A Digital Metadata Schema Repository

YEN-CHUN LIN, HSIANG-AN WANG, CHIEN-CHUNG HUANG, WEI CHEN
Department of Computer Science and Information Engineering
National Taiwan University of Science and Technology
43 Keelung Road, Sec. 4, Taipei 106
TAIWAN

Abstract: - The metadata schema of a digital archive describes the structure and attributes of metadata. Analysis and definition of metadata schema for a new digital archive must be carefully performed at the first stage. To ease the task, we implement a metadata schema repository to store metadata schemas as Extensible Markup Language (XML) documents in a native XML database. The metadata schema repository supports storing, creation, searching, and access management of metadata schemas. A user can access the repository through a Web browser. With this repository, various projects and organizations can share their metadata schemas over the Internet. Since the metadata schema must be displayed on a Web browser, we also present the method of translating the XML representation of metadata schema into the HyperText Markup Language document.

Key-Words: - Digital archive, HTML, Metadata schema repository, Native XML database, Web-based, XML

1 Introduction

Digital archives are archives in the digital form. Through them we store and manage valuable digitized resources of, for example, cultural heritage and artifacts. The digital form makes archives easily accessible over the Internet. Analysis of metadata plays an important role in constructing digital archives. Metadata is “data which describes attributes of a resource” [1], and is “the information that makes it possible to find, access, use, and manage information resources” [2]. The result of metadata analysis is crucial because it has an important impact on the content construction process, the accuracy of query results, and the feasibility of both data exchange with other systems and the construction of value-added applications. Therefore, analysis and definition of metadata schema, which describes the structure and attributes of metadata, for a new digital archive should be carefully performed at the early stage.

Creating a metadata schema requires a great amount of time and effort [3]. It usually takes four months to two years to create one [4]. Metadata analyzers and content providers should collaborate to decide the metadata schema. They usually need to consult related projects and international metadata standards. Software tools for assisting the management, analysis, comparison, and creation of metadata schemas are desirable.

ULIS Open Metadata Registry [5] and CORES [6] have been presented to manage metadata schemas. The former can only take care of the Dublin Core metadata schema [7]. The latter can handle many metadata schemas, but not metadata schemas with a hierarchical structure. Another drawback of CORES is that it cannot be used to create or edit schemas over the Internet. These two tools use the Resource Description Framework Schema language [8] to describe metadata schemas.

This paper presents a metadata schema repository for storing and managing various metadata schemas. This repository is useful for searching and creating metadata schemas. Users can query existing metadata schemas stored in the repository and create new metadata schemas. We store metadata schemas in Extensible Markup Language (XML) documents [9], which are suited to represent hierarchical metadata schema structures. Additionally, the structure of metadata schemas can be easily modified. Needless to say, the repository can help reduce the time and effort in creating metadata schemas and ultimately help generate digital archive systems.

Section 2 introduces a metadata schema representation with three tables, which is often used in Taiwan. Section 3 describes our XML version of the metadata schema introduced in Section 2. Section 4 presents the architecture of our metadata schema repository. Section 5 explains the main functions of the metadata schema repository by showing some operations. Section 6 introduces the translation from XML documents into HTML ones for displaying metadata schemas on a Web browser. Section 7 concludes this paper.
2 Metadata Schema in Tabular Form
A metadata schema can be described with three tables [10]. The specification of the Han Dynasty Woodslip Metadata Schema is used for illustration in the following [11]. Table 1 shows part of the Construction table. The Element column is used for naming metadata elements. In this example, the Element has a hierarchical structure. Its root, Title, has two subelements named Main and Other, and Other has three subelements named Type, Name, and Remarks. The Alias column is for the Chinese version of the Element column. The Data Type column describes the data types of atomic elements, which are at the lowest level of the hierarchy of the Element. Here, Varchar means characters of variable length. The Size column describes the maximum size of an atomic element in number of characters.

Table 1. Part of the Construction table.

<table>
<thead>
<tr>
<th>Element</th>
<th>Alias</th>
<th>Data Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Main</td>
<td>Varchar</td>
<td>50</td>
</tr>
<tr>
<td>Other</td>
<td>Type</td>
<td>Varchar</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td>Varchar</td>
<td>50</td>
</tr>
<tr>
<td>Remarks</td>
<td></td>
<td>Varchar</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 2 shows part of the Attribute table. The Element column has the same meaning as that in the Construction table. In the Required column, “*” indicates that the atomic element to the left, Main, must be provided. In the Repeatability column, “◎” specifies that the corresponding element, Other, can have more than one occurrence. In the Attribute column, the Menu indicates that the value of the corresponding atomic element should be selected from a predefined pull-down menu displayed on the monitor. The Provider column describes that the value of an atomic element of the Element is to be filled in by a human or generated automatically by the digital archive system.

