Journal of Universal Computer Science, vol. 8, no. 10 (2002), 944-956
submitted: 24/4/02, accepted: 15/10/02, appeared: 28/10/02 © J.UCS

Towards an XML-based Implementation of Structured Hypermedia Documents

Jörg Westbomke
(Research Institute for applied Knowledge Processing, Ulm, Germany
westbomke@faw.uni-ulm.de)

Gisbert Dittrich
(University of Dortmund, Germany
dittrich@ls1.cs.uni-dortmund.de)

Abstract: Document structures are a crucial mechanism for the creation and the usability of complex hypermedia documents. They form a possibility to deal with the inherent complexity of such documents and with document structures it is also possible to support the reuse of parts of hypermedia documents. In several theoretical approaches different kinds of document structures have been proposed. For example in the Dexter Hypertext Model or in the hypermedia model developed by Klaus Tochtermann.

In the creation process of such hypermedia documents, which is strongly influenced by the offered functionality of the existing editors and tools, only simple kinds of structures could presently be used. Furthermore the use of hypermedia documents is often somehow connected to special system requirements, which makes it difficult to use these documents in a network. Especially the use of such hypermedia documents in the internet with all its different platforms and operating systems still cause many difficulties. The profit of hypermedia documents could obviously be increased, if broad forms of structuring could be used to build hypermedia documents and when these documents fulfill at the time the demands of interoperability and platform independency.

This papers presents a contribution to this topic by introducing techniques for the implementation of structured hypermedia documents, which fulfill the demands of system and platform independency. These techniques are consequently based on the Extensible Markup Language. To form the basis for an XML-based implementation of structured hypermedia documents, the concepts of the Tochtermann model were transformed into a XML document type definition. Because we understand the process of creating a hypermedia document as an integrative process, not only the document type definition itself is described, but also the aspects of displaying such a XML-based hypermedia document. Due to the continuous use of XML conform techniques the developed HMDoc hypermedia documents are platform and system independent and can therefore be easily used in networks like the internet.

Key Words: hypermedia, document structuring, XML-based document notation
Category: H.1

1 Introduction

In the year 1945 Vannevar Bush presented the idea to combine information units with associative links on a mechanical machine called MEMEX (memory extender, [Bush 45]). In his considerations Bush describes a imaginary machine, that was able to connect pieces of information in an arbitrary order and to store the sequence these information pieces were viewed by an user. This recording was the basis to give other users the same view on the information resources. With the ongoing development in the field of electrical engineering and computer science the idea of Bush became more and more reality. AUGMENT [Engelbart 63] was the first software, which was capable to realize the idea of Bush. Ted Nelson ([Nelson 81]) coined in 1981 the term *hypertext* for this kind of information processing by linking information pieces together. With the increasing integration of different kind of media types in the following decades the term hypertext was more and more changed to *hypermedia* to stress the importance of the concept of linkage of information resources, not only for the usability of textual information resources but also for the use of video, audio and other media types.

With the dramatically increasing computational power of the computer technology the concept of hypertext/hypermedia gain access to different kind of computer programs. Today the use of hyperlinks to connect different information resources can be viewed as standard technique. Hyperlinks can be found in almost all realization of online help systems or in text processing or presentation programs. In the internet the hyperlinking is the main paradigm for the use of the World Wide Web. Despite the powerful development of the concept of hyperlinks there are still many problems, which are connected to this technology. Today one of the main problems in the daily use of hypermedia documents is the complexity of hypermedia documents and the missing interoperability. As a consequence, hypermedia documents can often only be used on a special platform with special player applications. Due to missing structuring concepts and solutions existing hypermedia software often lead to the *lost in hyperspace* problem. This term describes the situation, where a user of a hypertext or hypermedia document is overwhelmed by information and links, so that he can't use the information, which is in the document.

The development of techniques to enable the interoperable use of linked hypermedia systems is actually a big challenge for the research in the field of computer science. The core issue in this topic is to find a notation that is capable to build structured and platform independent hypermedia documents, because such a notation can be seen as a realization of a hypermedia model, which allows to use structured hypermedia documents under real life conditions. This paper presents first steps of the authors towards such a notation to realize the use of structured, platform independent hypermedia documents under real life conditions by using existing software systems and technologies like XML.

