2000]. This notification mechanism must be highly configurable, i.e. users should be able to stop notifications and review their user-profile in a regular manner.

8.5 Conclusions and Future Work

Electronic annotations are a powerful feature. Many implementations are already available. If many users work with this interactive feature the published material will automatically be improved over time. Therefore every involved participant – every author and every reader – will benefit from this annotation feature. This paper listed some additional necessary features to make annotations more powerful and to allow high-quality, peer-reviewed discussions about documents and document fragments.

An annotation feature is already implemented in J.UCS. At the moment it is possible to attach typed annotations (such as “Question”, “Answer” etc.) to content stored on the system. Authors of annotations may be informed by email (on request) of further discussions about this document. Currently we are implementing more granularity when attaching annotations to a document. Therefore it will be possible to annotate paragraphs, sentences and even words of documents. Quality control as explained in the previous section will be available.

References

- [Heinrich and Maurer, 2000] Heinrich, E. and Maurer, H. (2000). Active Documents: Concept, Implementation and Applications. *Journal of Universal Computer Science*, 6(12):1197–1202. http://www.jucs.org/jucs_6_12/active_documents_concept_implementation.
- [J.UCS, 2002] J.UCS (2002). Journal of Universal Computer Science. <http://www.jucs.org>.
- [Kahan et al., 2001] Kahan, J., Koivunen, M.-R., Prud’Hommeaux, E., and Swick, R. R. (2001). Annotea: An open rdf infrastructure for shared web annotations. In *Proceedings of the WWW10 International Conference*.
- [Krottmaier and Helic, 2002] Krottmaier, H. and Helic, D. (2002). More than Passive Reading: Interactive Features in Digital Libraries. In *Proceedings of E-Learn*, Montreal, Canada.
- [Krottmaier, 2003] Krottmaier, H. (2003). Enhanced Annotations. In *Proceedings of International Conference on Society for Information Technology and Teacher Education (SITE 2003)*.
- [Marshall, 1997] Marshall, C. C. (1997). Annotation: From paper books to digital library. In *ACM DL*, pages 131–140.
- [Nürnberg, 2002] Nürnberg, P.J. and Hicks, D.L. (2002). Models for Publishing Academic Hypertexts In *Proceedings of elpub2002*, Karlovy Vary, Czech Republic.

Chapter 9

Automatic Support in the Review Process

This chapter was presented at the “Web Content Mapping: A Challenge to ICT”-workshop held at the “The First EurAsian Conference on Advances in Information and Communication Technology” (EurAsia 2002) in Shiraz, Iran.

The following chapter gives an insight into the refereeing procedure used in the Journal of Universal Computer Science and points out the problems of common refereeing procedures used in many peer-reviewed journals. A completely new method of selecting potential referees is illustrated. This procedure will guarantee constant quality of publications in J.UCS, but can also be applied to other domains such as grant proposal reviews. At the time of writing an implementation is under development, but it is very likely that the introduced procedure is an ideal solution to the problem of expert finding and referee selection.

The original contribution was published in the workshop proceedings and can be found in [Krottmaier, 2002b].

9.1 Abstract

One of the most complicated problems of peer-reviewed journals, regarding the quality of publications, is the right selection of referees for submitted papers. In computer science related journals referees express their interest and main research areas in terms of the ACM classification system. Interests change over time and therefore the referee has to update the “table of interest” from time to time. This paper will give an overview of the current refereeing situation in the Journal of Universal Computer Science (J.UCS) where referees are able to select papers for refereeing from a broad range of topics. In the future, we want to narrow the range of potential papers per referee. We investigate a new approach for the selection of appropriate referees assuming that referees’ recently published papers express their current main interests.

9.2 Introduction

The reputation of peer-reviewed journals is very much dependent on the reputation of the referees of the editorial board. Referees are usually invited by the editor(s)-in-chief to join the editorial board of the journal and then usually sign-up for a certain category of papers. In computer science these categories are often expressed using the ACM Computing Classification System (ACM-CCS, [ACM, 1998]).

Submitted papers should be reviewed by the most relevant referees to guarantee constant quality of published papers. In this paper we are going to illustrate the current situation related to the communication between the referees and the editorial team in the Journal of Universal Computer Science ([Calude et al., 1994], [Maurer and Schmaranz, 1994]). J.UCS is a peer reviewed journal

and was founded in 1994 by Springer Pub. Co. ([Springer Pub. Co., 2002]) in cooperation with the Institute for Information Processing and Computer Supported new Media ([IICM, 2002]), Graz University of Technology in 1994. As of 2001 J.UCS is a publication of the Know-Center ([Know-Center, 2002]) in cooperation with Springer Co. Pub., JOANNEUM RESEARCH ([JOANNEUM RESEARCH, 2002]) and the IICM.

We introduce ideas to improve the communication and reduce the workload for both referees and the editorial team. Thereafter we show some further applications of the ideas presented.

9.3 Current Situation and Improvements

Authors submit their papers to J.UCS via email or via web-server application in the appropriate electronic format. The editorial team then extracts the abstract out of the documents and sends the abstracts to the referees by email. If at least 3 referees sign-up for the paper, the paper is accepted for refereeing, otherwise the submission is withdrawn and the authors are free to submit their papers to other journals.

There are several drawbacks in this procedure and much effort in communication is spent by the editorial team and the referees as well. The editorial team has to extract the abstracts and send emails to the referees on a regular basis. Referees must sign-up for reviews based on abstracts. Then the editorial team must collect the reviews of the papers and inform the authors about the reviews.

It is obvious that a web-application may solve some of the problems concerning this email-based communication. A simple first approach, currently under development, is to sign-up specific referees for particular topics described in

terms of the ACM-CCS. The application then decides on the basis of the author-provided categories or keywords to which referees the abstracts or a notification should be sent. Although the authors are requested to state the appropriate keywords and categories, many authors do not supply this information. A simple submission form with mandatory fields may solve this problem. Many conference supporting systems work like the approach described above.

In the current submission and refereeing procedure we have to deal with two problems and have to solve them with modern knowledge management features:

Submitted Papers do not follow formatting guidelines and do not provide the right information.

Referees do not update their area of interest and will therefore receive unwanted emails.

In the following sections we are going to illustrate solutions to these two problems.

9.4 Improvements in the Paper Submission Process

As described above papers are submitted to J.UCS by either email or via web-form (see figure 9.1). At the moment communication with the corresponding author is done exclusively via email with the editorial team. To reduce the communication workload for the editorial team we are going to assign a username and password at a first submission of an author's paper. Thereafter the author will be automatically informed about *any* change in the status of the

submitted article, i.e. the author knows exactly when the review process starts and when to expect results.

Figure 9.1: Current Paper Submission

Due to the anonymous review of the papers, authors do not know which referees are going to review or have reviewed their papers. The review form is partitioned in author-visible and editor-visible parts. A highly sophisticated access system with integrated user- and group-management is responsible for displaying the appropriate parts of the review to the author.

Some meta data (especially information about the author, such as firstname, surname, institution etc.) is collected by the form of the paper upload. The following data is collected in the web-form: (1) title of the paper, (2) the file (either PDF or PostScript) itself, (3) information about the author like first/last name, institution, country and email. Additional meta data is required for the distribution to the referees: (1) abstract, (2) keywords and (3) categories.

Automatic extraction of the abstract is quite easy since authors usually do provide an abstract: It starts with the word “Abstract” and ends either by a

paragraph starting with “Key Words” or “Category”, or the number and name of the first section.

Unfortunately, authors often submit papers with no keyword- and no category-field. A simple solution to this problem would be to let the author select the keywords and/or categories by hand from a given list (or via a simple text field). This would be easily possible because articles are categorized in terms of the ACM-CCS. For the current prototype it is planned to provide the author with these additional mandatory fields. After a short evaluation with some selected authors the prototype will be available to all authors. Many other publishing systems and conference supporting systems require the same information from authors uploading a paper. This additional workload is delegated to the authors, therefore the workload for the editorial team is reduced without losing quality in the whole workflow.

After the successful upload of a paper, appropriate tools check the given abstract, keywords and categories with the automatically extracted entities. In Hyperwave (HIS, [Maurer, 1996]) some knowledge management tools like Autonomy’s automatic summary are already integrated and available to the developer of a web-application. Automatic clustering is also available in the publishing system used. Thereby it is also possible to automatically categorize the article submitted in an ACM-CCS-style collection hierarchy.

Automatically extracted parts of the document in combination with the given parts of the document (such as abstract, keywords and categories) create a helpful profile of the article used as base for the review process.

9.5 Search for the Right Referee

After some helpful information is extracted from the submitted papers the most appropriate referees must be found. At the moment each referee must decide on the basis of title, abstract, keywords and category of the paper which paper to review and which not to review. While this process guarantees that a “self-assigned referee” is most appropriate for this topic, it also means that every referee receives every submitted abstract by email. This fact increases the workload for the referees. A preselection of potential referees may improve the reviewing process. In the following we illustrate the current available information of referees. Thereafter we sketch how to improve the dissemination of submitted papers more effectively.

9.5.1 Currently Available Information

We have already stored some information about a referees in our database system. Such an entry is shown in figure 9.2. Referees express their interests by simply registering a category according to the ACM-CCS and usually also mentioning some keywords of their current main research interests. Links to their personal homepage are provided on this information page. With the currently available information it would be possible to distribute the abstracts of new submissions to the corresponding most related referees. This procedure will reduce the workload of the referees in the beginning, but there is an enormous drawback to this approach: later on, referees do not update their information page and do not provide us with information about changes in the main research area! Therefore the situation will be very unsatisfactory in a few years when referees are asked to review papers related to *old* research

interests. The quality of publications will decrease over time because there are massive changes in computer science related topics.

The screenshot shows the J.UCS (Journal of Universal Computer Science) website. The browser address bar displays http://www.jucs.org/jucs_editorial_board/calude.c. The page features a navigation bar with links for Search, Subscription, Submission Procedure, and Login. A sidebar on the left lists various categories like Special Issues, Articles by Category, and Board of Editors. The main content area displays the profile of Cristian S. Calude, including a photo, his name, and detailed contact information.

Cristian S. Calude

Referee for: [E.4](#), [F.1](#), [F.2](#), [G.3](#), [H.1.1](#)

Institution: [University of Auckland](#)

Address: [Department of Computer Science](#)
University of Auckland
Private Bag 92019
Auckland
New Zealand

Phone: +64-9-373-7599, +64-9-373-5751

Fax: +64-9-373-7453

E-mail: cristian@cs.auckland.ac.nz
cristian@ibm.net

Home Page: <http://www.cs.auckland.ac.nz/~cristian>

Curriculum vitae:
Cris Calude was born in Galatz, Romania. Educated at Bucharest University (BA(Hons) in Mathematics and Computer Science, 1975; Ph. D. in Computer Science, 1977). Positions: Full Professor (with a Personal Chair in Computer Science), and Director of the Centre for Discrete Mathematics and Theoretical Computer Science, The University of Auckland, New Zealand. Full Professor of Mathematics and Computer Science, Bucharest University, Romania.

Cris Calude wrote more than 130 scientific articles and 14 books (including "Theories of Computational Complexity", North-Holland, 1988, and "Information and Randomness", Springer-Verlag, 1994). He is an editor of Springer DMTCS series.

Main research interests:

- * algorithmic information theory
- * computational complexity
- * recursion theory
- * automata-theoretic models in theoretical physics
- * discrete and constructive mathematics

Figure 9.2: Information about a Referee

Obviously there are two solutions to this problem: first, referees are actively asked to update their database record once a year or whenever there is a change in the research area, second, the publishing system tries to explore necessary information automatically without any additional human interaction. Unfortunately, people will not update database records (even on request!) therefore we decided to explore and implement the automatic exploration approach.

On the assumption that publications of the referees express their main research areas we have based our ideas on improving the proper dissemination of

abstracts. After the publications are found, the system performs a similarity search of submitted articles and recent publications of the referees. We think that we are able to assure the quality of publication into a long term without any additional human interaction!

9.5.2 Search for Publications of Referees

In the following some possible solutions to the problem of finding the publications of our referees are discussed. Some of the freely available services were explored. To enable similarity search the goal is to find at least the abstract of papers written by a particular referee.

Several available search-engines index a lot of material stored all over the world. The most powerful search-engine at the time of writing is the Google search engine with more than $2 * 10^9$ -indexed web pages. An application program interface (API) for Google using SOAP (Simple Object Access Protocol) is available for different programming languages. The problem of formulating the right search query still exists and there are some problems in extracting the right entities from the result. Therefore Google is unfortunately not suitable for our task in searching the publications of referees.

Many referees do supply us with the address of a personal homepage where a list of publications is very often provided. Often authors are allowed to make at least the abstract available at their home page after signing a copyright transfer statement with the publisher of the paper. Older papers (usually papers older than one year) are often available in fulltext on the homepage. Nevertheless, homepages differ in many respects (structure as well as format) and an automatic extraction of the publications of an author is not easily possible. There are still some referees without a homepage or without a list of

publication page linked from the homepage. It is conspicuous that information extraction from homepages is difficult and unmanageable in a long perspective.

Unstructured data is difficult to put into the right context therefore we were looking for structured, freely available material about publications. For computer science related material a service located at University of Trier ([Ley, 2002]) is very appropriate. The data is stored in an XML formatted file and is accessible via a high sophisticated search interface. Hyperlinks to online available electronic resources are provided. It is easy to search using first and last name of an author and get back a list of known publications of this author including some meta data (year, type, title etc.). We think that this service is a good start point for further explorations of publications of an author.

Another well-known service for the search of publications is NEC Research Institute's ResearchIndex ([NEC Research Institute, 2002]). Additionally to information offered by the system from University of Trier, ResearchIndex offers a citation index and caches also the publication itself. Since we are going to explore this system also in another project, we decided to use this system as a source of publication-gathering from our referees as well.

With these two systems we are able to gather at least some of the publications from our referees and are able to extract either the fulltext or the abstract of the publication. We are going to write adapters to make this information available to our system. Additionally we are going to encourage our referees to supply the system with abstracts of their recently published papers.

At the moment we are developing tools for gathering information about our referees concerning publications and are also developing a web-application to enable upload of abstracts and additional information about publications of our referees. We use web-data-extraction tools (a very nice overview of some of these tools is given in [Laender et al., 2002]) as well as simple text extrac-

tion tools to extract information out of web resources. Each publication is represented in the Hyperwave Information Server (HIS) as an entity consisting of a collection holding all the necessary meta data (title, year, keywords etc.) and the abstract of the publication. This collection structure makes it possible to add future developments like fulltext of the publication or citations of the entity. We are aware of many problems (e.g. author's name representation differ in format: one system supply the full information i.e. first-, middle- and lastname, other systems just supply the first letter of the firstname and the lastname...) already solved in the mentioned systems. The implementation of the prototype and the upcoming feedback and evaluation procedure will demonstrate if the described approach for gathering information about publications is applicable.

9.5.3 Similarity Search

Once the publications of the referees are stored in our database, searching for similar articles is just "a mouse click away" and very easy to handle in the used publishing server system. We use a highly sophisticated knowledge management systems (Hyperwave Information Server, HIS) where features such as "find similar articles" or "find a category for an article" are already integrated. Two systems are available at the moment: [Verity, 2002] and [Autonomy, 2002]. These systems support the programmer with slightly different parameters and implementations. Nevertheless, getting a similar document is very easy to implement.

