Aging Links

Claudia Niederée, Ulrike Steffens,
Joachim W. Schmidt, and Florian Matthes

Software Systems Institute,
Technical University Hamburg-Harburg,
D-21073 Hamburg, Germany
{c.niederee,ul.steffens,j.w.schmidt,f.matthes}@tu-harburg.de
http://www.sts.tu-harburg.de

Abstract Rooted in the principle of hypertext, linked information is almost ubiquitous due to the WWW and related services. Links between information objects are established for various reasons, aspects of which are encoded by a link type or expressed through link context, e.g., by the surrounding content. Such reasons may lose their validity through content evolution in the link target. Fine-grained solutions are required that enable the user to gain evolution awareness without being distracted from his main task.
In this paper we present *aging links* as a non-intrusive mechanism for improving awareness in cooperative work with linked information networks. A link may age, affected by the evolution of the link target, leading to a gradual loss of its validity. The aging process is driven by evolution-indicating events and may be flexibly controlled by link type specific aging strategies. A customizable service, *EvEnAge*, based on standard technologies, prototypically implements the concept of aging links for XML documents.

1 Introduction

In selecting, collecting, and structuring relevant information objects from the universal information space, cooperative work with linked information is an important activity in modern information environments such as digital libraries, intranets, and portals. Linked information networks are subject to permanent content evolution where the user community is strongly involved in such changes.

The rich semantics of linking is based on the manifold reasons for link establishment and the high degree of autonomy on both sides, the link target with its autonomous content evolution and the link source with its individual content use. If the reasons for linking are made explicit in the context of the link, e.g. by the surrounding text, the evolution in the link target may corrupt the correctness of the information object containing the link source. Thus, link validity depends greatly upon the rich and varying link semantics. Fine-grained solutions are required to enable users to gain awareness [5] of the evolution-related aspects of link validity without being hindered in using linked information networks.

For this purpose, we propose the concept of aging links. Increasing ages are assigned to links reflecting the effects of evolution in the information object targeted by the link. The aging process is

- driven by evolution-indicating events like link target version creation;
- controlled by link type specific aging strategies influencing the speed of aging;
- consolidated by the introduction of common age levels that create a uniform notion of age and facilitate a comparison of link ages.

In this paper we take the position that link aging contributes to evolution awareness by visualizing the evolution of linked information objects and by triggering cooperation- and evolution-related actions on the basis of link ages.

The paper is structured as follows: Section 2 motivates our approach by discussing the benefits of link aging and its exploitation in different cooperative scenarios. Section 3 introduces the concept of aging links. It describes the aging process based on evolution-indicating events that drive this process. *EvEnAge*, a web-based service implementing aging links, is presented in section 4. The paper concludes with a discussion of related work and future research directions.

2 Evolution in Cooperative Link Scenarios

Being part of an information network the modification of an information object such as a text document, an image, etc., may not be viewed in isolation but also its impact on the objects by which it is referenced. We propose the concept of *aging links* to cope with this situation. Driven by the evolution of the target object, a link ages, thereby reflecting a (possible) decrease in the validity of the link. Link aging contributes to evolution awareness in a wide variety of cooperation situations. In the following sections, a web-based competence center is used to illustrate and discuss the benefits of link aging.

Link Aging in Loose Cooperation Scenarios A web-based competence center is established to manage and distribute knowledge among the members of an organization or a community. Typically, it contains numerous annotated links to related information objects from different providers. This may for example be a link to a standard specification (e.g of XML) augmented with a comment on its relevance. The validity of such annotated links is often highly evolution dependent. The annotation on an external JDBC user guide may state that it does not cover JDBC 2. A new version of the guide may introduce this feature and invalidate the annotation. This decrease in validity can be reflected by aging the respective link.

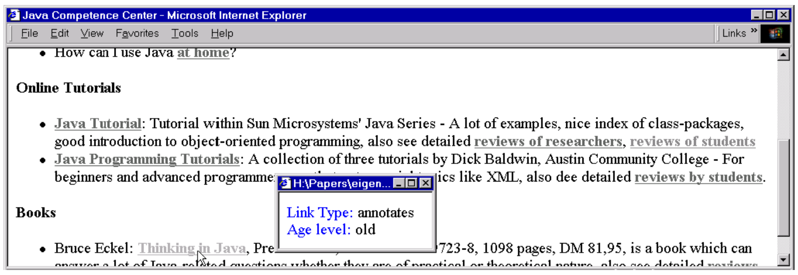
Due to the distributed situation in this loose cooperation scenario, link aging requires evolution protocols between the competence center that acts as an information consumer and the external information providers.