2 Approach

The aim of the presented work is the development of techniques for the XML-based implementation of structured hypermedia documents, which fulfill the requirements of platform and system independency. The techniques, which have to been developed, should base on open standards to fulfill the requirements of platform indpendency. The presented XML-based implementation of structured hypermedia documents is viewed as integrative process, that requires the existence of a suitable notation and the existence of useable presentation capabilities. Solutions for both aspects have to be found, which could be seamlessly integrated into the creation process.

The approach, that we followed in our work, is based on an analysis of different hypermedia models. We tried to find a model that expresses our demands in respect to the document structuring, rather than to create an own formal model. The next step was to make use of these concepts in the implementation process. To do this we needed a notation that could express all the concepts of the chosen hypermedia model and that can be created and edited with existing tools. We choose XML as base technology for this part of our work, because we believe XML is a well accepted technology in the hypermedia community and meets, as an open W3C standard, all requirements in respect to the platform independency. In this step we had to transform the concepts of the chosen hypermedia model into an XML document type definition. With this DTD it was possible to implement structured, platform independent hypermedia documents. To be sure that the XML-DTD was capable to cover the whole implementation process, we concentrated in the final steps of our work on how our XML hypermedia documents can be presented and how they can be created or edited with the help of existing software tools.

3 Hypermedia Models

Document structures are central concepts for the creation and especially for the use of complex hypermedia documents. In basic research work these concepts have been formalized and expressed as hypermedia models. Several different models for hypermedia can be found in the literature. For example the VDM-based model of Lange [Lange 90], the hypergraph based model introduced by Tompa in 1989 [Tompa 89] or the model for distributed hypertexts published by Meiser [Meiser 91]. In this context the work of the Dexter hypertext reference model [Halasz 94] has to be stressed. The Dexter approach is the most known model for hypermedia and was, since its introduction in 1990, enhanced in many follow-up papers. But we didn't choose the Dexter model as the basis of our work, because the Dexter model has in respect to our goals and the importance of document structures to achieve these goals some drawbacks. First of all, the Dexter model isn't continuously formalized. The structuring of media objects, which were called components in the Dexter terminology, could not be specified within the Dexter model itself and have to be specified with external specification techniques like ODA or SGML. In addition, document structures are not the main focus of the Dexter model, therefore the document structures, introduced by the Dexter model, are of less expressive power than needed for our goals. And the last reason why we don't choose the Dexter model was the used specification technique. The Dexter model is specified in Z, but our aim was to use XML as specification technique. So many difficulties have to be solved before a Z specification can be transformed into an XML specification. We can not go into detail here, the interested reader should see [Westbomke 02] for more details on this topic.

This paper builds up on the hypermedia model introduced in the Ph.D. thesis of Klaus Tochtermann ([Tochtermann 95]). This model was chosen because the document structuring in this model goes much beyond the structuring used in the Dexter model or in the other mentioned models. In addition to this the Tochtermann model is completely formalized, so that its forms a good basis for the transformation into an XML document type definition.

The Tochtermann hypermedia model distinguishes between basic hypermedia concepts and structuring concepts. In the first level the elementary hypermedia concepts like document node, media object, anchor or link are described. By using these concepts hypermedia documents without structures could be specified (see figure 1).