Table 2. Part of the Attribute table

<table>
<thead>
<tr>
<th>Element</th>
<th>Required</th>
<th>Repeatability</th>
<th>Attribute</th>
<th>Provider</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Main</td>
<td>*</td>
<td>Menu</td>
<td>Person</td>
</tr>
<tr>
<td>Other</td>
<td>Type</td>
<td>◎</td>
<td>Menu</td>
<td>Person</td>
</tr>
<tr>
<td></td>
<td>Name</td>
<td></td>
<td></td>
<td>Person</td>
</tr>
<tr>
<td>Remarks</td>
<td></td>
<td></td>
<td></td>
<td>Person</td>
</tr>
</tbody>
</table>

Table 3 shows part of the DC-correspondence table. It gives the mapping between the metadata schema and the Dublin Core standard. For example, each of the four atomic elements corresponds to Title defined by the Dublin Core.

Table 3. Part of the DC-correspondence table

<table>
<thead>
<tr>
<th>Element</th>
<th>Dublin Core Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main</td>
<td>Title</td>
</tr>
<tr>
<td>Other</td>
<td>Type</td>
</tr>
<tr>
<td></td>
<td>Name</td>
</tr>
<tr>
<td>Remarks</td>
<td></td>
</tr>
</tbody>
</table>

3 Metadata Schema in XML
XML is a markup language; it has self-defined tags, and thus is flexible and extensible and allows dynamic change of document structure and content. It is well suited for describing data with a complex structure [9]. Since a metadata schema usually is complex, has many items, and is subject to changes, XML is suited for describing metadata schemas.

The contents of Tables 1-3 can be expressed in XML as shown in Fig. 1. Line 2 shows that the Project element has a Name attribute with value Woodslip. This information is beyond the scope of the three tables and represents the project name with which the metadata schema associates. Lines 3-5, for example, show metadata schema elements Title and Main; XML attributes are used to represent metadata schema element attributes, Alias, Data Type, and Size of Main, as well as other related information.

In order to store and manage XML documents efficiently and reduce the complexity of software development, an appropriate database management system is required. Two main categories of database systems are usually used to store XML documents. One is XML-enabled relational database systems, and the other is native XML database systems [12]. The former must map XML schemas into relational...
Before being saved in the database, XML documents are transformed based on a mapping rule. A drawback of this approach is the loss of comments and the order of elements. Moreover, if the structure of XML documents is modified, the mapping must be redefined, which is usually time-consuming. Even worse, it is generally difficult, if not impossible, to retrieve and recover the original XML document with a complex structure from the database.

The native XML database is intended for storing XML documents; an XML document is a basic unit of storage. With such a database, no schema mapping is needed. Hence, we may access XML documents with their original structures and contents intact. Using an XML document as a basic storage unit allows the flexibility of changing metadata schema structure. Therefore, we use a native XML database to store XML documents.

### 4 Architecture of Repository

![Fig. 2. Architecture of metadata schema repository.](image)

The native XML database system is based on the open-source eXist [15]. The system supports XQuery to query XML documents directly [16]. The database stores existing, well-known metadata schemas, including standards such as Dublin Core and Categories for the Description of Works of Art [17], as well as newly created metadata schemas.

### 5 Functions and Operations of Repository

Three main functions are included in the metadata schema repository:

(1) Metadata schema retrieval. It retrieves metadata schemas that match a keyword such as an element name. Queries are transformed into the XQuery statements to retrieve metadata schemas from the database. The schemas in XML are translated into HTML and displayed on the Web browser.

(2) Metadata schema creation. Users can create new metadata schemas from scratch or by modifying existing metadata schemas in the repository. Created schemas can be saved to the repository or to personal computers.

(3) Access management. Access to metadata schemas is classified. Unauthorized users are not allowed to retrieve, modify, or delete any data.

The user interface of the metadata schema repository is shown in Fig. 3. It consists of three areas. On the north-west corner is the Function Area, which displays the functions available to users: Browse Metadata Schema, Search Metadata Schema, and Create Metadata Schema. Users can click to use them. The Operation Area, located to the right of the Function Area, allows users to provide input data after choosing a function. The Schema Area, occupying most of browser window, displays a metadata schema retrieved from the repository or being created.