Link Aging in Close Cooperation Scenarios In the process of cooperatively constructing the information artifact "competence center", information objects are collected, composed, classified, and linked to internal as well as external resources.

Parallel work is often unavoidable when teams work creatively with information. Two groups may, for example, work on an introduction to Java programming in parallel, both tackling the problem from different backgrounds. The results of such parallel work are later merged or kept as alternatives for a different readership. A link stating that some information object is an alternative to another one is rather stable with respect to changes in the objects. An adequately customized and type-dependent aging strategy reflects this fact.

Working with two copies of the same information object a link may, on the other hand, connect the copy with its master. Link aging informs the owner of the copy about evolution in the master. Here, different degrees of autonomy may be implemented via customized aging strategies.

Exploiting Link Age The competence center team may react to aging links by rechecking the referenced information objects and, e.g., adapting the link context to the changed link target. The support of different age levels refines the awareness mechanism: A middle-aged link may still be left alone for a while, whereas a very old link requires action, for example in the form of re-reading the changed information object. A user may choose to


Java Competence Center - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links

- How can I use Java at home?

Online Tutorials

- Java Tutorial: Tutorial within Sun Microsystems' Java Series - A lot of examples, nice index of class-packages, good introduction to object-oriented programming, also see detailed reviews of researchers, reviews of students
- Java Programming Tutorials: A collection of three tutorials by Dick Baldwin, Austin Community College - For beginners and advanced programme ics like XML, also dee detailed reviews by students.

Books

- Bruce Eckel: Thinking in Java, Pre 723-8, 1098 pages, DM 81,95, is a book which can

Link Type: annotates
Age level: old

Figure 1. Rendering Aging Links

be informed about the reaching of a new age level via a changed visualization or an explicit notification. Figure 1 shows the visualization of link aging via fading colors. Further possible exploitations of link age include the retrieval of links that are above a certain age level, e.g. to recheck them, or the automatic deactivation of links exceeding a certain threshold value.

3 The Concept of Aging Links

The metaphor of *aging* is used to reflect the effects of target evolution on a link: A mechanical link ages over time either by its use or non-use, leading to a gradual loss of its functionality, similarly a digital link ages, affected by the evolution of the link target, leading to a gradual loss of its validity.

3.1 The Aging Process

Links in our approach are subject to an *aging process*. This process is driven by rated events indicating evolution in the link target: Each link is associated with a value that reflects its age. This value is set to zero upon creation of the link. Whenever a new version of the link target is created or another evolution-indicating event occurs, the link age changes to reflect the new situation. Update-in-place semantics is assumed here. This may also be achieved by an indirection (see e.g. PURLs [17]).

Aging is considered for typed directed links between information objects. As discussed in [8], typed links provide added value in information networks e.g. by enabling machine supported, link type specific analysis and actions. Exploiting link type specific evolution dependencies empowers a fine-grained control of the aging process.

Age comparability is an important issue for the evaluation of link ages by the user. Unfortunately, links of distinct types differ in their aging characteristics impeding the direct comparison of ages. In order to create a *common* notion of age, so called *age levels* have been introduced together with link type specific mappings from concrete ages to these age levels. This idea was inspired by the use of the terms “old” and “young” in nature: A tree is still called “young” when it is 15 years old whereas a dog aged 15 is already an old dog.

A link traverses a sequence of *age levels* that partition its lifetime into phases of assumed validity. Links in the same age level can be expected to be in roughly the same state of validity, although their targets possibly have been subject to a considerably different degree of evolution. Age levels are associated with type specific threshold values. If the link age reaches a threshold value, the link transits to the associated age level. This transition is exploited to trigger evolution-related actions like changing link visualization or notifying interested cooperation partners.

An example set of age levels is: *up-to-date*, *very young*, *young*, *middle*, *old*, *very old*, and *outdated*. Optionally, a threshold value for the invalidity of a link may be defined: When the link reaches the associated age it is deactivated.

The aging process is controlled by an aging strategy that influences the speed as well as the impact of aging. This strategy determines several parameters of the process, i.e. the supported age levels, their threshold values, and the actions triggered by the reaching of an age level. This enables the adaptation of the aging process to the characteristics of different link types and applications.

Each link is associated with an aging strategy. Three levels of strategies are available: A *general strategy* provides a default setting. Link type specific evolution dependency, as it is discussed in section 3.4, is captured by *link type specific strategie*s. The definition of *individual strategies* copes with the semantics of special links. This approach combines user comfort with high flexibility.