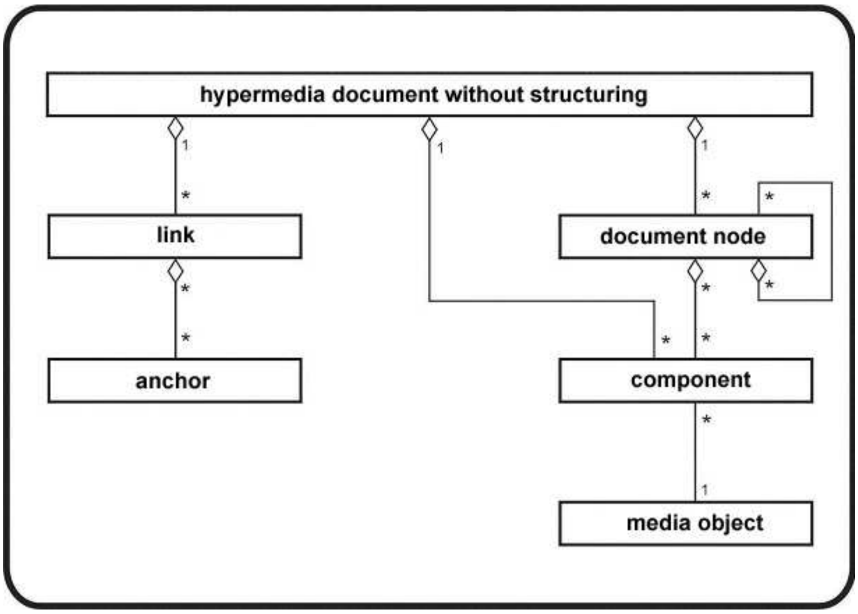

Figure 1: Schematic Composition of hypermedia documents without structuring according to Tochtermann

The second level introduces several forms of document structures, like link structures, views, view nodes, subdocuments and some more. An overview of these complex concepts and the correlation between them is given in figure 2. For a more detailed description of the Tochtermann hypermedia model see [Tochtermann 95] or [Westbomke 02].

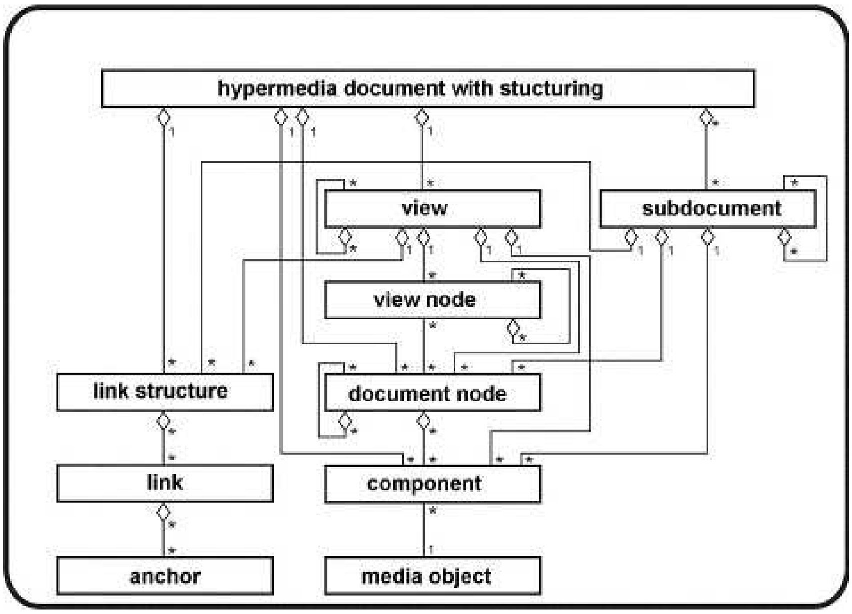

Figure 2: Hypermedia document with document structures

4 Transforming the VDM specifications into an XML Document Type Definition

On the theoretical level document structures are a well known mechanism to deal with the complexity of hypermedia documents, but they are rarely used in the existing authoring tools. Therefore it is presently not possible to draw advantages form complex hypermedia document structuring, like the concepts given in the Tochtermann model, because there are no tools, which support the use of these concepts. So we observe a big gap between the concepts introduced in the theoretical world and the concepts that are usable in the real world. This paper contributes to this problem, by introducing a way how the complex document structuring of the formal hypermedia model can be used in practice. To do this, it is necessary that the formal hypermedia model by Tochtermann – subsequently referenced as HM-model – is transferred into a notation, that

1.) can be executed by a runtime system.
2.) can be displayed through style sheet technologies on different output media.
3.) can be used on different platforms with different operating systems.

The Extensible Markup Language (XML, [Goldfarb, Prescod 99]) meets all these requirements and is additionally a technique, which is becoming more and more popular in the hypermedia community. So an increasing support for XML can be assumed, to use XML under practical conditions. Consequently, XML is the base technology, on which we build up our work.

