DocML: A Digital Library of University Data

AUTHOR
Ioannis Papadakis
Researcher
Department of Informatics, University of Piraeus
80 Karaoli & Dimitriou str, 18534 Piraeus, Greece
Phone: +30 1 4142122, Fax: +30 1 4142264
Email: jpap@unipi.gr
Homepage: http://thalis.cs.unipi.gr/~jpap

AUTHOR
Vassileios Karakoidas
Researcher
Department of Informatics, University of Piraeus
80 Karaoli & Dimitriou str, 18534 Piraeus, Greece
Phone: +30 1 4142122, Fax: +30 1 4142264
Phone: +30937-733642
Email: bkarak@unipi.gr
Homepage: http://rainbow.cs.unipi.gr/~bkarak

AUTHOR
Vassileios Chrissikopoulos
Professor
Department of Archives and Library Sciences, University of Ionio
Plateia Eleutherias, 49100 Corfu - Greece
Phone: +30 1 4142237, Fax: +30 1 4142264
E-mail: vchris@ionio.gr

ABSTRACT

In this paper, DocML, a Web-based digital library of University data is presented. The goal of this work is to build a system capable of preserving and efficiently managing student assignments in a University environment. It is based on a three-tier architecture, typical for distributed Web applications. Communication between the layers of the digital library and the various architectural components that reside in the middle layer is facilitated through the employment of the XML standard. XML is also employed for the development of a distributed metadata management system that is used to describe the location and content of the digital library’s documents.

KEYWORDS: Digital library, XML, Web, University, metadata

INTRODUCTION

The ultimate goal of a digital library is to provide ways of organizing and efficiently managing large data collections. In this context, DocML, a Web-based digital library of University data is presented. The proposed work is motivated from the fact that there is lack of adequate tools capable of efficiently handling large amounts of diverse data inside a University. Most frequently, students have to deliver a number of assignments in order to pass certain courses. Such assignments may vary from a simple document in ASCII format to a complex 3D environment illustrated as a Virtual Reality Markup Language (VRML) model. Under current practice, such documents are stored in physical form at the corresponding lecturer’s office or,
According to the most optimistic scenario, in digital form within the author’s personal Web site. Although students would benefit a lot from being able to study assignments of relevant content from previous years, access to such knowledge is practically impossible due to the absence of a system capable of storing, indexing and efficiently retrieving such data. The ability of digital library technology to handle large amounts of diverse data renders the employment of such technology as the most suitable solution to the aforementioned issue.

In this paper, DocML is introduced as a digital library system capable of efficiently handling student deliverables and relevant material in a University environment. DocML follows the logical structure of a University (i.e. University/Department/Course/Assignment) and relies on a distributed repository structure that is encapsulated in a three-tier architecture typical for Web-based applications. The employment of Web technology for the development of the proposed digital library is justified to a great extend from the fact that nowadays, the academic community is very keen to the Web. Consequently, little effort is required from the users of the digital library to adapt to this new environment. Moreover, access to the system can be established from virtually any kind of workstations, provided that a Web browser is previously installed. The above features dictate that DocML will be rapidly embraced from the academic community.

Another important feature of DocML is the employment of the emerging XML standard, which, according to related literature (Cole, 2000), has proved to support functionality that is common in the field of digital libraries. Specifically, XML is employed for the encoding and management of the various metadata elements of the proposed digital library as well as for the communication between the various software components.

RELATED WORK

A number of digital library technologies can be employed for the development of systems capable of managing large data repositories of similar content to the one presented in this paper. The importance of such applications has already been recognized from many major organizations and governments all over the world (Schauble, 1998). This section focuses on related digital library projects on the Web.

The architecture proposed by Gupta et al. in (Gupta, 1999) is based on data objects and links. Data objects constitute the underlying repository and may be related to other data objects via typed links. The authors claim that due to its navigational nature, this “object-oriented” approach is suitable for modeling scientific information.