3.2 Computing Ages and Age Levels

As part of the aging process an age function computes the new link age. It is the difficult task of this function to judge to which degree the evolution in the target information object has affected the validity of the link taking into account the reason for its establishment. This requirement raises three questions:

What are the metrics for link aging? The aging of a link does not depend directly upon time but is triggered by evolution in the link target. A straightforward approach is aging a link by one unit on each new version of the target object. However, this equivocal treatment fails to differentiate between versions introducing only minor changes and those substantially altering the target object. In our approach, evolution is rated with a value of type real that is used to compute the link age. This also enables aging process variants (see section 3.3).

How to handle and rate evolution? Evolution that effects the validity of the link may be based on changes in the link target as well as on other changes in the environment of the information network, e.g. the elapsing of a certain amount of time. In our model we rely on events that indicate that some kind of evolution has taken place in the target information object. The rating of the *evolution-indicating events* is based on event classification.

How to take reasons for link establishment into account? The reasons for link establishment are only partly made explicit in the link context, e.g. by surrounding text. Computing the effect of the evolution on the link from this text is forbiddingly complicated because of the high freedom in its composition. Therefore, this aspect is approximated by link type dependent heuristics (see section 3.4). Link types code part of the reason for link establishment.

3.3 Classification of Evolution-Indicating Events

The concept of link aging is generalized by introducing the notion of an *evolution-indicating event*, of which version creation is only a special case. Further types of evolution-indicating events lead to several variants of the aging process.

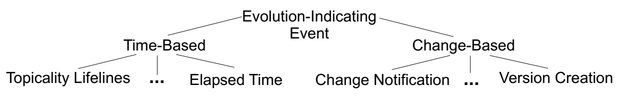


Figure 2. Classification of Evolution-Indicating Events

According to the classification in figure 2, evolution-indicating events are divided into change- and time-based events. Change-based events may be generated by version creation but may also rely on general change notification. In distributed environments such change notification may be provided by the server managing the target object or by a third-party service like Mind-it [12] that monitors the changes in a wider information space. In contrast, time-based events are triggered when a certain amount of time has elapsed. The impact of an evolution-indicating event on aging may be refined by further classification or parameterizing of the events. Viewing the aging process according to this model allows a smooth integration of time-driven with evolution-driven aging.

A fine-grained rating of events is achieved if versions are classified according to the degree of change they introduce. This can either be performed manually or automatically by difference computation based for example on change detection mechanisms [2] or the domain-based comparison of semistructured documents [10].

The topicality of an information object may vary over its lifetime. By defining "topicality lifelines" for information objects this phenomenon may be exploited as a further source of time-based link aging. For a discussion of this idea see [13].

3.4 Link Type Specific Evolution Dependencies

Link types imply a specific semantic relationship between the source and the target information object of the link. As a consequence, links of some types tend to be more evolution-sensitive than others.

Table 1. Link Type Specific Evolution Dependency

Link Type	**Evolution Dependency**	**Notes**
annotates	high	For details see Sect.2;
origin	middle to high	To avoid high divergence between master and copy, evolution dependency is generally defined as middle to high. It may be reduced to increase autonomy.
child	middle to high	The relationship between parent and child concepts is rather strong resulting in a considerable impact of child evolution on this relationship.
alternative	low	For details see Sect.2;
instance	low	A subject of an information object (in our case the basis for classification) is a rather stable property. It is only affected by (large) topic-influencing changes.

A wide variety of link types is possible in information networks [22]. Our choice of five different link types for a closer examination of their evolution dependency was inspired by the experience from two Digital Library projects (WEL [19], Kolibri [18]). Special focus is given to subject-oriented classification hierarchies and to support for cooperation, personalization, and autonomy [16].

Considering the classifiers as information objects of their own, links of type *child* are used to represent classification hierarchies (parent $\succ$ child) and links of type *instance* express the classification of information objects (classifier $\succ$ classified object). A basis for personalization and cooperation support is provided by links of type *annotates* (annotation $\succ$ annotated object) and by links of type *alternative* that avoid premature establishment of consent. A link of type *origin* is used to connect the copy of an information object to its original (copy $\succ$ original).

Table 3.4 summarizes the evolution dependency trends observed for the application domains examined. Deviating evolution dependencies may be found in other domains as well as for individual links requiring an adaptation of aging strategies.