4.1 Preparatory Work

But before transforming the HM-model, which is formulated in the Vienna Development Method (VDM), into an XML document type definition some problems had to be solved. VDM possess rich data types, for which there are no direct equivalents in XML. Therefore these data types had to be build up in a first step as element declarations. Further on, it was necessary to extend the XML formalism with the from the Vienna Development Method known concept of invariance. Invariance is a concept, that offers possibilities to formulate constrains for element declarations. An invariant is mathematically spoken a function, which map an instance of a data type to a boolean value. XML did not offer a comparable complex mechanism, not in the XML specification 1.0, nor in the XML schema specification. XML schema offer possibilities to formulate constrains for element declarations, but not with the needed expressive power. Our solution was to express invariance as validity constrains of the XML specification 1.0. But in the XML specification the validity constraints are only given as colloquial restrictions, for our purposes this was not enough. We needed an XML conform formalism to be able to express the VDM concept invariance adequately. So, we created a new XML conform formalism based on XPath expressions. With this, due to its analogy to the validity constrains of the XML specification also validity constrain named mechanism, it is possible to formulate constrains to arbitrary element declarations. The following figure shows an example how a VDM invariant looks like as an XML validity constrain.

```
<!ELEMENT document_graph (document_nodes?, view_nodes?,
                          components?, link_structures)>

<!-- GKB: document_graph {count(./document_nodes)>0 or
      count(./view_nodes)>0}
      The document graph must contain at least one document
      node or one view node.-->
```

Figure 3: VDM-Invariant transformed into a XML validity constrain.

With the help of these extensions of the Extensible Markup Language a document type definition could be realized, which expresses the hypermedia concepts of the HM-model in an XML conform syntax. This HMDoc-DTD called document type definition forms the basis to formulate structured hypermedia documents in XML. But this HMDoc-DTD did not represent a 1:1-transformation of the HM-model. Within the scope of the reformulation from VDM to XML, some conceptual modifications were also carried out. The structuring of media objects, which was introduced in the HM-model, were not transformed into the HMDoc-DTD. By this adoption the expressive power of structuring within the HMDoc-DTD is slightly cut, but as a consequence it becomes possible to use established formats for media objects like JPEG, MPEG or Quicktime in combination with the HMDoc-DTD. This was of high importance for the pretense of the work to take practical aspects of the implementation process into account.

4.2 HMDoc-DTD

Building up on the preparatory work, described in chapter 4.1, it was possible to realize an XML document type definition, called HMDoc-DTD. To build the HMDoc-DTD every concept from the HM-model was transformed into corresponding XML element declarations, expressing the same concepts with the same limitations. The following table contains a brief clipping from the HMDoc-DTD. Shown are the XML declarations, which define the hypermedia concepts *document node* and *hypermedia document*, whereby the validity constrains (GKB) for the hypermedia document object and for the document graph were abridged.

```
<!ELEMENT document_nodes (document_node+) >
<!ELEMENT document_node (node_object) >
<!ATTLIST document_node doc_node_id ID #REQUIRED >
<!ELEMENT node_object (content, attributes?, node_object*) >
<!ELEMENT content (components?,document_nodes?,docnoderefs?)>
<!ELEMENT docnoderefs EMPTY>
<!ATTLIST docnoderefs doc_node_refs IDREFS #REQUIRED >
<!-- GKB: docnoderefs {count(id(@doc_node_refs[@doc_node_id]) =
                       count(id(@doc_node_refs))}
References of type doc_node_refs must contain values, which are
defined as ID-values of doc_node_id attributes. -->


<!ELEMENT HMDoc (hyperdocument_object) >
<!ATTLIST HMDoc doc_id ID #REQUIRED >
<!ELEMENT hyperdocument_object (document_object)>
<!-- GKB : hyperdocument_object -->
<!ELEMENT document_object (document_base, document_graph,
document_structures?, attributes?) >
<!ELEMENT document_base (media_object*) >
<!ELEMENT document_graph (document_nodes?, view_nodes?,
components?, link_structures) >
<!-- GKB: document_graph -->
<!ELEMENT link_structures (link_structure+)>
<!ELEMENT link_structure (structure_object) >
<!ATTLIST link_structure structure_id ID #REQUIRED >
<!ELEMENT structure_object (links, attributes?) >
```

Figure 4: Clipping from the HMDoc-DTD

The above shown element declarations are only a very short clipping from the HMDoc-DTD, but they show how the document type definition is structured and give an impression how the hypermedia concepts of the HM-model are mapped to XML element declarations. A complete listing of the HMDoc-DTD is given in [Westbomke 02].