Since similarity search may also be scoped (i.e. the search scope may be restricted to a specific part of the server) there is an enormous potential for this function. It is possible to limit the search scope to all articles published last year. If no appropriate referees are found then the search scope may be

extended to a larger time-scope. If there is a lack of potential referees for a type of topic then the editorial board must be extended etc.

Search results may be sorted in different ways to guarantee quality in the review of articles. To give some examples: if the referee is the only author of the article *and* the article was written just a few months ago, then this referee is very suitable to review this new article. If, on the other hand, the referee is one of several co-authors and the article was published a long time ago, then the referee may be not the appropriate person to review the selected topic. As all reviewing activities of the referees are logged in the system, the activity of the referee may also be considered when selecting the referee for reviewing an article. Articles published in journals may be endowed with a higher priority than those published in conference proceedings. Further parameters including length of the article, additional ratings of other users of the article, relevance to the community represented by a citation index parameter etc. may be taken into account.

It has been shown that there are many parameters to adapt to find the right referee for a submitted paper. After an evaluation period of the system feedback from the referees will be evaluated to improve the performance of the system.

9.6 Further Applications

Many other applications are possible with the shown approach of information gathering and searching for similar documents. In every situation where experts about a topic must be found, this approach may be a good choice to improve the selection of experts.

Conference management systems (e.g. ConfMan, [Halvorsen et al., 1999], used at EurAsia 2002) support conference organizers in the reviewing process of submissions. Submitted papers are uploaded to the server system and meta data is added by the authors via web form. Additional information about a submission (like keywords) are added by the authors themselves. Members of the program committee sign-up for a review and return the results via email or web-form. These systems may be extended by the described ideas: automatic keyword extraction from the submitted papers will guarantee appropriate keywords, the most appropriate referee of a submission may be found by simply do a similarity search between the content of the submission and the abstracts or fulltexts of articles written by the referees. The number of referees per submission may be reduced and therefore time is saved without losing quality of reviews.

Refereeing of grant proposals may be improved by automatically selecting the appropriate referee for the proposal. Whenever an expert or expert group of a number of given persons must be found the described approach may improve the current situation. This idea can be extended to introduce a support system for writing scientific papers ([Krottmaier, 2002a]). Experts about a topic are found automatically and the system may generate an automatic reference section for publications in preparation. It is and will not be possible to explore and gather information about all experts. Nevertheless, it is planned that at least a selection of experts who published at least one paper in a well-known, reputable journal are considered to be included in the data base.

9.7 Conclusion and Future Work

Recent publications of authors express their current research interests in much greater details than information provided by the authors like unstructured formatted lists of keywords and/or ACM categories. Interests change over time and many experts do not update information given on their homepage.

It has been shown that there are several applications where experts must be found. An application is the assignment of a number of referees to submitted papers of a journal. Constant quality of publications and a minimum amount of time for the referees and editor(s)-in-chief are the main advantages of the described system. Several problems when finding publications of an author are already solved in other systems ([Lawrence et al., 1999] or [Ley, 2002]) therefore the system must not deal with these problems.

Future work is addressed to implement some of the ideas presented in this paper. Gathering of information about a referee will be implemented using the two mentioned systems. Contact to the operator of one system is already established. Other technologies of gathering information (like intelligent agent technology) are currently explored and compared to the described approaches. If publications are found, information must be extracted and the appropriate data must be inserted into the database system. Usability studies and feedback will guarantee a fast and easy to use application.

References

- [ACM, 1998] ACM (1998). ACM Computing Classification System.
- [Autonomy, 2002] Autonomy (2002). <http://www.autonomy.com>.
- [Calude et al., 1994] Calude, C., Maurer, H., and Salomaa, A. (1994). Journal of Universal Computer Science. *Journal of Universal Computer Science*, 0(0):109–115. http://www.jucs.org/jucs_0_0/journal_of_universal_computer.
- [Halvorsen et al., 1999] Halvorsen, P., Lund, K., Goebel, V., Plagemann, T., Preuss, T., and Koenig, H. (1999). ConfMan: Integrated WWW and DBS Support for Conference Organization.
- [IICM, 2002] IICM (2002). <http://www.iicm.edu>.
- [JOANNEUM RESEARCH, 2002] JOANNEUM RESEARCH (2002). <http://www.joanneum.ac.at>.
- [Know-Center, 2002] Know-Center (2002). <http://www.know-center.at>.
- [Krottmaier, 2002a] Krottmaier H. (2002a) Automatic References: Active Support for Scientists in Digital Libraries. In Ee-Peng Lim, Schubert Foo, and Chris Khoo, editors, *Digital Libraries: People, Knowledge & Technology: Proceedings of the 5th International Conference on Asian Digital Libraries (ICADL 2002)*. Springer.

- [Krottmaier, 2002b] Krottmaier, H. (2002b). Automatic Support in the Review Process. In Far, B. H., Shafazand, H., Takizawa, M., and Wagner, R., editors, *Proceedings of the Workshops of "The First Eurasian Conference on Advances in Information and Communication Technology" (EurAsia-ICT 2002)*, pages 467–471. Österreichische Computer Gesellschaft.
- [Laender et al., 2002] Laender, A., Ribeiro-Neto, B., da Silva, A., and Teixeira, J. (2002). A brief survey of web data extraction tools. *Sigmod Record*, 31(2).
- [Lawrence et al., 1999] Lawrence, S., Giles, C. L., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71.
- [Ley, 2002] Ley, M. (2002). Computer science bibliography. <http://dblp.uni-trier.de>.
- [Maurer, 1996] Maurer, H. (1996). Hyper-G now Hyperwave — The Next Generation Web Solution.
- [Maurer, 2001] Maurer, H. (2001). Beyond classical digital libraries. In Chih Chen, C., editor, *Global Digital Library Development in the New Millennium*, pages 165–173.
- [Maurer and Schmaranz, 1994] Maurer, H. and Schmaranz, K. (1994). J.UCS - The Next Generation in Electronic Journal Publishing. *Journal of Universal Computer Science*, 0(0):117–125. http://www.jucs.org/jucs_0_0/j_ucs_the_next.
- [NEC Research Institute, 2002] NEC Research Institute (2002). Researchindex. <http://www.researchindex.com>.

[Springer Pub. Co., 2002] Springer Pub. Co. (2002). <http://www.springer.de>.

[Verity, 2002] Verity (2002). <http://www.verity.com>.

Acknowledgment

I like to thank Prof. Hermann Maurer for the fruitful discussions and Dana Kaiser for giving me insights to the work of the editorial team.

Part IV

**Further Applications of Digital
Libraries**

“There are no such things as applied sciences, only applications of science.” *Louis Pasteur*

Chapter 10

Managing and Storing Digital Audio Data in Intranet and Extranet

The following chapter is an abridged version of a paper presented at the “20th International Convention on Sound Design” (1998) in Karlsruhe, Germany. A brief update to current technologies is given in this preface.

In the following sections the usage of a Hyperwave Information Server as server for digital audio data is described. Attributes of audio objects are stored and managed within the database. Since these attributes are indexed, audio data are searchable via a simple web-interface. In 1998 it was uncommon to separate attributes and content because many systems did not support management of metadata. Most applications store unstructured, proprietary types of metadata in HTML-pages (using the `META`-tag).

Nowadays different metadata-standards are available. The “Dublin Core Metadata Standard” is well known in the developer community and most of Web-

based applications use this type for storing metadata. The Dublin Core Metadata Initiative (DCMI) was founded in the early years of the WWW, at the 2nd International World Wide Web Conference in Chicago, October 1994. After several discussions about the future of HTML, Web authoring tools, etc. the discussion group decided to organize a workshop in Dublin, Ohio, in March 1995. At this event more than 50 people discussed how a core set of semantics for Web-based resources should look like. As a result, the “Dublin Core metadata” (DC) was defined. In DC it is possible to refine and qualify entries in the metadata set. Therefore a broad spectrum of applications may be covered by DC. A brief discussion about usage of metadata in digital libraries was already given in section 2.3.2. However, several application-specific standards based on Dublin Core were derived in the last few years.

Another popular standard for storing metadata of multimedia content is the Multimedia Content Description Interface (MPEG-7). [MPEG-7, 1999] summarizes the objectives and goals of MPEG-7:

MPEG-7 will specify a standard set of Descriptors that can be used to describe various types of multimedia information. MPEG-7 will also specify pre-defined structures of Descriptors and their relationships, as well as ways to define one’s own structures.

This ISO standard combines metadata related to content (such as title, creator, etc.) and technical-metadata about the resource (such as size, encoding, etc.) so that users can search, browse, and retrieve that content more efficiently and effectively.

At the time of writing, several document formats for audio data were established. Different operating systems used different default digital formats for audio data. On the one hand AIFF (Audio Interchange File Format) was well

known in the Macintosh-world, on the other hand WAV (Waveform Audio File Format) was established in MS-Windows based personal-computers. Although MP3 (MPEG-1 Audio Layer 3) was already developed in 1992 at Fraunhofer-Institute in Erlangen, Germany, it was not as popular as the two mentioned audio-formats. Nevertheless, in 1999 a very popular music-exchange services (Napster) was established. Napster used MP3 as exchange format and usage and acceptance of MP3 increased. Advantages such as very low bandwidth requirements for MP3-coded data (approximately 1 MByte/min.) aided fast dissemination of MP3. Today, MP3 is the most popular format for storing digital audio data.

Developments in network technologies and fast Internet connections allow users to exchange audio data quickly. Many users are nowadays connected via high-speed lines to Internet Service Providers (ISP). However, the situation was different in 1998: most of the users used dial-up connections. Size of audio-files and therefore transmission time of those files was a serious issue.

The Hyperwave Information System is an object-oriented database system with sophisticated metadata management. Therefore no additional implementation effort was necessary to enable storage of audio data and management of related metadata. This paper shows the broad field of digital libraries: Any kind of data may be stored inside such a library.

The original contribution was published in the proceedings of the conference and can be found in [Gütl et al., 1998].

10.1 Abstract

Internet, and in particular WWW, is heavily used by people. Over the last few years more and more multimedia components like sound applications have been

published or embedded in Web pages. Digital audio technique combined with internet services provide a wide range of applications, e.g. real audio streaming, support for blind persons, independent music sales etc. The large amount of digital audio files and the need for managing, storing and re-using them have led to the work presented. At the IICM many research activities have been done in the field of multimedia information management and electronic publishing over the last two decades. The work introduced here describes a future oriented digital audio database for intranet and extranet systems based on the Hyperwave Information Server, originally developed at IICM. The database system implemented allows to hold any sound information in arbitrary file formats. Also, additional information like multimedia objects and Meta Information can be stored and managed by the system. A further advantage is that inserting, managing, working and searching can be done using an ordinary Web browser. Sound links as well as sound annotations that are explained are innovative and new features. The information retrieval process can be done either by navigation through the information structure or by searching. Last but not least a smart read and write access system for different users and user groups, like anonymous access, publishing access, etc. is provided.

10.2 Introduction

The success story of the internet has begun in the 1960s. Up to the 1980s the internet and its services (email, ftp and telnet) was used by researchers and technicians for co-operations and providing data and information [Gütl, 1998]. By integrating the World Wide Web (WWW, W3 or web) the hypertext feasibility was introduced into internet technique. With the help of hypertext a distributed information structure could be built and followed by the so-called

hyperlinks [Maurer, 1996]. By shipping the first graphical browser client Mosaic the web becomes more and more interesting for public [Richter, 1994]. Not only pure textual information was requested, other kinds of media like graphics or pictures have been included. Since the middle of the 1990s multimedia applications have become increasingly common. Further research in the field of hypertext has led to the concept of hypermedia, a combination of hypertext and multimedia. Therefore hyperlinks not only could be followed to text documents, also links could be followed to other media like pictures and sound files [Maurer, 1996]. Vice versa e.g. videos or sound documents could also provide hyperlinks to other destination anchors.

Nowadays web browsers allow more of multimedia applications which are provided by the browsers themselves or can be integrated as plug-ins, active-x components or as a Java applet [Götzinger, 1998]. Web design increasingly uses multimedia components. Therefore more and more embedded sound and digital sound applications can be found on the web. GUV's WWW user survey concerning internet's "Indispensable Technologies" brings an interesting result: "No other technologies listed in the question come closer to these, but the next most popular are chat (22%), Java (22%) and audio (17%)" [GUV, 1997a]. A German survey from ComCult asks for user's interests and found out that approximately 40% of them are interested in audio [Comcult, 1998].

Digital audio techniques combined with internet services provide a wide range of applications, e.g. real audio streaming, support for blind persons, independent music sales and sound archives. Nowadays web standards and additional developments allow to provide digital audio as a further element of a HTML page (loading in background), as an integrative component of a multimedia presentation and as additional hyperlinked objects. The number of hosts on the web can be estimated by an exponential function [Gütl, 1998]. 320 million

web pages are estimated for Dec. 1997 [SEW, 1998]¹. Also more and more multimedia components (like sound files) and their representation of information are available for users. Therefore searching, managing, storing and reusing sound files will become increasingly important. These requirements have led to the presented work which shows a smart solution for an audio database. Using as well as managing and administrating are handled by an ordinary web client. Thus no additional client software is necessary and the system can be used on any computer platform.

10.3 Internet and Digital Audio Formats

An increasing number of users and network traffic as well as more lavish web pages and multimedia components causes much network load and might cause problems in future, because bandwidth is limited and can be seen as a bottleneck. Therefore multimedia components have to be small in terms of size.

For example Netscape's sound capabilities support sound file formats like WAVE and AIFF. WAVE format documents takes much memory or hard disk space as well as transmission time. In case of using a 28,8 kbit/s modem, it takes approximately 12 minutes downloading the 2,64 MByte WAVE sound document (22 kHz cut-off frequency, 16 bit, mono); using this sound file format one minute of spoken information takes 12 minutes for downloading. Considering professional digital audio formats, e.g. the Compact Disc system (CD) has 1,412 Mbit/s data rate for 16 bit and 44,1 kHz sampling rate and the Digital Audio Tape (DAT) has 1,536 Mbit/s for 16 bit and 48 kHz sampling rate. Such digital audio formats offers high quality but does not suffer users efforts using digital audio on the web. On the other hand such high end

¹In 2002 approximately 3.000.000.000 pages are indexed at <http://www.google.com> and more than 160.000.000 web-servers are online (<http://www.isc.org/>)

sound formats perfectly fit for professional work. One can imagine that such file formats could be used for storage reusage, data interchange and further processing steps of digital sound. For example a radio station's digital archive holds a wide range of reports (various local and international topics). For internal use (further processing, reuse, etc.) a broad-bandwidth connection is used and high data rates are on problem. If selected reports are transmitted via internet (e.g. transmission to correspondents or other stations), this will cause long transmission times. Therefore a kind of low quality pre-listening should be provided to reduce transmission time and network load.

To reduce digital audio file sizes as well as data rate requirements a wide range of data compression procedures and digital audio data formats have been invented. Götzingler describes several representative formats which reduce file sizes as well as data rates in case of transmitting such formats [Götzingler, 1998]. Audio compression procedures uses non linear sensation as well as masking. Comparing with CD data sizes, such procedures reach reduction rates between 3 and 7. Combining such reduction processes with less bit solution and sample rates will allow to provide sound files, which could be transmitted via internet. Past and present inventions and developments will let to the assumption that further digital data file formats will be released. A future oriented audio database should allow to use any sound file format and of course to use more than one format for providing different formats for different users and applications.