4 ***EvEnAge***: An XML-Based Service for Link Aging

The concept of aging links has been implemented by the prototypical aging service *EvEnAge*. This service focuses on web-based applications and enables the aging of links embedded in XML documents. The *EvEnAge* implementation relies on standard technology and is open for integration with existing services and tools.

4.1 The *EvEnAge* Service Architecture

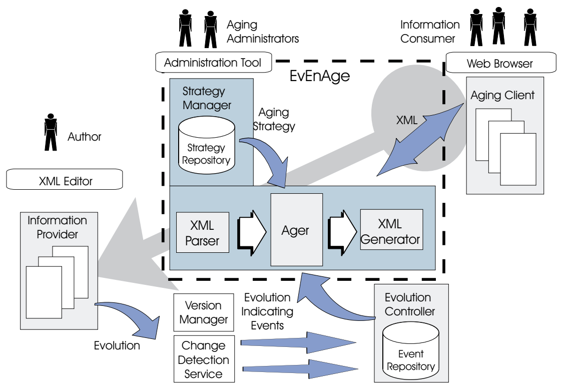


Figure 3. The *EvEnAge* Service Architecture

The architecture of *EvEnAge* mainly consists of two components (see Fig. 3): The *Ager*, that computes the ages and age levels for links, and the *Strategy Manager*, that provides access to a repository of aging strategies.

The *Ager* interacts with one or more *evolution controllers* to obtain information about relevant evolution-indicating events. The controller receives these events from components like version managers or third-party change detection services and stores them. The *Ager* also exchanges information with the aging client, where the aging information is stored. The aging links are embedded into XML documents enabling easy integration of link aging into existing web documents and applications.

The *EvEnAge* service is part of a service framework for evolution management in information networks, the evolution engine *EvEn*. *EvEn* contains evolution-related services like notification, version management and strategy-based document merging [11]. This service framework is currently under development at our department.

4.2 Link Ages: Management and Rendering

Representing Aging Links Avoiding a central storage on the aging server, the aging information is inserted into the client documents. This is achieved by extending XML link elements [4] with additional attributes:

```
<!ATTLIST AgingLink
     xmlns:xlink CDATA    #FIXED "http://www.w3.org/1999/xlink"
     xlink:href  CDATA    #REQUIRED       type        NMTOKEN  #IMPLIED
     age         NMTOKEN  #REQUIRED       ageLevel    NMTOKEN  #REQUIRED
     strategy    NMTOKEN  #IMPLIED        lastUpdate  NMTOKEN  #REQUIRED>
```

This meta information comprises the link age, the associated age level, the link type and the date of the last age update. In addition, the link element has an optional strategy attribute that identifies the chosen aging strategy. If the attribute is omitted, the aging strategy will be determined by the link type.

Rendering Link Age Focusing on non-intrusiveness, visualization of link aging by fading link colors was implemented: The color used for the link fades with the reaching of higher age levels (see Fig.1). In case a threshold value for link deactivation is defined, the link color changes to red when the age approaches this value. This color signal means that user interaction is required to avoid loosing the link.

Age level visualization is achieved via an XSL stylesheet[1]. Hence, age visualization may be easily adapted to user preferences (like colors) and the combination with existing stylesheets is conveniently supported [3]. For the rendering task a web browser environment is sufficient. These properties empower easy integration of the aging concept on the client side.

In future it is planned to generate such stylesheets from the information stored in the aging strategy repository. This enables a dynamic extension and adaptation of the supported age levels and visualization properties.

4.3 EvEnAge Components

The Ager The *Ager* regularly visits the XML documents of the aging clients and updates the age information. An incremental approach to age computation is implemented by the tool. The client documents are processed by an XML parser [9] that provides the input to the *Ager*. The output of the *Ager* is handed over to an XML generator. The computation of the output, i.e. the new age information, consists of the following six steps:

Step 1: Access the evolution controller for the classified evolution-indicating events that occurred for the link target since the last update of the age;
Step 2: Look up the rating of the occurred classified events and compute the new link age age_{New} incrementing the current age by the sum of those ratings;
Step 3: Access the strategy manager and look up the aging strategy identified by the optional attribute *strategy* or by the link type;
Step 4: Determine if the threshold value of a new age level has been reached by age_{New} for the chosen strategy. In this case update the age level;
Step 5: Enter the current date as the date of the last age update;
Step 6: Repeat steps 1 - 5 for all aging links in the considered information object.