With the help of the HMDoc-DTD it is possible to formulate hypermedia documents, which could make use of complex document structures and this in a notation, which is commonly accepted and well supported by existing authoring software, e.g. editors, converting tool, etc.

5 Presentation of Hypermedia Documents

In chapter three and four the needs for the development of the HMDoc-DTD were motivated and the HMDoc-DTD was briefly introduced. With the results from chapter three and four it is now possible to write XML-based hypermedia documents. But as stated in the introduction, the aim was to deal with the whole creation process of hypermedia documents. Up to now the developed HMDoc-documents only code the structural relations between the different hypermedia concepts. They did not contain any information on how the hypermedia document should be displayed on a special output device. For the presentation of the hypermedia document, layout rules, which describe the way the hypermedia concepts should be displayed, have to be assigned to the HMDoc document.

The Extensible Style Sheet Language (XSL) and the Cascading Style Sheets (CSS), developed for the display of HTML documents, offer both possibilities to describe the presentation of an XML document. We choose XSL to describe the presentation of HMDoc documents, because the expressive power of CSS isn't enough to achieve the desired document layout. The main reason why CSS fails, is its strong connection to HTML. CSS was created to describe the layout of HTML documents and HTML documents have a fixed set of structural elements each with a well defined meaning. So the main task of CSS elements are to add format attributes to the HTML elements. But XML elements did not have these fixed meanings, so the style sheet language needs mechanism not only to code the format attributes but also to express the layout of an XML document. The following figure, which is adopted from [Bach 01], show the basic architecture of an XSL-based presentation process.

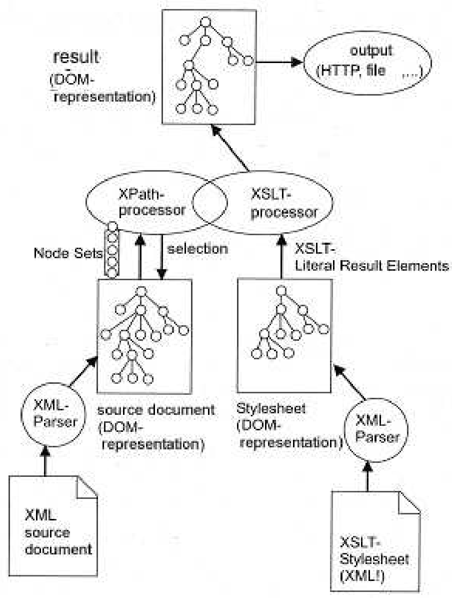

Figure 5: XSL-based presentation process

To use XSL for the formulation of the presentation of HMDoc documents two things are necessary to do. First, for each hypermedia concept possible presentation forms have to be developed. For the concept of a document node e.g. this task is not so difficult, but describing the presentation of a component or a link structure is a much more challenging task. Second, for each desired presentation of a hypermedia concept a description as XSL style sheet has to be implemented.