Summarizing above mentioned considerations, a digital sound database for internet and extranet as well as intranet should allow access via web browsers different sound file formats and different qualities. Furthermore such sound information should hold additional information for investigations and categorizations. Some details will be discussed in the following section.

10.4 Classification and the Need of Meta Information

Considering a book, a journal or a paper, additional information like author, title, date of publishing, description and keyword are used for categorization and for searching. This additional information is called meta information. Meta information is also usable for internet applications, e. g. meta information could occur as meta tags embedded in HTML files or as header and footer showing last modification date and author. On the other hand HTML files could also be used as a multimedia container holding just the meta information and references to a multimedia objects (e.g. picture, source code, video or sound file) [Dempsey, 1997].

Meta information also seems to be a good idea to describe and categorize digital sound information. A pure sound document in a file system only allows to inform about creation and update time of the physical file and the file name. In such a system there is no way of further categorization and description and therefore no way for extended search queries. But searching and structuring of multimedia becomes increasingly important as information supply grows.

In the work presented an object-oriented database provides meta information for any document and media type. Multimedia data as well as meta information are presented to the user via HTML files, most of the dynamically created. Forms and JavaScript Menus for configuring the system or to query for special documents. References allow to select different digital sound file types. Some Meta data is placed in the header or footer of the document or can be reached by special links and is the offered as a HTML table.

A set of proper meta data stretch a multidimensional information space, where each classified meta information is one dimension. So similar documents seem

to be in a virtual neighborhood in this space and form subspaces which can be seen as clusters or clouds. Using meta information for classifying multimedia information offers new possibilities for information retrieval processes. Possibilities for search operations are figured out later in this paper.

Götzinger has described possible ways of meta data sets [Götzinger, 1998]. The author figured out that only one set of meta information does not fulfill the requirements for practical usage of a sound database in general. Götzinger subdivided audio documents in three types: music, speech and sound samples.

A future oriented sound management system will not only have to support different sound data format and meta information. Also additional multimedia objects like cover-pictures, artist information, videos, text of songs, etc. should be managed by a sound database. Thus a well suited sound database will have to be something like a well tailored multi media service.

An important aspect of extranet applications is security and confidentiality. A sound database should provide user management and access rights, so that different users or groups of users can be treated different. You can think of an online music store where registered users may prelisten all offered CDs while the anonymous user only will get a 30 second prelistening. Both, managing multimedia objects and a user management system are handled by the Hyperwave Information Server, which will be introduced in the following section.

10.5 Multimedia and Hyperwave Information Server

The basic ideas of Hyperwave was born already in the end of the 1980s in Graz at the IICM [Maurer, 1996]. Unlike an ordinary Web sever the Hyper-

wave Information Server provides further smart and tricky features like link management and link integrity, multilingual function, user and user group right management, navigation and full-text search feasibility as well as an annotation system. The basic concept of the Hyperwave system is the underlying database which holds the whole set of stored objects and provides attribute fields (meta information like keywords, description and also user-defined fields) for managing and searching for them [Hyperwave, 1998].

As already mentioned the requirements of a future-oriented digital sound database should allow to store and work with arbitrary different sound file formats as well as arbitrary multimedia objects. The Hyperwave system not only allows to store such objects, furthermore the system provides a web user interface to manage such data. Hyperwave's structure elements (collection) allow to build up information container which hold all objects concerning a particular sound. That means that different audio data formats, text documents in different languages and other additional information are hold in one information cluster. This packet is presented to the user depending on his preferences as one single multi media document. Further the structuring features of Hyperwave can be used to build tree like information structures which can then be navigated through. The structuring of sound documents can be done concerning different categorization topics. So you can organize music data by date, interpret, country, music style and so on. Using this categorizations parallel, the users have a good chance to find the appropriate song. Using the Hyperwave's attribute feature one can define a set of meta information, e.g. creating a sound archive attribute for song title, composer, artist, music style etc. Configuring such attributes as indexed attributes, Hyperwave's search facility allows to search for these entries. Hyperwave also provides a full-text search engine, which allows to search for information in the content of text documents, e.g. additional documents for song lyrics, textual content

of spoken text, etc. Unlike text documents searching for other multimedia components like pictures, videos and sound documents requires the supply of a set of attributes, which can be searched for. In addition Hyperwave provides a smart right management which can be applied to any document as well as structure elements.

Combining the discussed features, the Hyperwave Information Server seems to be perfectly fit for a digital sound database application. Implementation considerations and details will be discussed in the following section.

10.6 Digital Audio Database Implementation and Features

As already mentioned in section 10.2, there are published many sound applications like sound databases. Götzingler has investigated five representative internet sound databases in his work [Götzingler, 1998]. He has summarized their advantages, extracted them and has added further smart features. Combining these results with already discussed considerations the implemented digital audio database is based on this experience.

Unlike the possibilities of ordinary Web servers and databases, the Hyperwave features allow a smart application realization. The audio database implementation is designed as a Hyperwave add-on and its additional functions are integrated in the standard Hyperwave menu bar. This implementation enables the integrated usage of the digital audio database embedded in a general intranet or extranet solution as well as the stand alone usage. The functionality can be divided into three different views: the reading user access (anonymous and identified users), the publishing user mode (inserting, editing and erasing of

objects) and the administration mode (managing user and user groups, sound attributes).

The Reading User Access

The represented implementation of a future-oriented digital audio database allows to navigate through the information hierarchy. Our former experience has shown that many users much more like to find their information by navigation through the hierarchy. For example a user is interested in spoken information about jazz sessions in Graz: the user will follow the provided entry “spoken information”, will afterwards select the entry “Graz” and than “concerts” etc. The introduced audio database application may use any structure object provided by the Hyperwave system. The collection object (signed by a folder symbol) can be used for building an information structure, the sequence object (signed by a blue-colored symbol) can be used for defining a sequence of audio data. Such a sequence allows to put arbitrary sound objects together and to decide the sort order. Another tricky feature is to define multilingual objects by using cluster objects. There can be defined e.g. an English and a German version of the same sound file. Users can specify their preferred language and will receive the proper (language-dependent) object.

Users (especially beginners) like to choose out of a set of provided items. Search forms and especially extended search forms are mainly accepted by advanced and expert users. But most times also beginners will want to search the database for very special and distinct information. Thus a future oriented system will have to provide simple to use tools for searching. You can think of Personal Digital Assistants (PDAs) or other intelligent agents as interface between user and database. Agents will be treated in some more detail later.

Another possibility to find information is by navigating through structured data.

A further interesting feature of the audiobase are query objects which allow to store filled search forms. You can very easy create a query object. You just need to fill out a query form and then store the query instead of posting it. The query object can be inserted at any position and looks like a link. By clicking on the link, the query is posted without the need to fill up the same query once again. Furthermore query objects allow to send the result as an email to defined users automatically. You can specify when to post the query (e.g. each day) and where to send the result. So this feature allows a kind of push technique where specified users get information about new entries in the database regularly.

The audio database system provides any relevant information of a sound object by clicking on the sound object icon. The meta information of the object will be shown in an additional window, the info-window. The info-window not only shows the relevant information, there will be provided also a button for downloading the sound file in the pre-selected format. The format selection can be done by selecting the “preferences” menu bar. The pre-listening process will be started by clicking at the sound object icon. database system provides a Java applet-based audio player for pre-listening the “au” audio format. This sound file format will also be used for sound links and sound annotations which is described in detail in the following Paragraph. The Java applet allows a platform-independent audio player support.

Textual information about the sound object, e.g. song lyrics or text of spoken information can be received by clicking the text icon.

It should be mentioned that full text search is a built in feature of Hyperwave. The text information window can also provide hyperlink destination references

as well as sound link feasibility. A so-called sound link is a particular realization of the hypermedia media link concept. Such a sound link allows to refer to a defined part (a section) of a sound object. Therefore one can get a particular part of the whole sound information, e.g. a quotation or a sentence of some spoken information. This functionality will be used at the web based training environment “GENTLE”, which is implemented at present at the IICM. For example this feature can be used for language training: learners can get the proper pronunciation of written words. Another example is to provide additional audio-oriented information for images (e.g. animals).

Another useful feature for comfortable using and working with such data archives are annotations. The concept of annotations allow to notice additional information to a object, to the whole content of the object or part of it. For creating annotations the user does not need to have write permissions on the annotated object, since the object isn't changed. An annotation is just a special link object. The user only need a writable space on the server where his annotation can be stored. For example an editor or a journalist adds annotations to sound objects for planning a broadcasting program or give useful hints to an artist. Even further sound objects (e.g. an interview) may be added as annotation. Annotations underly the same access mechanisms, thus an annotation may be private or visible only for specified users or user groups or even for visible for all.

Using the user management of Hyperwave, its multi parent and structuring possibilities and dynamic document creation features like server side JavaScript the presented sound database is able to provide different views for different users on the same data. Read, write and unlink access rights can be delivered in a very well tailored way, thus allowing confident or protected data. Another possibility is to provide less quality of music for the public as a preview and to give access to high quality data only for registered user (they may have to pay

for the service). The system may allow write access for identified users in a special private working space. This enables features like copying and managing sound objects in their home base, making annotations and defining some query objects for predefined queries and installing a kind of push technique. Fine grained user access rights may also be used to provide special views on the data for each user. This is done by linking only interesting parts of the whole sound data base into the private section.

The Publishing User Mode and the Administration Mode

The administration mode allows to manage user and user groups as well as defining users home bases and configuring attributes which should be indexed. All these function use provided standard Hyperwave features. Detailed information about such provided features could be found at the Hyperwave manuals [Hyperwave, 1998].

The publishing user mode allows to insert, edit and erase new sound objects, different sound files formats, text objects and further multimedia objects. Furthermore already mentioned sound links and annotations could also be managed as well as read and write permission to any object. The following screen shots will illustrate how to insert a new sound object as well as additional text information and sound links.

10.7 Some Notes on Data-Reuse

Nowadays huge information repositories hold information of any topic. More and more multimedia objects are available on the Web and therefore more and more information is stored in objects like graphics, videos, sounds etc. In

our so-called information society people must search for relevant information, wants to find them again and have to store them. The most important criteria is that users can find exactly and rapidly the information they are looking for. But also further processing of information and the re-use of such modules is increasingly important. Information re-use could be seen as a process for recycling parts of document contents or just to use only the pragmatic information of the document. The re-use process can perfectly be used e.g. in the area of web based training for creating new course ware, to fetch information modules for the management information process and for creating new broadcasting contribution. Re-use of information can be separated in two parts, information retrieval and information recycling.

Before the re-use process can be done, relevant and reliable information must be gathered. Finding the relevant information is the most important part of the re-using process. Relevant information can be found on the web or in any archive. But to find out which information is relevant, it has to be examined by referees and categorized by a quality number and further meta information. On the other hand new information can be produced in the own domain (e.g. in the company, the university, etc.). In general such information is generated by trusted sources. Both are candidates for the information re-use process, because they represent defined quality standards.

The second part is to use found information modules in other “domains”. Sound objects could be “recycled” in different ways. Recycling most times means to extract information out of documents, transform that information and include it in the new “domain”. A future work could e.g. be to re-use information by automatically transforming spoken information to textual information for further usage.

10.8 Current and Future Works

Similarity Search

Considering the sound database introduced, each sound object stored is described by a set of attributes (meta information) as well as the textual information about the content of the sound object (song lyrics, description of sound, etc.) which represents a point in n-dimensional information space. The information retrieval process could either be done by defining the “needed” information and looking for objects in neighborhood or defining one object as relevant and looking in neighborhood of the represented point in information space for quite similar objects. Such retrieval process defines similarity of “content” for objects being into a specific information cluster. At present the IICM is working on techniques finding “meaningful” information cluster. The above mentioned basic idea could also be used for investigating multimedia objects (e.g. sound files). On the other hand this technique can also be used to help authors for categorization new objects. One can imagine that objects which are placed in a information cluster will be quite similar properties. The first and also most important step is to find relevant information modules.

Virtual Address Space

The IICM is developing a protocol independent and platform independent middleware for accessing and managing distributed data, the Active Node Technology (ANT). The basic idea is to present the user a virtual address space, a hierarchical view of distributed objects no matter if they are accessed via file, http, ftp or e.g. the native Hyperwave protocol. The basic tool for operating on this information space is the native ANTEplorer (a Java application). The

explorer behaves and looks like a standard Explorer including features like drag and drop for structure modification, context menu and menu merging.

The Active Node Technology and the ANTEplorer will be used as a sophisticated tool for administrating and viewing the sound database. Simple drag and drop operations can be used to publish sound documents or to copy or link sound data into the users private space, context menus are e.g. used to create new folders. Each object in Active Node Technology (think of a sound cluster as one object) can specify special services. So a sound object may provide services like special attribute management or publishing services. Also speech to text conversion and creation of special sound links are managed by services. Adding new services can be done very easily and at runtime. The virtual address space can be configured by the user in a well defined way. So the user can virtually merge information from different sources to generate his special view. He will possibly link some sound database objects, according emails and some local files to form an information cluster concerning a special topic. For more information about ANT see [Zwantschko et al., 1998a] and [Zwantschko and Gütl, 1998b].

Agents

Another topic the IICM is working on are intelligent agents. Intelligent and possibly movable agents will be used for sophisticated search and for guiding the user. Personal Digital Assistants (PDAs) can help the user to work with the sound database advising the user and making suggestions. So special agents will perform the similarity search and for displaying the results graphically. In combination with Active Node Technology, agents will be used to manage distributed data. Learning from the users behavior, the agent will decide where

to store the documents (locally or on a data base), which quality to request or how to present the documents.

A future vision of a sound database can be formed by information agents, where each agent itself embodies the information. Thus information is no passive set of data but an active and communicative, “live” object. The agent is able to communicate with the user and displaying itself. In addition the agent has links to other agents with similar information and active manages this connections (when a user follows a link and put a value on it it is strengthened, otherwise it is removed). Users often will communicate with this information agents through PDAs or special service agents (like search agents). The whole info-sphere thus becomes a dynamic and self configuring network.

Acknowledgments

We want to thank all members of the IICM for their suggestions and help. Many thanks to the employees of Hyperwave R&D for their support.