The Strategy Manager For the strategy management a tool has been implemented in Java which enables the definition of new aging strategies (e.g. for new link types) and the update of existing ones. Supported updates are changing threshold values, adding or deleting age levels, etc. Figure 4 shows the user interface for mapping threshold values to age levels. A further functionality of this tool is the rating of evolution indicating events.

[1] The incorporation of a JavaScript function into the stylesheet enables the inspection of the link type and age level of each link in an extra window (see Fig.1).

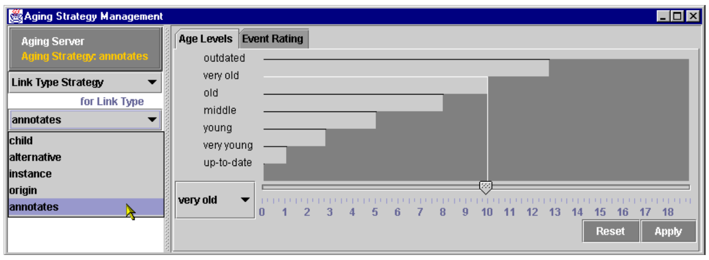


Figure 4. The Management of Aging Strategies

Handling Evolution Indicating Events In order to test our aging server, we constructed an environment for the generation and handling of evolution-indicating events. This environment consists of an XML-based event controller and an XML editor with a simple version management. For each considered target information object, the controller stores the occurred evolution-indicating events in a separate XML file and, on request, passes them to the *Ager*.

Currently only two classes of evolution-indicating events are supported, namely minor and major changes. The author classifies the changes manually. The event controller is informed about the new version generation and its classification.

5 Related Work

Our work on link aging has been inspired by research in different areas: The definition of expiry dates as it is specified in the HTTP [1] builds upon a simple notion of time-driven aging: A piece of information, here an HTML document, is only valid for a certain amount of time and looses validity when the expiry date is reached. In contrast to our approach, aging applies to documents, not to links. An expiry date may be considered as a simple form of a topicality lifeline (see Sect.3.3).

As link aging, link checking for Web sites is concerned with the effects of target object evolution on links. It addresses a very basic but also drastic kind of evolution: The referenced information object has been deleted or moved away. Link checking can be automated; different tools for this purpose exist (e.g. [6]).

Enriching links with additional properties, we conform with a number of other projects. In [15] links are equipped with probabilities which, combined with other facts, improve information retrieval effectivity within hypertext systems. We hold the view that link probabilities can be combined with link ages, as the age of a link may also influence the probability of link relevance.

Evolution awareness is an important topic in CSCW research and system development. Methods for its improvememt range from a very close coupling of the cooperation partners as it is implemented by WYSIWIS editors [21] over more relaxed methods as proposed in [7] to asynchronous notification in separate information spaces [14]. Since aging supports close as well as loose cooperation it could be effectively combined with other awareness methods.

6 Conclusions and Further Research

In this paper we present ongoing work on link aging as a non-intrusive mechanism to improve evolution awareness in cooperative work with linked information. Customization through aging strategies enables an adaptation of the aging process to different link types and project situations. The generalization of link aging allows for a smooth integration of time- and change-based aging, covering an extensible set of aging process variants. This ensures the applicability to domains with different evolution characteristics. First results with the *EvEnAge* prototype show that it can be easily integrated into existing link frameworks.

As a next step we will integrate a differencing engine, developed at our department [20]. This engine is based on a classification and rating scheme for document change operations which can be exploited for rating evolution-indicating events.

Our approach to link aging can be improved by user interaction. Therefore, we will integrate an option for interactively changing the age level, e.g. resetting it after validity has been checked by user intervention. The choice of age level thresholds and the rating of evolution-indicating events is crucial for the correct course of the aging process. With this in mind, we will experiment with different methods to facilitate their adequate fixing, e.g. the inclusion of user feedback, the statistical evaluation of value choices, and the collection of threshold profiles.

Within our system, we represent age levels as discrete time intervals. Modeling age by step functions can only be considered as a first approximation since aging is a continuous process. Thus, for our further research we will model age levels as fuzzy sets (see e.g. [23]). The resulting vagueness of age levels can be exploited for proximity queries and more intuitive graphical user interfaces.

Acknowledgments: The research described in this paper was partly supported by the HSPIII Project WEL and the DFG Project KOLIBRI (DFG Schm 450/7-1 and DFG MA 2005/1-2). Thanks are due to Helen Pert and Jack Koftikian for comments on the manuscript.