While realizing the first step, we worked out presentation elements for each hypermedia object. That means, each hypermedia object was examined in respect to its display properties. We distinguished hypermedia concepts, which have a direct influence on the presentation of the hypermedia document, like media objects, document nodes, etc. and hypermedia concepts, which fulfill a more structural function. These hypermedia concepts aren't displayed through own presentation objects, these hypermedia concepts are displayed by their influence on the presentation of other hypermedia concepts. For example, a media object of type MPEG is directly represented in the output, while its size, position, color, brightness, transparency, etc are given by the corresponding component. The following figure gives an impression of the work, which has to be done to specify the presentation of a HMDoc document. The upper part of the figure show a simple HMDoc document with one document node and two media objects and the lower part of the figure show the description of the presentation of the HMDoc document as XSL template rules. The result of the XSL style sheet is a formatting object description, which can be directly displayed through an XSL processors.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE HMDoc SYSTEM "E:\Dissertation\Diss\XML\DTDs\HMDoks mit
Struk DTD.e.1.2.dtd">
<?xml-stylesheet href="Beispiel.xslt" type="text/xsl" ?>
<HMDoc doc_id="Testdokument">
    <hyperdocument_object>
     <document_object>
      <document_base>
       <media_object media_object_id="Text">
        <discrete_media>
          <text Format="RTF">Dies ist ein Probetext!</text>
        </discrete_media>
       </media_object>
       <media_object media_object_id="Grafik">
        <discrete_media>
          <graphic>
            <pixel_graphic format="JPEG"
source="C:\EigeneBilder\Grafik.jpg"/>
          </graphic>
        </discrete_media>
       </media_object>
      </document_base>
      <document_graph>
       <document_nodes>
        <document_node doc_node_id="Knoten1">
          <node_object>
            <content>
              <components>
                <component comp_id="Komponente1">
                  <component_object
media_object_ref="Text"/>
                </component>
                <component comp_id="Komponente2">
                  <component_object
media_object_ref="Grafik"/>
                </component>
              </components>
            </content>
          </node_object>
        </document_node>
       </document_nodes>
       <link_structures/>
      </document_graph>
     </document_object>
    </hyperdocument_object>
</HMDoc>
```

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="1.0"

xmlns:xsl="http://www.w3.org/1999/XSL/Transform"xmlns:fo="http://w
ww.w3.org/1999/XSL/Format">
    <xsl:output method="html" version="1.0" encoding="iso-8859-1"
indent="yes"/>
    <xsl:template match="HMDoc" >
       <fo:root>
          <fo:layout-master-set>
             <fo:simple-page-master master-name="DIN-A4">
                page-height="29.7cm"
                page-width="21cm"
                margin-top="0.5cm"
                margin-left="1cm"
                margin-right="0.5cm"
             </fo:simple-page-master>
          </fo:layout-master-set>
          <xsl:apply-templates/>
       </fo:root>
    </xsl:template>
    <xsl:template
match="/hyperdocument_object/document_object/document_graph">
       <fo:page-sequence master-name="DIN-A4">
          <fo:flow flow-name="Seite1">
             <fo:block
                text-indent ="1em"
                font-family="sans-serif"
                font-size="12pt"
                space-before.minimum="2pt"
                space-before.maximum="6pt"
                space-before.optimum="4pt"
                space-after.minimum="2pt"
                space-after.maximum="6pt"
                space-after.optimum="4pt" >
                <xsl:value-of select=
"/hyperdocument_object/document_object/document_base/
media_object/discrete_media/text" />
             </fo:block>
             <fo:block
                text-align="end">
                <fo:external-graphic
src="{/hyperdocument_object/document_object/
document_base/media_object/discrete_media/graphic/@source}"
width="99px" height="109px" />
             </fo:block>
          </fo:flow>
       </fo:page-sequence>
    </xsl:template>
</xsl:stylesheet>
```

Figure 6: HMDoc-document and XSL template rules

6 Conclusion and Further Work

We presented in this paper an approach, how structured hypermedia documents could be implemented using XML. The paper focuses on the integrative application of existing implementation techniques for the practice-oriented creation and usage of structured hypermedia documents. First, the paper selects a model for hypermedia, that consists of structuring as central and integrative part of the model. In a second step it was shown, how the concepts of the chosen hypermedia model can be transferred into an XML document type definition. This document type definition form the basis to implement structured hypermedia documents as XML documents. In chapter five we discussed, how these HMDoc XML documents can be presented on different output devices using existing display technologies. The Extensible Style Sheet Language was the preferred solution to describe the layout of a HMDoc document. We described, which steps have to be taken to specify the presentation of HMDoc documents and gave a very simple example for that.