References

- [Altavista, 1998] Altavista search engine <http://www.altavista.digital.com>
- [Comcult, 1998] ComCult Reserch User Studie, 1998 <http://www.comcult.de/ccstudie/index.htm>
- [Dempsey, 1997] Dempsey, L. Metadata: An Overview of Current Resource Description Practice *Journal of Documentation*, 1997
- [Götzinger, 1998] Götzinger, W. Testimplementation einer Audiodatenbank auf dem Hyperwave Information Server; Diplomarbeit an der Technischen Universität Graz, IICM, Graz 1998
- [Gütl, 1998] Gütl, C. Future Information Harvesting and Processing on the Web “European Telematics: Advancing the Information Society” Barcelona, 4-7 February 1998
- [Gütl et al., 1998] Gütl, C., Zwantschko, B., Götzinger, W., and Krottmaier, H. Managing and Archiving Digital Audio Data in Internet and Extranet. In *20. International Convention on Sound Design (ICSD 1998)*, pages 1116–1136, Karlsruhe.
- [GUV, 1997a] GUV’s WWW Surveying Team Indispensable Technologies; College of Computing Georgia Institute of Technology Atlanta 1997 http://www.guv.gatech.edu/user_survey-1997-10/graphs/use/Indispensable.html

- [GUV, 1997b] GUV's WWW Surveying Team Connection Speed; College of Computing Georgia Institute of Technology, Atlanta 1997 http://www.guv.gatech.edu/user_survey-1997-10/graphs/technology/Connection_Speed.html
- [Hyperwave, 1998] Hyperwave Manuals. Hyperwave Information Server 4.0; Hyperwave R&D, Graz 1998 <http://www.hyperwave.com>
- [Maurer, 1996] Maurer, H. Hyper-G now Hyperwave — The Next Generation Web Solution. Addison-Wesley Publishing Company.
- [Meyer, 1991] Meyer-Wegner, K. Multimedia-Datenbanken; Teubner, Stuttgart 1991
- [MPEG-7, 1999] MPEG-7: Context, Objectives and Technical Roadmap, V.12, Vancouver 1999 <http://ipsi.fhg.de/delite/Projects/MPEG7/Documents/W2861.htm>
- [Richter, 1994] Richter, U. WWW-Begriffe Friedrich-Schiller-Universität, Jena 1994
- [SEW, 1998] Search Engine Watch Search Engine Size, March 1998 <http://searchenginewatch.com/reports/size.html>
- [YAHOO] Yahoo search index <http://www.yahoo.com>
- [Zwantschko et al., 1998a] Zwantschko, B., Freismuth, D., and Schmaranz, K. Telematic platform for patient oriented services Proceedings Web-Net98, Florida US, November 1998
- [Zwantschko and Gütl, 1998b] Zwantschko, B. and Gütl, C. Active Node Technology: Technischer Überblick und Anwendungen Accepted at Java Informations Tage (JIT) in Frankfurt, November 1998

Chapter 11

Adaptive Learning Environment Framework

The following chapter was presented by a co-author at the “International Conference on Computers in Education / International Conference on Computer Assisted Instructions” (ICCE/ICCAI 2000) in Taipeh, Taiwan.

A very important issue in e-learning systems is the usage of metadata which are attached to learning material. Since the published paper does not discuss metadata in learning environments, a short overview of currently available standards and metadata-sets is given in this brief preface.

As already mentioned in the previous chapter, Dublin Core is a well known metadata standard used by many information systems. A special working group (Dublin Core Education Working Group) is involved in the development of element qualifiers and value qualifiers for education-specific elements.

A U.S.-initiative (GEM, Gateway to Educational Materials, [GEM, 2002]) works closely with DC Education Working Group. GEM is based on DC with a few additions related to education-specific elements. Metadata formatted in

GEM-style is widely used in projects such as CHIN's Learning With Museums (CHIN, Canadian Heritage Information Network).

The Institute of Electrical and Electronics Engineers (IEEE) Learning Technology Standards Committee (LTSC) has created a draft standard for Learning Object metadata (LOM). A set of attributes needed for management, exploration and evaluation of digital or non-digital learning objects which can be mapped to DC is defined by the LTSC. LOM is well known in the field of learning systems and often used as basis for other developments. In the draft standard of LOM the following 9 metadata-categories are defined:

General Category: groups the general information that describes the learning object as a whole.

Lifecycle Category: groups the features related to the history and current state of this learning object and those which have affected this learning object during its evolution.

Meta-Metadata Category: groups information about the metadata instance itself (rather than the learning object that the metadata instance describes).

Technical Category: groups the technical requirements and technical characteristics of the learning object.

Educational Category: groups the educational and pedagogic characteristics of the learning object.

Rights Category: groups the intellectual property rights and conditions of use for the learning object.

Relation Category: groups features that define the relationship between the learning object and other related learning objects.

Annotation Category: provides comments on the educational use of the learning object and provides information on when and by whom the comments were created.

Classification Category: describes this learning object in relation to a particular classification system.

A detailed description of these categories and examples can be found in [Draft Standard LOM, 2002].

Another global consortium (IMS, Instructional Management System, [IMS, 2002]) started in 1997 and became a non-profit organization (IMS Global Learning Consortium) in 2000. IMS produces among other specifications a metadata- and content packaging specification. IMS uses IEEE LTSC LOM as its base and it is possible to map metadata to more general DC elements.

A recent agreement in 2001 (called the Ottawa Communiqué) among Dublin Core, IEEE and IMS promises “significant harmonization and collaboration [...] in the areas of educational metadata interoperability and implementation”.

In the following sections an overview of an adaptive framework is given. With this framework it is possible to integrate different systems in the context of learning environments. It helps to use different systems at the same time. An adaptive framework will take different system aspects into consideration when creating views of the connected systems.

The original contribution was published in the conference proceedings and can be found in [Dallermassl et al., 2000a].

11.1 Abstract

In this paper the *Adaptive Learning Environment Framework* is presented. This Framework allows to re-use, combine, and improve existing learning systems or knowledge-databases and equip the resulting meta-learning environment with a new advanced user interface. Especially the last mentioned feature supports a completely new way to learn different materials from different sources without having to switch between systems. The introduction of an abstraction layer between different learning environments and the introduction of different views on the contained learning material allows to combine, extend, and improve available learning systems easily.

11.2 Motivation

Different learning environments offer different ways to visualize the information and also different functionality (e.g. add annotations, communicate with other students, ...), but *views*, *functionality*, and *data* usually are tightly coupled. This implies that people using more than one learning system have to adopt to different user interfaces, a fact that does slow down learning speed a lot. As can also be seen in the classical learning environment (table, books, paper, pencil) even minor changes in the environment very often have a negative impact on the learning quality and so on the learning speed. Once the learning environment (the table) suits students well, only the learning material (the books) and the used tools (preferred pencil, ruler, or calculator) are subject to change.

Today's world of computer based learning environments consists of various systems that are mostly incompatible with each other. There exist products

offering a broad range of high quality information, but they come along with a poor user interface, others try to impress the user with a fully featured user interface, but lack sophisticated material or lack much needed functionality. Some programs use a local database (e.g. on a CD-ROM) to retrieve their lessons, others use the world-wide-web as a source for information and for communication with other users.

Neither of these ways leads to a perfect solution concerning transfer speed, actuality of the material or the way information is presented. Additionally, the majority of learning systems are incapable of using information prepared for other systems, either because they do not understand the data format or because they are incapable to use the appropriate communication protocols.

A system that allows to combine different electronic learning systems and that also allows to use the superset of their features, thus extending it with additional functionality (e.g. the ability to add annotations to the learning material), would increase the quality of the learning process enormously.

In the following sections, we will introduce the *Adaptive Learning Environment Framework*, a library that allows to create such a system easily by making the needed technology available in a component-based way.

11.3 System Requirements

Usually learning systems are very hard to extend, as the learning material is tightly coupled to the view on the data. The classical method to overcome this problem is to split up data, view and functionality (Model-View-Controller design pattern [Buschmann et. al., 1996, Gamma et. al., 1995]). This is also the first and most important requirement of the described framework: provide a

modular, component-based architecture that makes it easy to create an adaptive learning environment.

These modules should allow to *re-use data* from already available learning environments (e.g. Gentle Web-Based-Training [Gentle], Dictionaries, ...). As much material as possible should be extracted from the learning systems, so the user must not lose any information in comparison to the usage of the “original” system. Also the systems’ functionality must be made available to the newly created environment. If someone is used to the annotation mechanism in e.g. the Gentle-Web Based Training environment, this functionality must also be available when the Gentle System is “re-used” by the *Adaptive Learning Environment Framework*.

Separation of material, representation, and functionality offers great possibilities:

It allows us to create a *superset of information*: Imagine having different learning systems available, each of them representing a specific subject. Take the following information systems as an example:

- A CD-ROM about World War Two, including historic images and reports.
- A geographical information system that publishes maps and additional information from all over the world. This material is accessible via Internet and a Web browser only.

Both systems work independently from the other. But only a combination of these two information sources allows the users to get a deeper understanding of the facts and figures of each system: the CD-ROM supplies the user with images and films that would otherwise overstress the bandwidth of an Internet

connection, while the online system supports current information about the areas presented on the CD-ROM during war time. This example shows that even the sources containing the learning material might be distributed across a network.

Merging of only two related materials shows, how a simple combination of two systems, even without adding any functionality, offers new advantages for students. The technique can also be used to improve existing systems and supply them with new functionality that wasn't previously foreseen. Once a module is created that is e.g. able to handle annotations or allow online discussion with other students, all participating learning systems benefit from this functionality.

Getting the ability to merge independent learning material and add some useful functionality is only one half of the requirement. The interface that the user is confronted with, is at least as important as the contained material, as a good human computer interface optimizes the learning effort [McFarland, 1995, Hedberg et. al., 1993].

We use the term *view* to describe the (mostly visual) appearance of the learning material. This appearance depends on various factors, e.g.

- The *output device* the user works with
- The *environment* the user works in
- The *role*, the user plays (student, administrator, instructor, ...)

More characteristics could be listed, but for the moment let us concentrate on the ones mentioned above. Please see section 11.5 for a more complete and detailed list of factors.

According to the Model-View-Controller design, the view on the information must be decoupled from the data-model. This implies that there may exist *different views for the same content*. The task to create a view for e.g. the output device “Web browser” implies that the learning material has to be converted to HTML (or any other format that the web-browser is able to display) and then has to be provided via a web-server.

The fact that different users play different roles in connection with the learning material implies that different roles have different *access rights* to the information. Even if the data source does not provide an access control system, it must be guaranteed that only users with sufficient rights may edit/change/delete the educational or administrative data.

These system requirements lead us to the following concept of the *Adaptive Learning Environment Framework*.

11.4 Concept

The facts that the *Adaptive Learning Environment Framework* has to be modular, has to decouple data source(s) from view(s), and may add functionality to the underlying learning systems (see section 11.3) results in a middleware design approach: Figure 1 illustrates how the framework is put in between the data source containing learning material and views that are used to work with the information.

The Dinopolis [Dinopolis] framework, which is being developed on the IICM provides such a generic framework [Dallermassl, 1999]. It allows the integration of various types of databases or applications and is able to provide the content in a highly dynamic way to a broad range of clients/views. The integration of learning material is one special application of the general concept of Dinopolis.

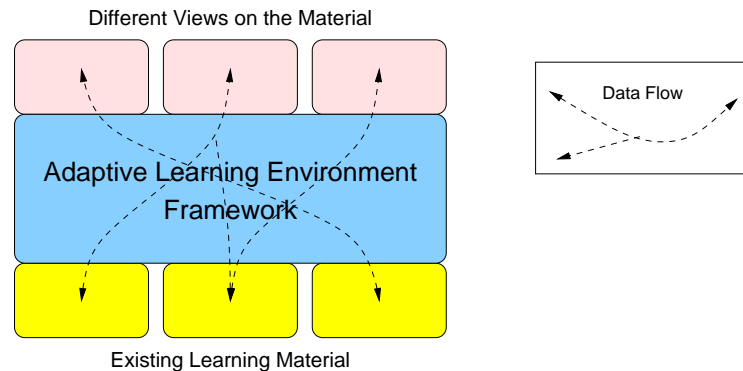


Figure 11.1: *Adaptive Learning Environment Framework:* a layer is put in between the data sources containing learning material and the views that are used to work with the information. Data flows from one or more material sources to one or more destination views.

Integration of any kind of data sources was one of the main issues for the Dinopolis framework, but this is out of scope of this paper. Our main concern in this work is to show some aspects on how the integrated material is “republished” by use of different views (section 11.5). Nevertheless we will give a short overview of other features that are involved in creating learning environments with the use of Dinopolis:

- The *internal document model* of Dinopolis follows the Document Object Model (DOM) specification of the W3C [W3C DOM, 1998]. DOM is used as a mediator between different document formats. As DOM is a widely accepted industry standard, support for a broad range of document formats is available.
- The *security model* of Dinopolis allows very fine-grained restrictions of user access rights [Haub, 1999], so roles like *administrator*, *student*, *instructor*, or *candidate for exams* are easily adaptable.

- The usage of the Dinopolis framework allows to connect more than one Dinopolis system in a *peer-to-peer network*. The middleware layer encapsulates all network-specific operations so every system is able to retrieve the content from the remote Dinopolis system *transparently*.

This modular concept of the framework makes it possible to highly customize and personalize the appearance of the used material. These modules called *views* are described in the following section.

11.5 Views

A *view* is responsible for giving a visual and logical representation of the learning material. It is completely decoupled from the actual information, so it may be (re)used independently from the information stored in the learning environments. A default view on the course material is usually created by the instructor or the administrator where in principle the sources of the course material are selected and positioned on the screen in an easy-to-handle way. Since we do not assume that the users of the system are computer-experts, there must be an easy-to-use interface to define such a view.

The selected source can also depend on system or environment parameters. We mentioned the ability to use different kinds of information sources. It is not suitable if the system always tries to deliver the same kind of data source. To make this fact clearer, assume you are consuming a fully featured multimedia-supported course. If you have a computer with a sound device, it makes sense to transfer the sound files. On the other hand, if you are using a small PDA (personal digital assistant) without any sound device, it doesn't make any sense to transfer the sound files, because the device is incapable to

play the sound. There are other parameters described below which influence the kind of information source.

Due to the individual style of learning it is absolutely necessary that such a default view is completely personalizable. Not just the colors [Kienegger-Domik, 1995] and the style of the displayed text should be modifiable, but also the position of the different kinds of content of a course (navigation bar, the links to the background library, links to the used tools, the content itself etc.) must be exchangeable. This view – other systems call it “personal workspace”, “personal environment” etc. – must also be applicable to other courses. As an example: If somebody is used to having the navigation bar on the bottom of the screen, the navigation bar must remain there in all enrolled courses. It is clear, that the position of such an element can be changed either for all courses or just for one specific course.

The view might not just depend on the personal preferences and the system parameters, but also on the role the user currently has in the system. As another example let us consider a person preparing some course material (usually, but not limited to, the instructor). After adding and selecting the sources of the course the user might switch to the role of a student to have a look at the newly created course. It is obvious that the available tools change, depending on the user’s role.

Let us now consider some circumstances where it is necessary to change the view on the learning material. Take a fully featured multi-media course using different sources of information (integrating online and offline systems), different levels of detail (overviews but also information for specialists), etc.

Transfer Speed: Since we are working in a network environment, transfer speed is a big issue. The higher the available bandwidth, the higher the quality of e.g. the displayed images might be. But also the latency

accepted by users must be taken into account. If users need the best available quality of images and accept some seconds for retrieving the files, the system must deliver these files. If users on the other hand don't want to wait for more than two seconds for a certain image to be displayed, the system has to select an appropriate lower-resolution image file. If the appropriate image file is not available, the system may convert the image to a matching image on the fly.

Output Device: Several output devices may be used for consuming a prepared course. Workstations, laptops, terminals, PDAs etc. which are connected to the used network are able to be used as browser-platforms for the system. As a simple terminal is not able to display images, why should the client on a terminal request them? This makes it clear that also the type of the output device must be considered when delivering information.

Available Sources: We mentioned the online- and offline systems and a combination of those, the hybrid systems using both technologies. If the user is using a computer with the appropriate CD-ROM in the drive, the system might deliver a high-quality movie on request. If on the other hand the CD-ROM is not in the drive, the system has to use the online system to retrieve the movie. In this case also the available transfer speed has to be taken into account and it might happen that not a movie, but some images are shown on the client instead.