A CGI-based, three-tier architecture of remote sensing archives that utilizes XML is proposed by Aloisio et al. in (Aloisio, 1999). CGI script technology is employed for the implementation of the various software components of this digital library. As compared to similar CGI-based architectures, the authors of this work dictate that XML-based services can be used in the context of previous requests facilitating this way richer human-computer interaction.

A neural network model is employed to facilitate access to distributed repositories according to the SOMLib approach (Rauber, 1999). Standard Web-based interfaces are incorporated and “computational intelligent tools” provide high-level of functionality including scalability, user-profiling and integrated searching. Special attention is given to the visualization of the information repository’s contents which, in turn, facilitates more efficient acquisition and retrieval of stored information.

THETIS (Houstis, 1996) is a Web-based, distributed environment consisting of one or more underlying repository nodes. Each node provides local information retrieval services to its content. According to this architecture, different metadata sets
describe the data objects contained in each node. Queries to the digital library are addressed to the intersection of these sets.

FEDORA (Payette, 1998) is another digital library based on the distributed model. This system takes under consideration many of the issues that are mentioned in the field of digital libraries. Based on the well-known dienst protocol (Lagoze, 1995), FEDORA proposes a generic digital object model and repository structure for encapsulating and securing multimedia content. It supports aggregation of distributed, heterogeneous types of data along with external rights management schemes. Extensibility of content types is provided as well as access to multiple views of the same data.

Finally, the Networked Digital Library of Theses and Dissertations (NDLTD) (Powel, 1998-Fox, 1997) in the US is an inter-University digital library based on the SGML-XML and Z39.50 standards. In this work, an argument in favor of XML is presented, claiming that it offers a great deal of flexibility compared to other alternatives such as storing document descriptions in a relational database.

**REQUIREMENTS FOR A DIGITAL LIBRARY OF UNIVERSITY DATA**

A number of specific requirements should be taken under consideration for the development of a Web-based digital library of University data. Thus, like any other application that employs Web technologies, such an effort must follow certain fundamental design principles [3]. Thus, implementation should be based on protocols and software tools that are well accepted from the Web community in order to maintain interoperability between the various components of the digital library. Furthermore, design principles like simplicity, modularity, and extensibility will increase the chances that the digital library will be able work with emerging technologies that are likely to dominate the Web in the near future. Last but not least, decentralization is a requirement that has recently emerged in the field of large-scaled, Web-based applications. In order to allow a digital library to scale without limitations while avoiding errors and breakdowns, the underlying architecture must limit dependencies on central registries.

Further requirements are posed from the fact that the proposed digital library consists of University data. Thus, such a digital library’s repository structure should be designed in a way that facilitates administration of large amounts of data items that are stored in a variety of formats. Many times, documents in a University collection may include multiple files of various formats. Moreover, access to the repository structure should be performed in an efficient way [5]. During our 2,5 years experience with S.K.E.P.S.I.S digital library of lecture notes [3], we have observed that requests for access to the digital library increase a lot a short period before each semester’s final examinations. A system that handles University data should therefore be able to cope with many requests that occur in a small period of time.

The digital library community has also identified a number of requirements that are applicable to any digital library application. Thus, interaction with users should be easy but powerful and information retrieval should be accurate. For this purpose, the employment of metadata facilitates a solution that has been tested through time in a number of digital library applications (Baldonado, 1997-lannella, 1997). Specifically, in a digital library environment, metadata are defined as data that are used to describe the content of the underlying repository structure. Moreover, the fact that a digital library is not a static application suggests that such an application cannot survive in time without proper maintenance and regular updating. However, the big size and diverse nature of its underlying content, dictate that such tasks should be based on automated procedures. Thus, separating content from functionality is a fundamental rule that accommodates the development of such
METADATA, XML AND DIGITAL LIBRARIES

As it has already been mentioned, digital libraries aim at providing ways of organizing and efficiently managing large data collections. In order to satisfy this goal, such applications should be based on repositories that can be efficiently accessed. In the context of well-structured data repositories, documents within digital libraries are most frequently accompanied by metadata. Metadata are defined as data that are used to describe such documents. In the more specific case of a Web-based digital library of University data, documents may vary from a simple ASCII text to a complex file structure consisting of files of various formats. It is therefore fair to expect from a digital library consisting of such documents, to rely on metadata for the development of more efficient information retrieval techniques. To be effective, metadata should follow adequate standards (Balciado, 1997). In this context, the ability of XML to represent text as an ordered hierarchy of content objects (Cole, 2000) and simultaneously embed metadata within such objects, renders this standard as a very powerful solution for designing a Web-based digital library of University data.