To view a schema, users can click Browse Metadata Schema to scan through all schemas in the repository. Alternatively, they can click Search Metadata Schema and enter a keyword to search for matching metadata schemas. The Operation Area allows the user to enter a keyword and select Project Name, Element Name, Alias, Data Type, Size, or DC-Element from a pop-up menu to retrieve all the metadata schemas that have a field name matching the keyword. Suppose a user enters the keyword Title and chooses Element Name from the pop-up menu, and this request is sent to the Web server. The Web server then translates the request into the XQuery language to fetch the corresponding XML documents from the database. The search result displayed in the Schema Area, for example, is shown in Fig. 4.
The metadata schema creation process is shown in Fig. 5. After logging into the repository, users may choose either to create from scratch or to modify an existing schema. To create from scratch, users must first enter the project name and the element name to create a root element. Alternatively, to create by using an existing metadata schema, users must first select a metadata schema. After that, users can choose an element to add a subelement, modify contents, or delete an element. When the modification is completed, the schema can be saved with a new name into the repository.
To modify an existing metadata schema, the user has to click Create Metadata Schema in the Function Area shown in Fig. 3, and then clicks Modify Metadata Schema that appears in the Operation Area. After that, all schema names are shown in the Operation Area, and the user chooses a metadata schema to use. The contents of the metadata schema are then shown in the Schema Area. The user can click a hyperlinked element below Element Name to modify. A “★” is prefixed to the clicked element.

6 From XML to HTML
XML documents are not suited for the human to read. To view them with a Web browser, we need to translate the XML data into HTML. Taking the XML document in Fig. 6 as an example, we want to display it as the table shown in Fig. 7 on the browser. Fig. 7 is in fact generated by the browser from the HTML data shown in Fig. 8, which is translated from the XML document shown in Fig. 6. The upper and lower parts of Fig. 7 correspond to the upper and lower parts of Fig. 8, respectively.

The program that translates the XML document into HTML is written with the Document Object Model (DOM) [18] package from JAXP. DOM is a language-neutral interface that allows a program to dynamically access and update XML documents. The translator needs to count the number of layers and the number of child-elements of each element and then finds out the name of each element to create the HTML file.

7 Conclusions
To ease the creation of metadata schemas of digital archives, we have implemented a Web-based metadata schema repository to help metadata analyzers use existing metadata schemas and create a new one. The repository allows projects of various domains to share their metadata schemas over the Internet, and serves as a centralized portal of knowledge for searching and producing metadata schemas. Ultimately, the repository is useful for fast construction of digital archive systems. Since the metadata schema must be displayed on a Web browser, we also presented the translation from the XML representation of metadata schema into the HTML document.

Acknowledgment
This research was supported in part by the National Science Council of Taiwan under contract NSC 94-2422-H-011-001.

<?xml version="1.0" encoding="Big5"?>
<Storage_Place Alias="典藏位置" Repeatability="" >
  <Name Alias="名稱" Data_Type="Varchar" Size="30" Required="★" Repeatability="" Attribute="Select_List" Provider="Person" DC-Element="Description" > </Name>
  <Number Alias="編號" Data_Type="Varchar" Size="30" Required="★" Repeatability="" Attribute="Select_List" Provider="Person" DC-Element="Description" > </Number>
</Storage_Place>
Fig. 6. An XML document.

<table>
<thead>
<tr>
<th>Element Name</th>
<th>Alias</th>
<th>Data Type</th>
<th>Size</th>
<th>Required</th>
<th>Repeatability</th>
<th>Attribute</th>
<th>Provider</th>
<th>DC-Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage_Place</td>
<td>名稱</td>
<td>Varchar</td>
<td>30</td>
<td>★</td>
<td></td>
<td>Select_List</td>
<td>Person</td>
<td>Description</td>
</tr>
<tr>
<td></td>
<td>編號</td>
<td>Varchar</td>
<td>30</td>
<td>★</td>
<td></td>
<td></td>
<td></td>
<td>Description</td>
</tr>
</tbody>
</table>

Fig. 7. A table displayed on the browser based on the XML document shown in Fig. 6.
<table>
<thead>
<tr>
<th>Element Name</th>
<th>Alias</th>
<th>Data Type</th>
<th>Size</th>
<th>Required</th>
<th>Repeatability</th>
<th>Attribute</th>
<th>Provider</th>
<th>DC-Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage_Place</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Select_List</td>
</tr>
<tr>
<td>Name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Person</td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 8. HTML data translated from the XML document shown in Fig. 6 to display the table shown in Fig. 7.

**References:**