Consuming Environment: Considering the environment might be a bit confusing. But after the following example it should be clear, that also the environment where the course is consumed must be considered by the system. Since laptops are getting cheaper and also smaller, they can be used almost anywhere. Not just in the office or in the home office, but also in subways, in trams, in the park etc. It's clear that you might not

want the system to play sound-files, disturbing your neighbors when you are on the bus. Of course you could switch off the sound device, but the sound-files would also allocate some bandwidth and reduce the overall transfer speed of the material.

Level of Knowledge: Depending on their actual knowledge of the learning material the users might not need every basic explanation on the topic of the course. In this case a special hyperlink that leads to the necessary information in a background library might be the solution to the problem. On the other hand, if you are new to the topic, you might want that the explanation for a term is displayed directly on the screen after the first occurrence of that term.

Role: We already mentioned different roles when consuming a course. There might not just be an instructor and some students but also separate administrators or other roles. A typical task of an instructor is creating courses by reusing already existing material or maybe even start from scratch. A student might also create course material, but surely is not involved in administrating user accounts. The student may also be in the role of a candidate for an exam and may fill out an examination in a limited amount of time. A pure administrator may not create courses but may consume information provided by the system.

Desired Level of Detail: The system should also consider the level of the desired detail set by the user. It makes a difference if the user wants to know something about a particular topic just as a hobby, or for profession. It must be possible to switch between different levels in one learning session.

Personal Preferences: It is an obvious fact that some users prefer different fonts, colors etc. Also the position of e.g. the navigation bar, the content

window/frame etc. depends on a unique personal learning style. Since it is possible that a user is using the environment on different output devices, it must be possible that the personal preferences are accessible from every device.

As one can see, there are many different circumstances that can be taken into account when visualizing learning material. Due to the concept of the *Adaptive Learning Environment Framework* it is possible to consider these different aspects on every integrated existing learning environment.

11.6 Conclusion

The *Adaptive Learning Environment Framework* presented in this paper is a powerful tool to add a surplus value to existing learning systems and to reduce the burden on students to get used to new interfaces to the material with every virtual class they enroll. Additional features relieve all participating persons, including the tasks of administration and preparation of courses.

It is obvious that the integration of different learning systems may be a very complex task, but the modular structure of Dinopolis including the internal document model is a powerful tool the learning community will not want to miss in the near future.

References

- [Buschmann et. al., 1996] Buschmann F., Meunier R., Rohnert H., Sommerlad P., Stal M.: *Pattern-Oriented Software Architecture: A System of Patterns*, John Wiley & Sons, New York, (1996).
- [Dallermassl, 1999] Dallermassl Ch.: *Aspects of Integration of Heterogenous Server Systems in Intranets - The Java Approach*, IICM (Institute for Information Processing and Computer Supported new Media) 1999
- [Dallermassl et al., 2000a] Dallermassl, C., Haub, H., Krottmaier, H., Schmaranz, K., and Zambelli, P. (2000a). Adaptive Learning Environment Framework. In *Proceedings of the International Conference on Computers in Education/International Conference on Computer-Assisted Instruction (ICCE/ICCAI 2000)*, Taipeh, Taiwan.
- [Dinopolis] Dinopolis: *Homepage of the Dinopolis Open Source Project*, available at <http://www.dinopolis.org>
- [Draft Standard LOM, 2002] IEEE Learning Technology Standards Committee: *Draft Standard for Learning Object Metadata*, IEEE 1484.12.1-2002, 15 July 2002
- [Gamma et. al., 1995] Gamma E., Helm R., Johnson R., Vlissides J.: *Design Patterns: Elements of Reusable Object Oriented Software*, Addison Wesley, 1995.
- [GEM, 2002] The Gateway to Educational Materials (GEM), 2002 <http://www.geminfo.org>

- [Gentle] Gentle-WBT: *Homepage of Gentle-WBT: Web Based Training*, available at <http://wbt.iicm.edu>.
- [Haub, 1999] Haub H.: *Aspects of Access Management in Heterogenous Distributed Object Systems*, IICM (Institute for Information Processing and Computer Supported New Media) 1999
- [Hedberg et. al., 1993] Hedberg, John G., B. Harper, and C. Brown: *Reducing Cognitive Load in Multimedia Navigation*. Australian Journal of Educational Technology 9, no. 2 (1993): 157-81.
- [IMS, 2002] The IMS Global Learning Consortium, Inc. <http://www.imsproject.org>
- [Kienegger-Domik, 1995] Kienegger-Domik G.: *Intelligent Visualization Systems in Educational Environments*, Proceedings of ED- MEDIA '95, Graz, Austria, pp.17-22, June 1995
- [McFarland, 1995] McFarland, Ronald D.: *Ten Design Points for the Human Interface to Instructional Multimedia.*, T.H.E. Journal 22, no. 7 (February 1995): 67-69.
- [W3C DOM, 1998] W3C: *Document Object Model (DOM) Level 1 Specification* (1998), available online <http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001>

Chapter 12

Using Highly Sophisticated Middleware for Building Arbitrarily Distributed Teaching Environments

This chapter was presented by a co-author at the “International Conference on Computers in Education / International Conference on Computer Assisted Instructions” (ICCE/ICCAI 2000) in Taipei, Taiwan.

Single systems, especially digital libraries, cannot cover all of a user’s needs, therefore several systems must be integrated into one coherent system with a single “look-and-feel”. A user should not become aware of systems boundaries. A highly sophisticated middleware may remove systems boundaries. This chapter explores, similarly to the previous one, the Dinopolis middleware system and applications related to teaching environments. Using the Dinopolis

systems makes it very easy for system administrators to seamlessly integrate different systems into one environment with an adaptive user interface.

The original contribution was published in the conference proceedings and can be found in [Dallermassl et al., 2000b].

12.1 Abstract

This paper deals with the development of highly sophisticated teaching environments. We took a look at the requirements that such a system has to fulfill to meet the needs of our modern society and to remain easily adaptable to forthcoming, new technologies. The results of our research show that the cost for design and implementation of a distributed teaching environment can be dramatically reduced by using a middleware system. We thereto present the concepts of the Dinopolis middleware system which is highly modular and extendable and show how it may be used as the basis for the development of teaching environments. Using the Dinopolis middleware system not only eases the process of development but also guarantees that new technologies are becoming available without any further effort.

12.2 Motivation

We are living in times of ongoing rapid changes. New technologies support our everyday life in a way we could not even imagine a few years ago. New media are also disrupting our old fashioned way of thinking about education and pave the way for new visions [Grudin, 1994].

Many different approaches for developing electronic teaching environments already exist. All of them have in common that they try to react on the changes taking place in our modern society. Acquired knowledge is becoming obsolete in a short amount of time and has to be updated and expanded. The term “Lifelong Learning” is becoming more and more important. Teachers and students are often no longer present at the same location. They are not even active at the same time. Therefore lectures have to be available whenever and wherever one likes. On the other hand new technologies have to be made available as soon as possible. As an example consider the new WAP protocol which could make it possible to pursue a course using a mobile phone. This implies a highly extensible and expandable system.

Teachers preparing a course often want to use already existing data. This data might be distributed among the network and might only be available in proprietary formats. It is an unbearable effort to collect and extract this data to feed it into a teaching environment. It has to be possible to easily include existing data with no further effort. It has to be possible to add or manipulatable information at any time without knowing any details of the underlying systems. When updated information has become available, documents have to be replaced. Nevertheless replaced information should not be lost as it shows the history of a course and might be interesting for some research. Thus a version control mechanism is desired as well.

Students need easy access to the teaching system, no matter which system they use (platform, browser, etc.). The way in which information is provided has to be adapted to the user’s special needs and interests. The users’ skills have to be taken into account as well as the capabilities of the system used. As an example, consider the network bandwidth with which the user is connected. Lower bandwidth could be considered by sending images of a lower resolution and just sound without the video. Users of different skills need information

prepared differently, which means that different applications are provided for novice and expert users. Thus the system has to be highly customizable, not only by the user but also automatically by the system itself. Different views at the information space are desirable as well. As an example the information could be provided in different languages or with different, localized examples.

Another important point is the use of background libraries [Marchionini and Maurer, 1995] and to find more details on a topic of interest. This includes dictionaries, encyclopedias and glossaries which again have to be adopted to the user's needs. As an example native English speakers would need an explanation of an unknown English term in English, whereas German native speakers would like to get a German explanation.

Though students may be distributed among the world they need the ability to discuss certain topics with the teacher or among themselves. Collaboration and communication tools are therefore required. This includes chat, discussion forums, video conferencing tools, etc.. Another important aspect is the possibility to add annotations and make personal notes to a topic which may only be seen by a specified group of others, maybe even excluding the teacher. This directly leads us to another interesting aspect in a teaching environment: user and group access management. Information presented in a course as well as additional information has to be protected from unauthorized access. The more sophisticated the user management is, the more configurable the system becomes.

Obviously, the best user access management system is not worth anything if the transmission via the network is not secure at all. Highly sensible data has to be additionally encrypted. Imagine companies teaching their personnel internal knowledge.

For performance reasons hybrid systems are often desirable. Most of the course-ware comes along on a CD-ROM. Only the logical part of a course could reside on a server. Thus lessons can be arbitrarily constructed using the information stored on the CD-ROM. Only additional or updated information has to be fetched via the net. This keeps network traffic low without omitting special kinds of media such as videos. Hybrid systems are also kept up-to-date more easily, since outdated information stored on the CD-ROM can be replaced dynamically by new data from the server.

Linking is a good way to increase the value of information if used for defining the course flow [Van Dyke and Sollins, 1995]. This makes it easier to add or replace information used in a course. That way even statically stored information (e.g. on a CD-ROM) can be dynamically restructured to meet different needs. Since links are often subject to change or become invalid (especially links to external resources), a highly sophisticated link management is required to keep the system consistent.

Developing a teaching environment that meets all the functionality described above implies a big effort for implementing the base functionality. Most of this functionality has little to do with a certain teaching concept pursued. It requires a lot of time to develop the system so far that the actual vision can be put into practice. This paper describes how the cost for developing can be reduced dramatically when using a middleware system. This offers more time to concentrate on the crucial points of new ideas. In the following sections we present the abilities of the Dinopolis [Dinopolis] middleware system which is highly integrative, expandable and configurable and provides an easy-to-use interface for building distributed applications.

12.3 A Distributed Environment

As already mentioned, distance learning gets more important every day. Teacher and students do not have to be physically present in a classroom. The teacher for example holds online lessons in his office, while the students follow the lessons from at home or from some Internet terminal on the campus. It is important to recognize that the students do not only need to follow the lessons conducted by the teacher, but also need the possibility to interact or communicate with the teacher during and after the lessons.

Please note that not only human beings are separated in such a distributed environment but also resources can be spread across the network. As an example, the lecture notes can be stored on the teacher's laptop while a chat tool may be located on the university server.

The main task of modern teaching environments is to bridge this physical separation of teachers, students and resources. This can be supported by using the Dinopolis middleware system. A middleware system adds another layer to the teaching environment. It decouples the network and data from the actual teaching application. It somehow resides between them. The Dinopolis middleware system allows the integration of arbitrary systems and provides a uniform way of access. Applications using Dinopolis do not have to worry about where the data is located and which protocol has to be spoken to retrieve the data.

From an application's point of view there is no difference if the information resides on a web server, on a database server or just on the local file system. The underlying systems may even be exchanged transparently. An important point is that integration within the Dinopolis system is not done on a common denominator basis but on the contrary, additional functionality is added when

needed. As an example a file system can be combined with a database system to allow the storage of meta data which the file-system itself does not support.

The most intuitive way to access data on a remote Dinopolis system is by “merging” the two Dinopolis systems, since it allows transparent access to remote Dinopolis instances. In other words the middleware layer bridges the physical separation and applications can be written as if they would run on one single system. Nevertheless the Dinopolis *External Access Gateway* concept allows access in various forms. At the moment this includes HTTP, FTP, WAP, LDAP, CORBA, RMI, etc.. CORBA and RMI are remote object systems which enable the use of objects over the network. In the case of CORBA [OMG, 2000] the Dinopolis object system can be used by programs written in various programming languages.

Thus no matter how distributed teachers, students and resources are, the Dinopolis middleware system acts as if they were all local. For a more sophisticated discussion of middleware systems and Dinopolis see [Dallermassl, 1999].

12.4 Data and Course Flexibility

As already mentioned information is subject to change and the amount of information grows dramatically nowadays. Data must not be stored within a static teaching environment providing no possibility to adapt the data. It is also desirable to change the course flow dynamically to increase the value of information. As an example it has to be possible to add actual information to a course at any time. Thus developing a teaching environment built on static information is not worth the effort.

The Dinopolis middleware system allows to add, exchange and modify data at any time without needing to know any specific details about the underlying heterogenous information space. Version controlling guarantees that no information is lost. The distributed concept of Dinopolis allows to store data at arbitrary places transparently. Applications do not have to worry about that. This makes it relatively easy to develop so called hybrid systems, which combine local statically stored data (e.g. on a CD-ROM) with dynamically retrieved data (e.g. from the web). Due to the network bottleneck hybrid systems have the advantage that they do not have to omit bandwidth consuming information, such as videos, since they can be retrieved locally. Only updated information has to be downloaded via the net.

The course flow is best modeled using links. This makes it possible to insert or remove any kind of information without having to construct a complete new course. The problem with links is that they often become invalid or point to unintentionally changed data. Dinopolis comes along with a fully featured consistent link management system. Links do not point to a location but to an object. Thus even if objects are moved in the Dinopolis system, links remain valid. It is also possible to add meta data to links to provide additional information. Again the possibility to set links does not depend on the abilities of the underlying systems. As an example imagine a courseware CD-ROM. Since a CD-ROM is read-only it is impossible to add or modify links directly on that medium. Nevertheless Dinopolis adds this functionality using an external link database (e.g. Oracle). This database can easily be exchanged transparently which then leads to a completely new course or course flow.

Dinopolis is able to handle arbitrary document formats. This is achieved through the Dinopolis internal document model, which follows the *Document Object Model* (DOM) specification of the W3C [DOM, 1998]. DOM is a highly accepted standard which supports the exchange of data between various kinds

of different document formats. The internal document model again makes the whole system independent from the underlying ones.

12.5 An Extendable and Exchangeable Environment

A modern teaching environment not only has to be flexible concerning the course data and course flow. As applications evolve new technologies and requirements arise which require modifications of existing teaching environments.

Dinopolis is built on a completely modular basis. This allows to add or replace arbitrary parts of the system without producing unexpected side effects on the remaining parts of the system. Since Dinopolis is completely written in *Java*, it is possible to load new modules via the network even at runtime. Statically designed systems would not only have to be completely rewritten but also have to be redistributed among all users. As an example for a newly available technology consider the WAP protocol which could make it possible to pursue a course using a mobile phone. Dinopolis only requires a small WAP speaking module to be added and all applications using Dinopolis are becoming WAP aware. This means new technologies are available without any further effort. By the way adding new functionality to an application by hand would also require to be familiar with all the underlying details, which might be unnecessarily time consuming. Since Dinopolis is an *Open Source* project it is also guaranteed that modules supporting new technologies will be rapidly available.