One could argue though that the fact that XML is a descendant of SGML predetermines a short and unsuccessful future for this technology at the Web. SGML has been around for more than 15 years and nonetheless never managed to dominate this environment. This was primarily due to the complexity of the language and the resulting overhead required using it. Having these thoughts in mind, XML did away much of the complexity of its ancestor and kept only the parts needed for use in the Web (Pardi, 1999). Consequently, it is just a matter of time before XML-based architectures like the one presented in this paper dominate the Web.

Another argument in favor of employing XML technology in Web-based digital libraries is that XML allows the existence of different views of the same data (Liechti, 1998), since it separates content from presentation. This is a very useful feature for digital libraries where most frequently, various user categories require different views of the same documents. User interfaces can also be easier maintained without having to access the actual data.

Moreover, interoperability between different components of a digital library can be easily established through the employment of XML and related standards. By defining a common format for data structures that are both human readable and computer understandable, complexity of computer-to-computer interaction is significantly reduced (Aloisio, 1999).

Having the above thoughts in mind, it is fair to claim that the employment of XML technology provides a simple and straightforward way of developing and managing robust and functional digital library applications.

THREE-TIER ARCHITECTURE

The proposed digital library is based on a three-tier architecture capable of handling XML data on the Web. The XML standard is employed to describe the documents that compose the digital library’s distributed repository structure. Such documents are essentially the deliverables from students within a University
environment throughout an academic year. With trivial modifications, DocML can be extended to manage University data of many kinds.

The back-end consists of a number of nodes that correspond to the various Universities that are members of the digital library. Each node maintains a repository that hosts the various documents of the corresponding University. The conceptual design of each node is based on the organizational structure of a University. Although documents rarely have anything in common as we shift from department to department, the user is presented with a homogeneous and highly organized repository structure. This is due to the fact that XML facilitates management of documents with diverse structure/content.

The middleware implements the core functionality of the proposed architecture. Two modules, namely the translator and the merger module are responsible for the communication (through a dedicated Web server) with the client and back-end layer respectively. Specifically, the translator module accepts and satisfies requests as obtained from end-users (usually students) while the merger module is responsible for collecting XML data describing the various documents that reside at the back-end. The XML data is then stored in the DocML database.

The client layer consists of a common Web browser that communicates with the middleware’s Web server. A general description of the proposed architecture is presented in figure 1, providing knowledge for the various components of the digital library.

**FIGURE 1:** The DocML architecture

**Back-end**

The back-end consists of a number of digital library nodes that correspond to the Universities that are members of DocML. As it can be observed in figure 1, each node maintains a Web server and a document repository that is mapped to a flat file system hierarchy of XML and other files. The overall structure of each node’s hierarchy is presented in figure 2 [4].
The root folder of each DocML node hierarchy is the "docml" folder and contains three XML files, namely repository.xml, universities.xml and sareas.xml. The repository.xml file has general information about the corresponding University. An example of such a file is presented in figure 3.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<repository>
    <id value="unipi"/>
    <url value="http://rainbow.cs.unipi.gr"/>
    <description value="University Of Piraeus"/>
    <comment>
        This is the university of piraues
    </comment>
    <image src="logo.gif"/>
    <contact value="webmaster@unipi.gr"/>
    <address value="80 Karaoli and Dimitriou Str"/>
    <tel value="4142231"/>
</repository>
```

FIGURE 3: A typical repository.xml file

The remaining two files, namely Universities.xml and sareas.xml are copied from the middleware where they are primarily maintained and contain information about the participating Universities and the scientific areas respectively. A typical sareas.xml file is presented in figure 4.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<sareas>
    <area id="1">
        <name value="Mathematics"/>
        <description value="Description"/>
    </area>
    <area id="2">
        <name value="physics"/>
        <description value="Description"/>
    </area>
</sareas>
```