References

[1] T. Berners-Lee, R. Fielding, and H. Frystyk. Hypertext Transfer Protocol – HTTP/1.0. http://www.w3.org/Protocols/HTTP/1.0/spec, February 1996.

[2] S. Chawathe and H. Garcia-Molina. Meaningful Change Detection in Structured Data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 26–37, New York, May 1997.

[3] J. Clark. XSL Transformations (XSLT) Version 1.0, http://www.w3.org/TR/xslt, November 1999.

[4] S. DeRose, E. Maler, D. Orchard, and B. Trafford. XML Linking Language (XLink) - W3C Working Draft 21-February-2000, http://www.w3.org/TR/xlink.

[5] P. Dourish and V. Bellotti. Awareness and Coordination in Shared Workspaces. In *Proceedings of the ACM CSCW'92, Toronto, Canada*, pages 107 –114. ACM-Press, November 1992.

[6] R.T. Fielding. Maintaining Distributed Hypertext Infostructures: Welcome to MOM-spider's Web. In *First International World-Wide Web Conference (WWW94) in Geneva, Switzerland.* Elsevier Science, May 1994.

[7] S. Greenberg, C. Gutwin, and A. Cockburn. Using Distortion-Oriented Displays to Support Workspace Awareness. In *Proceedings of the HCI'96 Conference, London*, pages 299 –314. Springer-Verlag, August 1996.

[8] K.M. Hansen, C. Yndigegn, and K. Grønbæk. Dynamic Use of Digital Library Material - Supporting Users with Typed Links in Open Hypermedia. In *Proceedings of the ECDL'99 Conference, Paris, France*, page 254 ff. Springer-Verlag, September 1999.

[9] IBM alpha Works. XML Parser for Java. http://www.alphaWorks.ibm.com.

[10] F. Matthes and U. Steffens. PIA - A Generic Model and System for Interactive Product and Service Catalogs. In *Proceedings of the ECDL'99 Conference, Paris, France*, pages 403–422. Springer-Verlag, September 1999.

[11] J. Munson and P. Dewan. A Flexible Object Merging Framework. In *Proceedings of the ACM CSCW'94 Conference, Chapel Hill, NC*, pages 231 – 242. ACM-Press, October 1994.

[12] NetMind. Mind-It Notification Service. http://www.netmind.com/.

[13] C. Niederée, U. Steffens, J.W. Schmidt, and F. Matthes. Improving Evolution Awareness through Aging Links. Technical report, Technical University Hamburg-Harburg, August 2000. http://www.sts.tu-harburg.de/papers/STS-Rep-2000.html.

[14] U. Rauschenbach. Supporting Awareness in Shared Workspaces Using Relevance-dependent Event Notifications. In *Proceedings of the CVE'96 Workshop, Nottingham*, September 1996.

[15] T. Rölleke and M. Blömer. Probabilistic Logical Information Retrieval for Content, Hypertext, and Database Querying. In *Hypertext - Information Retrieval - Multimedia, HIM'97*, pages 147–160. Universitätsverlag Konstanz, 1997.

[16] J.W. Schmidt, G. Schröder, C. Niederée, and F. Matthes. Linguistic and Architectural Requirements for Personalized Digital Libraries. *International Journal of Digital Libraries*, 1(1), 1997.

[17] K. Shafer, S. Weibel, and J. Fausey. Introduction to Persistent Uniform Resource Locators. http://purl.oclc.org, 1996.

[18] Software Systems Institute, Technical University Hamburg-Harburg. Homepage of the KOLIBRI project. http://www.sts.tu-harburg.de/projects/Kolibri.

[19] Software Systems Institute, Technical University Hamburg-Harburg. Homepage of the Warburg Electronic Library project (WEL). http://www.sts.tu-harburg.de/projects/WEL.

[20] C. Spreen. Change Management für Metadaten in Digitalen Bibliotheken. Master's Thesis, February 2000. http://www.sts.tu-harburg.de/papers/2000/Spre00.

[21] M. Stefik, D.G. Bobrow, G. Foster, S. Lanning, and D.G. Tatar. WYSIWIS Revised: Early experiences with Multiuser Interfaces. *ACM Transactions on Office Information Systems*, 5(2):147–167, 1987.

[22] R. H. Trigg and M. Weiser. TEXTNET: A Network-Based Approach to Text Handling. *ACM Transactions on Office Information Systems*, 4(1):1–23, 1983.

[23] L. A. Zadeh. A Theory of Approximate Reasoning. *Machine Intelligence*, 9:149–194, 1979.