Dinopolis is not only modular concerning external communication gateways. Internal parts of the Dinopolis system may be exchanged or added transpar-

ently as well to meet different needs. As an example let's take a look at the *User Access Management and Security* system which will be explained in section 12.7 more in detail. For some systems it is sufficient to define simple read and write access rights for users. Other systems require a much more sophisticated mechanism, such as a rule based one which only grants access if certain complex conditions are fulfilled. As an example students who have already passed an exam might get access to the results which are protected otherwise.

Teaching environments transmitting highly sensible data might also require to encrypt the information sent over the network, while a public system would not require it. This implies a *Security Manager* that can easily be adapted to different strategies. This also includes that some systems demand a user authentication based on smart-cards, finger prints or retina scans.

Dinopolis makes it easy to exchange or adapt internal parts to meet different requirements without the need to write a new application or modify existing ones. Dinopolis even allows to exchange the internal data structure or communication protocol used. At the moment data is stored according to the XML standard [Bray et al.,1998] and communication between Dinopolis instances is done using RMI or CORBA. As soon as new technologies and standards become available they will be integrated as well.

12.6 A Highly Customizable Environment

In a modern teaching environment we expect the main parts of the system to be highly customizable. The whole system has to allow users to turn on and off certain features according to their needs and their systems' capabilities. On the other hand applications and tools have to be customized depending

on personal settings. Additional communication tools have to be provided to support a better information flow between users. The Dinopolis system is highly modular which allows adding and removing of integral parts, features, and services at runtime.

The users should have the possibility to choose between a variety of tools and applications that support their studies, depending on the users' skills and the used network connection.

The Dinopolis system makes it easy to write distributed applications. Existing tools can be reused and adapted for certain lectures as needed. It is easy to configure the system with different applications at runtime and the Java Classloader allows starting applications which reside on the local terminal, a CD-ROM, or to download them from an application server.

It is important that only those applications are part of a certain course, which are actually necessary.

Next it should be possible to customize the applications and tools themselves. As an example consider a video-conferencing tool which can be run with different frame-rates and resolutions. On the other hand it has to be possible to turn on and off special features like, for example, strong data encryption. Applications written for Dinopolis can make use of certain features or not, according to the environment in which they are used.

Apart from the applications and tools used, such a teaching environment has to allow customizing the users' view on the data stored in the information system. As an example consider a teacher who wants to adapt the data representation according to the different skills and needs of the users.

Therefore the Dinopolis system provides so-called Views, which support different data-representations. Using a highly sophisticated link management (see section 12.4) the data representation can be customized in a very high degree.

Another considerable point is that it has to be possible to decide where the data is actually stored. So it could be desired to store certain data on a file-system on the users' terminal or in a central database. It should be possible to transfer these data from one storage medium to another. Since the Dinopolis system uses an internal data representation which is independent from the medium it is stored on, it is possible to store the data on different devices according to the actual configuration of the system.

12.7 User Access Management and Security

All distributed application have in common that their reliability and consistency heavily depends on the security and user access management. Unauthorized access to certain resources has to be prevented. This is also appropriate for a teaching environment. Just imagine students modifying and corrupting course data or exam results.

There exist many different approaches to solve this security task. Which one is appropriate for a certain environment depends on the desired degree of security. As already described in section 12.5 a simple security system differentiating between read and write access rights may be sufficient for some applications. More sophisticated environments could require a rule based access control system or secure (encrypted) transmission over the network.

One of the main parameters to measure well designed software is its level of reusability. In this case this means that security strategies have to be

exchangeable easily. As already mentioned replacing the Dinopolis security concept is no complex task due to its modular structure.

Another point of high interest concerning access management in a teaching environment lies in the fact that data is distributed among heterogenous systems, all of which coming along with different or even no security management. It has to be guaranteed that users are forbidden to access data which they would not be allowed to access otherwise. Additionally if a system does not support any access control (e.g. MSDOS file-system) it has to be possible to add a security functionality. Dinopolis allows to integrate existing security mechanisms transparently. It is also possible to add access control to systems which do not support that by themselves. Thereto Dinopolis uses its integrated link management system. Access rights and rules are simply assigned to users and data through links which may be stored in any arbitrary external database. This concept makes it even possible to assign access rights to read-only systems (e.g. CD-ROM). Last but not least Dinopolis supports mandatory as well as discretionary access control. This means that access rights may be assigned to users as well as to data, which allows highly sophisticated combinations of rights.

For a more in detail discussion of access management in distributed systems and in Dinopolis see [Haub, 1999].

12.8 Conclusion

The Dinopolis open source middleware system presented in this paper is a powerful tool for building arbitrarily distributed teaching environments. It allows to integrate data from heterogenous information space transparently and makes them available through a common interface. Thereby functionality

is not reduced to a minimum but on the contrary, additional functionality is appended where needed.

The modular concept of Dinopolis makes it easy to adapt the system to specific needs without producing unexpected side effects on the remaining parts of the system. Its high customizability makes it possible to configure the system to meet personnel needs. Additional communication tools such as chat, video conferencing, etc. may be used together with the teaching software without any further effort.

Courses may be constructed using all available distributed resources which also eases the creation of hybrid systems. Easy access to the system is possible not depending on the clients' platform or desired protocol.

The conglomerate of Dinopolis' features presented allows developers of teaching environments to concentrate solely on the crucial points of their ideas. Thus we believe that Dinopolis is going to play an important role in the future development of distributed teaching environments.

References

- [Bray et al.,1998] Bray T. et. al.: “Extensible Markup Language (XML) 1.0 (1998)”, <http://www.w3.org/TR/1998/REC-xml-19980210>.
- [Dallermassl, 1999] Dallermassl, Ch.: “Aspects of Integration of Heterogenous Server Systems in Intranets - The Java Approach” IICM (Institute for Information Processing and Computer Supported New Media) (1999)
- [Dallermassl et al., 2000b] Dallermassl, C., Haub, H., Krottmaier, H., Schmaranz, K., and Zambelli, P. (2000b). Using Highly Sophisticated Middleware For Building Arbitrarily Distributed Teaching Environments. In *Proceedings of the International Conference on Computers in Education/International Conference on Computer-Assisted Instruction (ICCE/ICCAI 2000)*, Taipei, Taiwan.
- [Dinopolis] Dinopolis: “Homepage of the Dinopolis Open Source Project” <http://www.dinopolis.org>
- [Haub, 1999] Haub, H.: “Aspects of Access Management in Heterogenous Distributed Object Systems”, IICM (Institute for Information Processing and Computer Supported New Media) (1999)
- [Grudin, 1994] Grudin, J.: “Groupware and Social Dynamics: Eight Challenges for Developers (1994)”.
- [Van Dyke and Sollins, 1995] Van Dyke, J.R. & Sollins, K. R.: “Linking in a Global Information System, World Wide Web Journal - A Publication of the W3C” (1995), <http://ana-www.lcs.mit.edu/anaweb/pdf-papers/tr-659.pdf>.

- [Marchionini and Maurer, 1995] Marchionini, G. & Maurer, H.: “Digital Libraries as Components of Modern Computer Supported Learning Environments” in proceedings of ED-MEDIA 95. World Conference on Educational Multimedia and Hypermedia, Graz Austria 1995.
- [OMG, 2000] Object Management Group, Inc.: “CORBA Language Specifications Summary”, OMG (2000) <http://www.omg.org/store/publications.html>.
- [DOM, 1998] W3C: “Document Object Model (DOM) Level 1 Specification” (1998), <http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001>

Part V

Summary and Outlook

“The best way to predict the future is to invent it.” *Alan Kay*

“Predictions are very difficult, especially about the future.” *Niels Bohr*

Chapter 13

Conclusions and the Future of Digital Libraries

This chapter summarizes and concludes the dissertation. Main research results are listed and discussed briefly. Thereafter the future of electronic publishing in general and the Journal of Universal Computer Science in particular is anticipated.

13.1 Summary, Results and Conclusions

This dissertation showed current research issues in the field of digital libraries and discussed details of the Journal of Universal Computer Science. After an introduction to electronic publishing systems and digital libraries, an overview of currently available systems was given.

Archiving of contents is a main issue in electronic publishing systems. In the research environment used information entities are organized in collections. In Hyperwave terminology, a collection is a kind of “folder”, where different

objects may be located. At the moment three different electronic formats and two different types of contents (the abstract of an accepted paper and the paper itself) are stored in such a collection. Since it is not predictable which formats will be viewable and accessible in the future, this concept will ease staying up-to-date with new electronic document formats.

There are many differences in electronic document formats and each of them has advantages and drawbacks. Some are easy to view on screen but difficult to print and vice versa. Therefore the design of the information structure (i.e. the open collection hierarchy) with several different document objects (in different electronic formats) was a good choice. Content is available in high quality printable format (PDF and PostScript) as well as in highly adaptive format. Therefore it is easy to generate a printed version of contents stored in J.UCS.

The following three topics in the field of electronic publishing were discussed in detail and presented at different conferences:

- Concept and possible implementation of transclusions
- Improvements in the usability of digital libraries
- Applications and integration of electronic publishing systems

Transclusions, a concept of reusing content without coping it, is discussed in detail in this dissertation. It has been shown that transclusions have many advantages over ordinary “cut-and-paste”. Since no content is actually copied but just referenced, the rights of property are not violated. When using transclusions disk space can be saved because there is *no duplication* of content and problems with outdated content will disappear. Transclusion will allow users to work with a document in many ways. Since all parts that are reused in a

document will be marked by special signs, it will be visible to users, which parts are reused and which are original contributions of the author. Reused parts may be explored simply by visiting a link. When looking at a *transcluded document* the reused part will be marked. Transclusions are therefore a kind of *implicit metadata to parts of a document*.

Many existing applications (such as a discussion forum or a course material publishing system) will be easier to handle for users if transclusions are used. Especially in digital libraries this kind of metadata to *parts of a document* will provide the user with additional information about the document.

Using a highly sophisticated server system makes it possible to offer knowledge management tools like automatic summarization of content to users. Automatically generated keywords are of great value to users. Such knowledge management features are absolutely necessary to give a user an overview of the content stored in the system.

Metadata is essential in digital libraries and enriches the stored information. It allows the user to extract more information out of a document collection. In J.UCS different user-defined attributes are used for the storage of metadata. A conversion to other formats (like Dublin Core) is easily possible.

Personalization tools (see e.g. chapter 5) supply a user with personal representations of a digital library. Different functions used in connection with content and even parts of content were discussed. In some available digital library systems personalization is hard to implement because the used platform does not support different authentication models. In J.UCS, the implementation effort will be very low because the used platform does support different users and groups.

Many functions (such as rearranging of content published in the collection, subscription to predefined categories etc.) will be available within the introduced personal workspace. Already implemented interactive features (such as an interactive annotation tool) will also be available to identified users.

Integration of different systems and exchange of content and metadata are key issues in the management component of a digital library. It is quite usual in the community to exchange metadata between different digital libraries. Metadata may be expressed in different formats. In J.UCS metadata is exchanged via automatically generated BibTeX-files to the Computer Science Bibliography located at University of Trier. Therefore it is possible for users of this bibliography to get information about articles published in J.UCS. Both readers and authors profit from this information exchange. Currently we investigate an exchange of metadata via the Open Archive Initiative.

It is possible to store (nearly) arbitrarily large digital objects in digital libraries. Information objects itself (“stand alone”) may be used in a narrow way: it is possible to search within such an information object, displaying this information in different ways etc. Nevertheless, highly sophisticated knowledge management tools add an additional layer to the information, so that it is possible to extract metadata out of the documents, creating automatic summaries etc. In this respect a digital library is more powerful than a traditional library. However, the real power of a digital library is the feature of *interconnecting different information entities*. It is possible to interlink documents in several electronic document formats (such as HTML, PDF, eBook,...). Nevertheless, extracting this information about relations between documents from the contents layer into the management layer, makes it possible to extract much more information out of a data collection stored in the digital library.

Electronic publishing in the scholarly community is very special in this respect. A single (peer-reviewed and accepted) paper about a specific area *does not* reflect that the author is an expert on this topic. Nevertheless, if the author wrote more than one paper related to this subject, it is very likely that he is an expert. In chapter 9 a system using this kind of information was introduced. Also, a tool to support the editorial team in selecting the most appropriate referees for submitted papers was introduced. In this application the tool will guarantee steady quality of publications in the journal.

This feature is currently under development but may be used in several other contexts. Finding an expert in a peer-reviewed journal is just one application of this idea. It may be extended and used in a research grant committee, automatic reference selection for scholarly papers, expert-search etc.

Content stored in a digital library may be accessed via different output devices. Different views on stored information must be provided by the system and must be available to the user. Access to contents must be as easy as possible for users. In the latter chapters a framework for the integration of different information systems was introduced. In chapter 11 an adaptive learning environment was described, where different systems including digital library systems are interconnected seamlessly. Functions as well as contents may be shared via different devices by different users. A more detailed discussion about the creation of such an environment was discussed in chapter 12.

It has been shown that the field of electronic publishing is very broad. Therefore many improvements are possible in existing systems. The next section will take a closer look at these improvements.

13.2 The Future of Digital Libraries

The last section of this dissertation tries to sketch the next steps in the development of an ideal digital library. This section will reflect on general features of electronic publishing systems and will not deal with very special features in special-purpose digital collections like audio- or video-collection. However, it will be possible to apply the following aspects to any digital library system of any type.

A few of the current research interests are enumerated in the first chapter. In the following subsections future aspects are discussed related to:

- Creation of digital content in a journal environment
- Dissemination of digitally stored content
- Usage of digital libraries.

13.2.1 Future Creation of Digital Content

As mentioned in section 1.2.1, several tools must be developed to create digital collections as easily and fast as possible without losing quality of the published content. There are different requirements for different kinds of digital libraries. Special purpose video-collections need different tools than systems for simple text-based content. Concepts of authoring (such as transclusions, see e.g. chapter 3) must be (and certainly will be) supported by electronic publishing systems.

In chapter 7 a simple approach for getting the right references for a scientific article is introduced. By submitting an outline of a paper, the system will find the right references. Therefore, the author of upcoming publications will know

who the “key-players” in the field are and which publications should be read *before* the publication is finished. There are many positive side-effects of such a feature. To mention just two of them: first, the workload of the author is reduced because irrelevant papers can be ignored and second, such a “recommender system” will increase the number of references to papers published in the system. Apparently, this feature will be implemented in the near future in every available digital library system.

Regardless of the type of content, the *publishing workflow* is very similar in different systems. When quality is an issue in the data collection some kind of quality control must be integrated in the publishing process. This process must be supported by the publishing workflow. In scholarly publishing environments a widespread concept of quality control is “peer-review”. However, the right referees must be found for a submission. Authors submit their contribution and the editorial team usually selects the appropriate referees for the submission. Thereafter the referees review the paper. The paper is then either accepted (with or without minor or major changes) or rejected. The comments of the referees are somehow shared with the author of the contribution (e.g. by email or with a system immanent cooperation module). This workflow and all attached processes have to be supported in many aspects by the publishing system.

Submission Process: It must be easy for an author to write and submit papers. Many publishing systems, if not all, allow users to upload their contributions via simple web-forms. Such a workflow-initiation includes tasks like account creation, information distribution, alerting etc.

Refereeing Process: Either the editorial team or the editors-in-chief select the right referees for an uploaded contribution. In chapter 9 a system supporting this task is introduced. Such a “referee-selection” tool will

reduce the workload of referees of a publication. This tool is currently under development. Referees are able to add comments to the submission and rate the contents.

