

# XML and the authoring environment of the Hyperbook

Abdoulmajid Hakki

Lappeenranta University of Technology, Department of Information Technology, Laboratory of Communications Engineering, P.O.BOX 15, FIN 53850 Finland  
{Abdoulmajid Hakki, ahakki}@lut.fi  
<http://www.it.lut.fi>

**Abstract.** This paper presents architecture for the development of the Hyperbook, using XML to describe the application domain to adapt the content of the Hyperbook to the user's needs. XML data-centric orientation makes it possible to describe application domain, data access and dynamic data composition functions.

## 1 Introduction

Hypertext books or Hyperbooks, are characterized as a grouping of electronic text considered as an entity [1]. In most cases, these Hyperbook still retain the conventional book structure and are partitioned into subdocuments called chapters, sections, subsections or appendices. Basic Components of Hyperbook Systems are: a) The Adaptation Domain Model [2], b) User Adaptation Model, c) Bookmark Model and d) Storage Model [3].

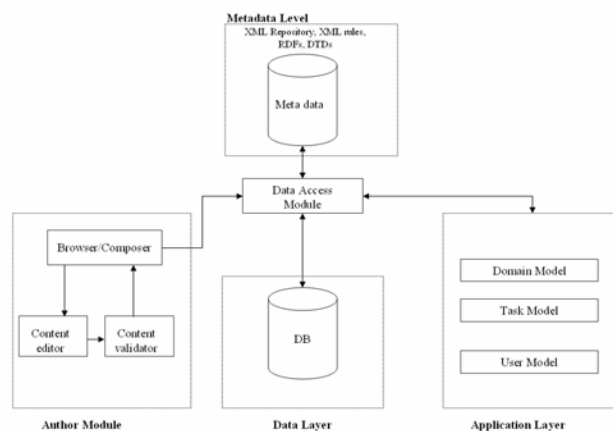
## 2 System architecture

The main goal of the Hyperbook system is to provide a customised view of the content of the book responding to different preferences, interests and users. The application is to serve the strategy of active reading as the need of society in the information age. The system has a three-tier architecture comprising the *Author Module, and Application* and *Data Layers* (Fig. 1). The *Author Module* supports authors/publishers in the creation and editing of the content of the Hyperbook. It allows designing and validating the XML documents to create the content of the Hyperbook.

Customising is an efficient way to improve the usability of the Hyperbook. The goal of the architecture of the Hyperbook is to present a content that is best suited to the user of the application. Accessing the Hyperbook from different devices will provide better future possibilities for the publishing industry. This will involve in some cases accessing local information that provides background information about the local environment that is related to the content of the Hyperbook. This service could be in XML (eXtensible Markup Language) web service format [4].

The Hyperbook data related to a particular module is represented by the XML-based document according to its DTD [5]. Each element contains several conditions, which define the model of presentation according to the given context. Attributes and their values within each element represent conditions. The aim of the Hyperbook is to

experiment with attributes related to the type of data, granularity of presentation, time and spelling [5] of the presented information to several views.



**Fig. 1.** Hyperbook system architecture - Application and Data Layers and Author Module.

## 7 Discussion and further work

In this paper, the system architecture to support the Application Domain and User Model was suggested to store XML-based Hyperbook documents. The XML binding offer Hyperbook system is able to manage Adaptation Modules from different types and domains of application using database technology which is particularly suited to improve the access to huge amounts of documents. The experiences are encouraging and promising for further work. The next study will extend the current work by examining the details of the system architecture.

## References

1. Henze Nicola, Adaptive Hyperbooks: Adaptation for Project-Based Learning Resources, Universität Hannover 31 – 32, (2000)
2. Mario Cannataro, Andrea Pugliese, An XML-Based Architecture for Adaptive Hypermedia System Using a Probabilistic User Model, IEE 257 – 265, (2000)
3. Abdoulmajid Hakki, The importance of modeling metadata for Hyperbook, International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering CIS<sup>2</sup>E
4. Robert Steele, Yuri Ventsov, Tharam Dillon, XML Schema-based Discovery and Invocation of Mobile Services, IEEE (2004)
5. Mária Bieliková, Adaptive Presentation of Evolving Information Using XML, IEEE, 103 – 106 (2001)