Our future work concentrates on applying the developed concepts to real world documents. We did a first step in that direction by investigation how the HMDoc-DTD in combination with HTML can be used to code a real world application. In [Thiemann 00] is described, how the HMDoc-DTD was used to code parts of an existing archaeological information systems as HMDoc document. In a second step XSL(T) rules were define to transform the HMDoc document into a HTML document, which could be displayed in an ordinary web browser. The experiences gained from that project showed us, that it is indispensable to have an authoring tool or editor, that supports the creation process, because the HMDoc documents get easily very complex, so that is very difficult to handle them only with the help of an ordinary text editor. Like in many programming languages, like C++ or Java, it is very important to support the programmer/editor with an integrated development environment. Our first experiences in that direction ([Westbomke 02] and [Neubach 01]) showed us, that the existing tools and editors are presently not fulfilling the requirements for such an authoring environment and can not be used to implement HMDoc documents in the desired way. But these first investigations also encouraged us to go deeper into that topic, because we believe that these authoring environments can be build with arguable efforts.

Acknowledgements

This paper is based on the Ph.D. thesis of the first author. He gratefully thank Prof. Dr. Gisbert Dittrich, who supervised the Ph.D. thesis, for his support and advice over the years and for the intense and fruitful discussions with encouraged him to do this challenging work.

References

[Bach 00] Bach, M.: „XSL und XPath – verständlich und praxisnah“; Addison-Wesley Verlag, ISBN 3-8273-1661-8, 2000

[Bush 45] Bush, V.: „As we may think.“; Atlantic Monthly 7, (1945), 101-108 http://www.theatlantic.com/unbound/flashbks/computer/bushf.htm

[Engelbart 63] Engelbart, D. C.: „A Conceptual Framework for the Augmentation of Man’s Intellect.“; P. D. Howerton and D. C. Weeks (editors): „Vistas in Information Handling“, Volume 1, 1-29, Spartan Books, Washington D.C. (1963)

[Goldfarb, Prescod 99] Goldfarb, C. F., Prescod, P.: „XML Handbuch“; Prentice Hall (1999)

[Halasz 94] Halasz, F., Schwartz, M.: „The Dexter Hypertext Reference Model“; CACM (Communication of the ACM), 37, 2 (1994), 30-39.

[Lange 90] Lange, D.B.: „A Formal Model of Hypertext“; Proc. of the Hypertext Standardization Workshop 1990, NIST Special Publication 500-178 (1990), 145-166.

[Meiser 91] Meiser, D.: „Konzepte eines verteilten Hypertextsystems“; in: Hypertext/Hyperpermedia ’91, Proc. der Tagung der GI, SI und OCG, Graz (1991), 191-204.

[Nelson 81] Nelson, T. H.: „Literary Machines“; Ed. 87.1, Mindful Press, Sausalito (1987)

[Neubach 01] Neubach, P.: „Entwurf und prototypische Implementierung eines Editors zur Bearbeitung von XML-basierten Hypermediadokumenten“; Diplomarbeit am Fachbereich Informatik der Universität Dortmund, Interne Berichte (2001)

[Thiemann 00] Thiemann, J.: „Erweiterung einer XML-basierten Hyperdokumentbeschreibung um Konzepte der Visualisierung und Anwendung dieser auf ausgewählte Inhalte eines archäologischen Informationssystems“, Diplomarbeit am Fachbereich Informatik der Universität Dortmund, Interne Berichte (2000)

[Tochtermann 95] Tochtermann, K.: „Ein Modell für Hypermedia – Beschreibung und integrierte Formalisierung wesentlicher Hypermediakonzepte“; Dissertation, Shaker Verlag, ISBN 3-8265-0618-9, Dortmund (1995)

[Tompa 89] Tompa, F. W.: „A Data Model for Flexible Hypertext Database Systems“; ACM Transaction on Information Systems, 7, 1 (1989), 85-100.

[Westbomke 02] Westbomke, J.: “XML-basierte Implementierung strukturierter Hypermediadokumente - Aspekte der Erzeugung, der Notation und der Darstellung”; Dissertation, Shaker Verlag, ISBN 3-8265-9986-1, Dortmund (2002)

[W3C 98] Bray, T., Paoli, J., Sperberg-McQueen, C. M.: „Extensible Markup Language (XML) 1.0“; http://www.w3.org/TR/1998/REC-xml-19980210, 1998.

[W3C 01] Adler, S., Berglund, A., Caruso, J. et al.: „Extensible Stylesheet Language (XSL) Version 1.0“; http://www.w3.org/TR/xsl/, 2001.