Publishing Process: If a contribution is accepted by the referees and its final version is uploaded, some editorial work must be performed. Different kinds of metadata (descriptive, administrative and structural) must either be extracted or assigned, as well as the layout of the contribution must be edited to follow the publication standard. Therefore tools for adding headers and/or footers to the original contribution must be generated. Ideally the content is marked up in a user adaptable way, therefore the user may change the layout of the content related to the output device. Currently many stand-alone tools exist. In the future these tools will be seamlessly integrated into the publishing systems and it will be possible to interchange these tools.

The workflow described above must be available in every electronic publishing system. Aspects of identification and privacy must be considered.

Quality is a main issue in serious scientific publishing systems. Peer-review is just one example of quality control. Many other strategies are possible and some of them are described in [Arms, 2002]. In an electronic publishing environment, editorial work is the most expensive one. Therefore reducing the editorial effort will reduce costs. [Doyle, 2001] reported the costs for a published paper in the “American Physical Society” (APS):

Administrative costs of a single published article is about \$1,800. Unpublished articles are more expensive than published ones because of more appeals and cycles.

The workflow supporting a publishing process should reduce costs but will guarantee quality of the digital resources.

One big difference between digital and traditional content is the possibility to attach unprintable references or resources to the digital content. Such an unprintable reference may be an audio- or video resource, interactive software demos etc.

In a scholarly article usually many references are given. At the moment many publishers insert hyperlinks from references directly into an electronic representation of the reference. In many cases a hyperlink to the fulltext version stored on another server system is provided for the user. Nevertheless, references – especially references on the web – disappear over time. An experiment described in [Lawrence et al. 2001] analyzed URLs given in papers published between 1993 and 2000 in the area of computer science. In papers published in 1994 53% of the URLs were invalid. Concepts of Universal Resource Identifiers (URI, [Berners-Lee et al., 1998]), Digital Object Identifiers ([DOI, 2002]) or Persistent Uniform Resource Locators ([PURL, 2002] are already introduced but not widely available. Large publishing companies like Springer or Elsevier provide in their systems ([Link, 2002] and [Science Direct, 2002]) such DOIs for each published digital resource. Nevertheless, not every referenced entity has such a stable identifier. References to books where no electronic version is available to the reader should directly link either to the nearest public library of the user (or any other predefined library with an electronic cataloging system) and should also provide the users with a direct link to the book retailer of their choice. Many different systems will be integrated and system boundaries will disappear.

Therefore, a system must on the one hand make reading easier for the user and on the other hand support the author with highly sophisticated mechanisms to supply readers with additional resources.

Future systems will allow to attach multimedia entities either stored in the same publishing system or stored in any other system accessible via some protocol. If “attachments” are stored in some other server system the publishing system must check if the resource is still available. Since links are unidirectional this task must be scheduled by the publishing system. If the link destination is stable, no further action must be taken. However, if there are any changes in the destination or the destination disappears, the system must react accordingly.

Destination is unreachable: If a destination is temporarily unavailable, the system should hide the link. Additional information should be pointed out to the user. Continuous problems when reaching a destination should inform the system administrator or corresponding author of the paper. The concept of “robust linking” (see e.g. [Phelps and Wilensky, 2001]) adds a description of a destination contents to the link. Therefore it is possible to search for the destination via a search engine. This description (also known as “signature of a document”) is a key issue when trying to find an article again. Whenever a resource is unavailable over some time – this time must be specified by the administrator or author of the document – robust linking may find the destination again.

Changes in destination: Obviously, not every change in the target document will make the link invalid. However, if the content changes completely, it is very likely that the attachment does not support the reader of a publication in a way the author originally intended. [Francisco-Revilla et al., 2001] describes a system which automatically detects and

categorizes changes. If on the one hand there are changes in the layout of the destination document, then probably no further actions are necessary. On the other hand, major changes in the content should notify the administrator or corresponding author automatically. Please note that the system must detect changes in the structure as well. In learning environments it is necessary in some circumstances to react to this type of change.

In the future it will be possible for authors to attach arbitrary multimedia documents to their contributions. In J.UCS it is already possible to either upload such documents to the server system or to provide the user with a URL (in Hyperwave terminology this is a “remote object”) where further information is stored. The described tools will be available in the near future for remote documents.

13.2.2 Future Dissemination of Information

Currently information entities are stored either in a filesystem or in a database. Metadata, either in Dublin Core, Marc, etc., is available for most of the stored information entities. This information is largely disseminated via http-server systems, therefore it is easy for people to access information via simple web-browsers.

Most digital library systems are designed to provide information for *human users*. Nevertheless, digital library systems also want to collaborate together. Therefore content provided for users must be “machine-readable” and information must be *meaningful* to machines. Semantics must be added to data stored on web-servers. To quote [Berners-Lee et al., 2001]:

To date, the Web has developed most rapidly as a medium of documents for people rather than for data and information that can be processed automatically. The Semantic Web aims to make up for this.

What is true for the web in general is definitely also true for digital libraries. Even in metadata management systems keywords have different meanings (e.g. “creator”). Applications using digital libraries, such as the reference-linking described in the previous section, need the meaning of data. Automatic reference-linking would be much easier to implement if data were marked up with meaningful information. Two technologies which make the semantic web possible are already available and specified: XML and the Resource Description Framework ([RDF, 2002]).

With XML it is possible to define an arbitrary structure and to define tags without assigning meaning to them. RDF expresses meaning. Meaning is expressed in sets of triples: *subject*, *predicate* and *object*. These triples are encoded using XML. In RDF assertions about things having certain properties with certain values are expressed. *Subjects* as well as *objects* are each expressed via URIs (Universal Resource Identifier, [Berners-Lee et al., 1998]). Because of the open architecture of RDF it is also possible to define a URI for the *predicate*, therefore anyone can define a new concept.

At the time of writing many different payment methods are used. But what is the right price for an article? Should articles be cheaper if a certain amount of them are sold to readers? Should a reader pay in advance for a certain number of articles? Should users be allowed to buy single articles or just a certain collection of articles?

As mentioned before, the real administrative costs for a published article are about \$1.800.-. If the article is read just once, defacto price for the article is

\$1.800.- (probably nobody is willing to pay this amount of money just for one article). If 100 users read the article, the payment should be \$18.-. Nevertheless, it is at the risk of the publisher to assign the right price to an article or provide reasonable pricing for collections of articles. It is worth to notice that prices for subscribers are still too high. If prices would be much lower, many more potential readers would have the chance to read articles and all of the involved participants (i.e. publishers, readers, and authors) will profit from this policy.

In the future there will be a dynamic price schema for electronic content. The price of an article will be between \$X and \$Y, depending on overall customers of a specific article.

The number of users reading an article is one factor of price-building. However, many other aspects must be considered: e.g. quality and usage of the article. Quality may be pictured via the total rating of the readers community. On the one hand it is very likely that users are willing to pay more for a high-quality article than for a bad one. On the other hand high-quality articles are probably read by many people, therefore the price will decrease over time. Based on such a price calculation users will be honest regarding rating of content.

Usage of an article should also be considered for the calculated price paid by an individual user. Users select an article based on the ratings of other users and by the abstracts. Abstracts are usually presented to the reader free of charge. A user who discovered that he does not want to read the full article is not willing to pay the full price for it either. If, on the other hand, the user intensively works with the article (maybe even prints it) the article is certainly worth the price. Nevertheless, it is not possible to track the user for printing activities or reading the whole article due to the client-server concept of the web. Therefore a very flexible billing system must be developed, where user's

feedback is taken into account. This billing system will depend on the honesty of users. Someone may argue that users are not honest. This might be true, however, many other systems work very well based on the assumption that users *are* honest and do contribute in a candid way. An open editing system for web-sites – Wiki (see e.g. [Wiki, 2002]) – is just one example of a running system.

13.2.3 Future Usage of Digital Libraries

Different features will be implemented in digital library systems of the future. These features will be interchangeable throughout the systems and easily adapted by the user. Different distributed systems will appear to the users as one single system with one-time identification and one “look-and-feel”.

The system will monitor many user-activities to improve the service of the library. User feedback will be possible and also be taken into account in many areas. Building of communities will be supported.

A primary issue for publishers is to find users who find the service and content valuable, who are able to contribute ideas and influence upcoming articles. Therefore it is necessary to evaluate systems usage and react correspondingly. In the following some ideas on how to evaluate usage statistics are presented. A digital journal environment is considered in the discussion. However, the results may apply to any other environment.

To extract information out of log-files it is necessary to identify *which information* is consumed by *which reader*. Standard log-files do provide this information. It is just a matter of evaluation to get the *right* information out of the log-files. The collected and evaluated information must be provided for users, i.e. if articles are consumed frequently, this information must be given

to other users. Articles viewed by a certain number of users and which received a certain user-rating during a time-range should be considered for a *best paper award*.

Ratings of users should not just reflect on the specific article itself. All involved parties, including authors, referees who reviewed the article and even the user who rated it, must be included in these ratings. Therefore it is very likely that they are all interested to deliver the best possible work. Small *digital acknowledgments* (such as logos reflecting reputation, web-space, upload area etc.) must honor the involved persons.

An author who wrote highly rated articles should actively be invited by the editors of a journal to write new articles. All articles written by certain authors must be marked with appropriate signs to visualize the rating of the author. If certain issues of papers (or any other collections of papers like categories) have high ratings, the issue should be rated with high marks. Editors of special issues – i.e. guest editors – should also benefit from high quality papers published in “their” special issues.

Referees who reviewed highly rated articles should also be rated highly. Over time the reputation of a referee will increase and the editors-in-chief will be automatically notified by the system about this fact. On the other hand referees who have reviewed *bad articles* (whatever metric is used to define *bad*, *good* or *best*...) should be revised by the editors.

Article ratings as well as annotations provided by the users should rate the user itself. If a user answers many questions via an annotation feature and this answer is helpful to other users, the system should acknowledge this circumstance. Many discussion forums acknowledge the work of users via simple symbols attached to the user-name. They work with different digital entities, but the ideas are very similar: They all try to receive the best work from all

participants. In the context of a digital library, there are much more involved persons than in simple discussion forums: the authors of the articles, the referees, the (guest-)editors and the reader community. All of them should profit from one another.

References

- [ACM Digital Library, 2002] ACM Digital Library (2002). <http://www.acm.org/dl>.
- [Arms, 2002] Arms, W. (2002) What Are the Alternatives to Peer Review? Quality Control in Scholarly Publishing on the Web. *The Journal of Electronic Publishing*, Vol. 8, Issue 1
- [Berners-Lee et al., 1998] Berners-Lee, T., Fielding, R., Irvine, U., and Masinter, L. (1998). Uniform Resource Identifiers (URI). RFC 2396.
- [Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., and Lassila O. (2001) The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities http://www.scientificamerican.com/print_version.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21
- [DOI, 2002] DOI (2002). Digital Object Identifier. <http://www.doi.org>.
- [Doyle, 2001] Doyle, M. (2001). Peer review's future in a world of Open Archives Presentation at the Workshop on OAI and Peer Review Journals in Europe CERN, Geneva, March 22-24, 2001.
- [Francisco-Revilla et al., 2001] Francisco-Revilla, L., Shipman, F., Furuta, R., Karadkar, U., and Arora, A. (2001). Managing change on the web. In *Proceedings of the first ACM/IEEE-CS joint conference on Digital libraries*, pages 67–76. ACM Press.

- [Lawrence et al. 2001] Lawrence, S.; Pennock, D.M.; Flake, G.W.; Krovetz, R.; Coetzee, F.M.; Glover, E.; Nielsen, F.A.; Kruger, A.; Giles, C.L. Persistence of Web References in Scientific Research Computer, February 2001, Vol. 34, No. 2, 2001 <http://computer.org/computer/co2001/r2026abs.htm>
- [Link, 2002] Springer Link (2002). Link Service. <http://link.springer.de>.
- [Phelps and Wilensky, 2001] Phelps, T. and Wilensky, R. (2001). Robust Linking: Robust Hyperlinks and Robust Locations Workshop Proceedings 'First DELOS International Summer School an Digital Library Technologies' Pisa, July 2001
- [PURL, 2002] Persistent Uniform Resource Locator OCLC Online Computer Library Center <http://purl.oclc.org>
- [RDF, 2002] Resource Description Framework (RDF) <http://www.w3c.org/RDF>
- [Science Direct, 2002] Science Direct (2002). Science Direct. <http://www.sciencedirect.com>.
- [Wiki, 2002] Wiki: Open Web Site Editing Tool (2002) <http://www.wiki.org>

Chapter 14

References Overview

14.1 Contributions of the Author

[Dallermassl et al., 2000a] Dallermassl, C., Haub, H., Krottmaier, H., Schmaranz, K., and Zambelli, P. (2000a). Adaptive Learning Environment Framework. In *Proceedings of the International Conference on Computers in Education/International Conference on Computer-Assisted Instruction (ICCE/ICCAI 2000)*, Taipei, Taiwan.

[Dallermassl et al., 2000b] Dallermassl, C., Haub, H., Krottmaier, H., Schmaranz, K., and Zambelli, P. (2000b). Using Highly Sophisticated Middleware For Building Arbitrarily Distributed Teaching Environments. In *Proceedings of the International Conference on Computers in Education/International Conference on Computer-Assisted Instruction (ICCE/ICCAI 2000)*, Taipei, Taiwan.

- [Gütl et al., 1998] Gütl, C., Zwantschko, B., Götzinger, W., and Krottmaier, H. (1998). Managing and Archiving Digital Audio Data in Internet and Extranet. In *20. International Convention on Sound Design (ICSD 1998)*, pages 1116–1136, Karlsruhe.
- [Krottmaier, 2001] Krottmaier, H. (2001). Improving the Usability of a Digital Library. In Hübler, A., Linde, P., and Smith, J. W., editors, *Electronic Publishing (ELPUB 2001)*, pages 178–182, Canterbury, Kent, United Kingdom. International Council for Computer Communication (ICCC) and International Federation for Information Processing (IFIP).
- [Krottmaier, 2002a] Krottmaier, H. (2002a). Automatic References: Active Support for Scientists in Digital Libraries. In Lim, E.-P., Foo, S., and Khoo, C., editors, *Digital Libraries: People, Knowledge & Technology: Proceedings of the 5th International Conference on Asian Digital Libraries (ICADL 2002)*. Springer.
- [Krottmaier, 2002b] Krottmaier, H. (2002b). Automatic Support in the Review Process. In Far, B. H., Shafazand, H., Takizawa, M., and Wagner, R., editors, *Proceedings of the Workshops of “The First Eurasian Conference on Advances in Information and Communication Technology” (EurAsia-ICT 2002)*, pages 467–471. Österreichische Computer Gesellschaft.
- [Krottmaier, 2002c] Krottmaier, H. (2002c). Current and Future Features of Digital Journals. In Shafazand, H. and Tjoa, A. M., editors, *Proceedings of “The First Eurasian Conference on Advances in Information and Communication Technology” (EurAsia-ICT 2002)*, LNCS 2510. Springer Verlag.

- [Krottmaier, 2002d] Krottmaier, H. (2002d). Transcluded Documents: Advantages of Reusing Document Fragments. In Carvalho, J. A., Hübler, A., and Baptista, A. A., editors, *Advances in Media Technology (ELPUB 2002)*, pages 359–367. International Council for Computer Communication (ICCC) and International Federation for Information Processing (IFIP).
- [Krottmaier, 2003] Krottmaier, H. (2003). Enhanced Annotations. In *Proceedings of International Conference on Society for Information Technology and Teacher Education (SITE 2003)*.
- [Krottmaier and Helic, 2002a] Krottmaier, H. and Helic, D. (2002a). Issues of Transclusions. In *Proceedings of E-Learn (E-Learn 2002)*, pages 1730–1733, Montreal, Canada.
- [Krottmaier and Helic, 2002b] Krottmaier, H. and Helic, D. (2002b). More than Passive Reading: Interactive Features in Digital Libraries. In *Proceedings of E-Learn (E-Learn 2002)*, pages 1734–1737, Montreal, Canada.
- [Krottmaier and Maurer, 2001] Krottmaier, H. and Maurer, H. (2001). Transclusions in the 21st Century. *Journal of Universal Computer Science*, 7(12):1125–1136. http://www.jucs.org/jucs_7_12/transclusions_in_the_21st.

14.2 Further Readings in Digital Libraries

- [ACM, 1998] ACM (1998). ACM Computing Classification System.
- [Adobe, 1989] Adobe (1989). *PostScript Programmier-techniken*. Addison-Wesley Longman, Inc.

- [Adobe, 1993] Adobe (1993). *Portable Document Format Reference Manual*. Addison-Wesley Longman, Inc.
- [Arms, 2000] Arms, W. (2000). *Digital Libraries*. The MIT Press.
- [Autonomy, 2002] Autonomy (2002). <http://www.autonomy.com>.
- [Bannan, 1997] Bannan (1997). *Intranet Document Management*. Addison-Wesley Longman, Inc.
- [Baubin, 1996] Baubin, T., editor (1996). *Electronic Publishing: Strategische Entwicklungen für die Europäische Verlagsindustrie im Hinblick auf das Jahr 2000*. Europäische Kommission.
- [Berk and Devlin, 1991] Berk and Devlin, editors (1991). *Hypertext / Hypermedia Handbook*. Software Engineering Series. McGraw-Hill Book Company.
- [Berners-Lee et al., 1996] Berners-Lee, T., Fielding, R., and Frystyk, H. (1996). Hypertext Transfer Protocol – HTTP/1.0.
- [Berners-Lee et al., 1998] Berners-Lee, T., Fielding, R., Irvine, U., and Masinter, L. (1998). Uniform Resource Identifiers (URI). RFC 2396.
- [Berners-Lee et al., 1994] Berners-Lee, T., Masinter, L., and McCahill, M. (1994). Uniform Resource Locators. <http://www.w3.org/Addressing/rfc1738.txt>.
- [BMM3, 2000] BMM3 (2000). Brockhaus Multimedia. <http://www.brockhaus-multimedia.de>.
- [Borgman, 2000] Borgman, C. L. (2000). *From Gutenberg to the Global Information Infrastructure*. The MIT Press.
- [Bush, 1945] Bush, V. (1945). As we may think. *The Atlantic Monthly*, 176(1):101–108. <http://www.theatlantic.com/unbound/flashbks/computer/bushf.htm>.

- [Calude et al., 1994] Calude, C., Maurer, H., and Salomaa, A. (1994). Journal of Universal Computer Science. *Journal of Universal Computer Science*, 0(0):109–115. http://www.jucs.org/jucs_0_0/journal_of_universal_computer.
- [Card et al., 1999] Card, S. K., Mackinlay, J. D., and Shneiderman, B., editors (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers.
- [Chien et al., 2001] Chien, S.-Y., Tsotras, V. J., and Zaniolo, C. (2001). Xml document versioning. *SIGMOD Record*, 30(3):46–53.
- [Clark and DeRose, 1999] Clark, J. and DeRose, S. (1999). XML Path Language (XPath) Version 1.0. <http://www.w3.org/TR/1999/REC-xpath-19991116>.
- [Clarke, 1997] Clarke, R. (1997). Beyond the Dublin Core: Rich Meta-Data and Convenience-of-Use Are Compatible After All. Technical report, Xamax Consultancy Pty Ltd, Canberra. <http://www.anu.edu.au/people/Roger.Clarke/II/DublinCore.html>.
- [CVS-HOME, 2001] CVS-HOME (2001). Concurrent versions system homepage. <http://www.cvshome.org>.
- [DeRose et al., 2001a] DeRose, S., Maler, E., and Daniel Jr., R. (2001a). XML Pointer Language (XPointer) Version 1.0. <http://www.w3.org/TR/2001/CR-xptr-20010911>.
- [DeRose et al., 2001b] DeRose, S., Maler, E., and Orchard, D. (2001b). XML Linking Language (xlink) Version 1.0. <http://www.w3.org/TR/xlink>.
- [DOI, 2002] DOI (2002). Digital Object Identifier. <http://www.doi.org>.
- [Dublin Core Metadata Initiative, 1999] Dublin Core Metadata Initiative (1999). Dublin Core Metadata Element Set Version 1.1. Technical report, Dublin Core Metadata Initiative. <http://purl.oclc.org/dc/documents/rec-dces-19990702.htm>.

- [Eckstein, 1999] Eckstein, R. (1999). *XML Pocket Reference*. O'Reilly Associates, Inc.
- [Fayyadmou, 1996] Fayyadmou, U. M., editor (1996). *Advances in knowledge discovery and data mining*. The MIT Press.
- [Fellner and Zens, 2000] Fellner, D. and Zens, M. (2000). Electronic Submission, Managing and Approval of Grant Proposals at the German Research Foundation based on Standard Internet and Office Tools. *Journal of Universal Computer Science*, 6(3):356–366. http://www.jucs.org/jucs_6_3/electronic_submission_managing_and.
- [Fielding et al., 1999] Fielding, R., Gettys, J., and Mogul, J. (1999). Hypertext Transfer Protocol – HTTP/1.1.
- [Frank, 2002] Frank, E. (2002). Kea: automatic keyphrase extraction.
- [Gauss, 1995] Gauss, W. (1995). *Dokumentations- und Ordnungslehre*. Springer Verlag, 2 edition.
- [Haeberli, 1996] Haeberli, P. (1996). Merge: A General Transclusion Facility. Technical report, Silicon Graphics Computer Systems.
- [Halvorsen et al., 1999] Halvorsen, P., Lund, K., Goebel, V., Plagemann, T., Preuss, T., and Koenig, H. (1999). ConfMan: Integrated WWW and DBS Support for Conference Organization.
- [Henke, 2001] Henke, H. (2001). *Electronic Books and ePublishing – A Practical Guide for Authors*. Springer Verlag.
- [Huck et al., 1998] Huck, G., Fankhauser, P., Aberer, K., and Neuhold, E. (1998). JEDI: Extracting and Synthesizing Information from the Web. In *Cooperative Information Systems (COOPIS)*.
- [Hyperwave, 2001] Hyperwave (2001). Hyperwave Information Server. <http://www.hyperwave.com>.
- [IEEE-xplore, 2002] IEEE-xplore (2002). Digital Library of the IEEE. <http://ieeexplore.ieee.org>.
- [IICM, 2002] IICM (2002). <http://www.iicm.edu>.

- [JOANNEUM RESEARCH, 2002] JOANNEUM RESEARCH (2002). <http://www.joanneum.ac.at>.
- [JODI, 2002] JODI (2002). Journal of Digital Information. <http://jodi.ecs.soton.ac.uk>.
- [J.UCS, 2002] J.UCS (2002). Journal of Universal Computer Science. <http://www.jucs.org>.
- [Sparck Jones and Willett, 1997] Sparck Jones, K., Willet, P., editor (1997). *Readings in Information Retrieval*. Morgan Kaufmann Publishers.
- [Khare, 1998] Khare, R. (1998). Requirements for interactive access to html and xml documents. <http://www.w3.org/MarkUp/future/papers/rkhare-1998-501.html>.
- [Kipphan, 2001] Kipphan, H. (2001). *Handbook of Print Media – Technologies and Production Methods*. Springer Verlag.
- [Know-Center, 2002] Know-Center (2002). <http://www.know-center.at>.
- [Kopka, 1994] Kopka, H. (1994). *LaTeX Einführung*, volume 1. Addison-Wesley Longman, Inc.
- [Kunze, 1995] Kunze, J. (1995). Functional recommendations for internet resource locators, network working group rfc 1736. <ftp://ds.internic.net/rfc/rfc1736.txt>.
- [Laender et al., 2002] Laender, A., Ribeiro-Neto, B., da Silva, A., and Teixeira, J. (2002). A brief survey of web data extraction tools. *Sigmod Record*, 31(2).
- [Lamport, 1995] Lamport, L. (1995). *Das LaTeX Handbuch*. Addison-Wesley Longman, Inc.
- [Lamport, 1999] Lamport, L. (1999). Home page. http://www.research.digital.com/SRC/personal/Leslie_Lamport/home.html.
- [Lawrence et al., 1999] Lawrence, S., Giles, C. L., and Bollacker, K. (1999). Digital libraries and autonomous citation indexing. *IEEE Computer*, 32(6):67–71.

- [Ley, 2002] Ley, M. (2002). Computer science bibliography. <http://dblp.uni-trier.de>.
- [Li and An, 2001] Li, X. and An, J. (2001). Cernet infrastructure and digital library challenges. In Chih Chen, C., editor, *Global Digital Library Development in the New Millennium*, pages 153–158.
- [LINK, 2002] LINK (2002). Link Service. <http://link.springer.de>.
- [Marcu, 2000] Marcu, D. (2000). *The Theory and Practice of Discourse Parsing and Summarization*. The MIT Press.
- [Marsh and Orchard, 2001] Marsh, J. and Orchard, D. (2001). XML Inclusions (XInclude) Version 1.0. <http://www.w3.org/TR/2001/WD-xinclude-20010516>.
- [Maurer, 1996] Maurer, H. (1996). Hyper-G now Hyperwave — The Next Generation Web Solution.
- [Maurer, 2001] Maurer, H. (2001). Beyond classical digital libraries. In Chih Chen, C., editor, *Global Digital Library Development in the New Millennium*, pages 165–173.
- [Maurer and Schmaranz, 1994] Maurer, H. and Schmaranz, K. (1994). J.UCS - The Next Generation in Electronic Journal Publishing. *Journal of Universal Computer Science*, 0(0):117–125. http://www.jucs.org/jucs_0_0/j_ucs_the_next.
- [McGilton and Campione, 1990] McGilton, H. and Campione, M. (1990). *PostScript by Example*. Addison-Wesley Longman, Inc.
- [Meadow, 1992] Meadow, C. (1992). *Text Information Retrieval Systems*. Academic Press, Inc.
- [Miller, 1996] Miller, P. (1996). Metadata for the masses. *Ariadne*. <http://www.ariadne.ac.uk/issue5/metadata-masses/>.
- [Moats, 1997] Moats, R. (1997). *URN Syntax*. RFC 2141.
- [Morris, 2000] Morris, T. (2000). *Multimedia Systems: Delivering, Generating and Interacting with Multimedia*. Springer Verlag.

- [NEC Research Institute, 2002] NEC Research Institute (2002). Researchindex. <http://www.researchindex.com>.
- [Nelson, 1987] Nelson, T. (1987). *Literary Machines*. Ted Nelson.
- [Nelson, 1995] Nelson, T. (1995). The Heart of Connection: Hypermedia Unified Transclusion. *Communications of the ACM*, 38:31–33.
- [Nelson, 1996] Nelson, T. (1996). Generalized Links, Micropayment and Transcopyright. <http://www.almaden.ibm.com/almaden/npuc97/1996/tnelson.htm>.
- [Odlyzko, 1994] Odlyzko, A. (1994). Tragic Loss or Good Riddance? The Impending Demise of Traditional Scholarly Journals. *Journal of Universal Computer Science*, 0(0):3–52. http://www.jucs.org/jucs_0_0/tragic_loss_or_good.
- [Pam, 1996] Pam, A. (1996). Methods for implementing transclusion of text into HTML pages. Technical report, Xanadu Australia.
- [Poon and Kontogiannis, 2001] Poon, F. and Kontogiannis, K. (2001). i-Cube: A Tool-Set for the Dynamic Extraction and Integration of Web Data. *LNCS 2040*, pages 98–115.
- [Raggett et al., 1999] Raggett, D., Hors, A. L., and Jacobs, I. (1999). HTML 4.01 Specification, W3C Recommendation. <http://www.w3.org/TR/html4>.
- [Rivest, 1992] Rivest, R. (1992). The md5 message-digest algorithm.
- [Sahuguet and Azavant, 1999] Sahuguet, A. and Azavant, F. (1999). Wysiwyg web wrapper factory. <http://db.cis.upenn.edu/cgi-bin/Person.perl?sahuguet>
- [Schmaranz, 1998] Schmaranz, K. (1998). Aspects of Electronic Publishing in Hypermedia Systems. Dissertation at Graz, University of Technology. June 1998.

- [Schwabe et al., 2001] Schwabe, G., Streitz, N., and Unland, R., editors (2001). *CSCW-Kompendium: Lehr- und Handbuch zum computerunterstützten kooperativen Arbeiten*. Springer Verlag.
- [Science Direct, 2002] Elsevier (2002). Science Direct. <http://www.sciencedirect.com>.
- [Springer Pub. Co., 2002] Springer Pub. Co. (2002). <http://www.springer.de>.
- [Strong, 1999] Strong, W. S. (1999). *The copyright book: a practical guide*. The MIT Press.
- [Systems, 2001] Systems, A., editor (2001). *PDF reference : Adobe portable document format version 1.4*. Addison-Wesley Longman, Inc.
- [Tochtermann, 2002] Tochtermann, K. (2002). Personalisierung im Kontext von Digitalen Bibliotheken und Wissensmanagement Habilitationsschrift, Graz, University of Technology. March 2002.
- [Verity, 2002] Verity (2002). <http://www.verity.com>.
- [W3C, 1998] W3C (1998). Document Object Model (DOM) Level 1 Specification. Technical report, W3C. <http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001>.
- [W3C, 2000] W3C (2000). Extensible Markup Language (XML) 1.0 (Second Edition). <http://www.w3.org/TR/2000/REC-xml-20001006>.
- [W3C, 2001a] W3C (2001a). HyperText Markup Language. <http://www.w3.org/MarkUp>.
- [W3C, 2001b] W3C (2001b). Micropayments Markup Working Group. <http://www.w3.org/ECommerce/Micropayments>.
- [W3C, 2002] W3C (2002). Extensible Stylesheet Language. <http://www.w3.org/Style/XSL>.
- [Weibel et al., 1998] Weibel, S., Kunze, J., Lagoze, C., and Wolf, M. (1998). Dublin Core Metadata for Resource Discovery. RFC 2413.

[WIKI, 2001] WIKI (2001). Usemodwiki. <http://www.usemod.com/cgi-bin/wiki.pl>.

List of Figures and Tables

List of Figures

2.1	List of Journals in Science Direct	32
2.2	Browsing in J.UCS	33
2.3	Bibliographic Search in Springer-Link	36
2.4	ACM-DL: GUI for the creation of a personal binder	39
4.1	Compound document: different visualizations depending on user- preferences	77
9.1	Current Paper Submission	133
9.2	Information about a Referee	136
11.1	Adaptive Learning Environment Framework	179

List of Tables

2.1	Browsing the Digital Library: Overview of Indexpages	34
-----	--	----