FIGURE 4: A typical sareas.xml file
The root folder "docml" contains two sub-folders, namely "Prof" and "university". The former contains the staff.xml file, which holds data about the University's faculty and the latter contains the departments.xml file, which describes the various departments inside the University. Such a file is presented in figure 5.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<departments>
  <department id="cs" name="Dept. of Informatics" description="Description" url="http://www.cs.unipi.gr/"/>
  <department id="stat" name="Dept. of Statistics" description="Description" url="http://www.ode.unipi.gr/"/>
</departments>
```

FIGURE 5: A typical departments.xml file

The attribute "id" in the department tag is associated with the name of the next subfolder in the hierarchy. Thus, according to the XML structure illustrated in figure 5, there will be two subfolders, namely "cs" (i.e. department of Computer Science) and "stat" (department of Statistics). Each "department" folder contains a course.xml file, which describes the various courses that are being taught in each department. Similarly, each course contains a number of assignments (corresponding to the assignment.xml file) that the students must deliver and each assignment contains a list (corresponding to the submit.xml file) of the students that have submitted this assignment. The documents that have been delivered from each student reside in subfolders that are named after a serial number. By the term document we mean the files and/or folders that may be submitted to the digital library for a single student's assignment. Documents contain the actual data of the digital library. Each document in the collection may have multiple files of various formats that constitute it. Since these files may be in formats that don't describe their content, an additional XML file (i.e. deliverable.xml) accompanies each document. Thus, the deliverable.xml file mainly consists of metadata that provide description and references to the rest of the document contents (e.g. pdf, ppt, doc, etc). A typical deliverable.xml file is presented in figure 6.

```xml
<deliverable>
  <sarea name="Informatics"/>
  <supervisor id="cdoulig" name="Christos Douligeris" email="cdoulig@unipi.gr"/>
  <abstract>a brief abstract</abstract>
  <comment>general comment</comment>
    <author date="15-03-2001" name="Ioannis Papadakis" id="p95053" homepage="http://thalis.cs.unipi.gr/~jpap" email="jpap@unipi.gr"/>
    <author date="30-08-2001" name="Basilis Karakoidas" id="p97058" homepage="http://thalis.cs.unipi.gr/~bkarak" email="bkarak@unipi.gr"/>
  <data>
    <installation os="Windows 95/98/2000/NT">
      <file value="file.doc" mime="text/doc"/>
      <file value="file.exe" mime="application/zip"/>
    </installation>
    <installation os="Linux">
      <file value="docml.ps" mime="application/ps"/>
      <file value="docml.html" mime="text/html"/>
    </installation>
  </data>
</deliverable>
```

FIGURE 6: A typical deliverable.xml file
Note that the element “web” is employed to provide the location of the document’s online Web-based version.

Considering the distributed nature of the back-end layer, each document within the digital library should have a unique name (i.e. identifier). Thus, the name of each document is separated in five sectors, namely university name, department name, course name, assignment serial number and deliverable serial number. For example, consider a document located at the University of Piraeus (uni), at the department of Computer Science (cs), submitted to the course of digital libraries (dlib), being the third assignment for the semester (3) and the fifth in order of submission (5). The unique name of this document will then be “uni.pis.dlib.3.5”.

### Middleware

The middleware encapsulates the core functionality of the proposed digital library. Such functionality is provided through the employment of two software components namely the merger and collector module, a relational database namely the DocML database, an XSL engine and two XML files namely the universities.xml and sareas.xml files.

The various XML files that contain metadata for each document residing at the back-end are collected and stored through the merger module in the DocML database, as shown in figure 1. The merger module can be scheduled to visit each node of the digital library at times when DocML’s usage is minimal, reducing this way the risk of degrading the system’s overall performance.

The translator module is a software component that facilitates information retrieval for the users of the digital library. Thus, queries are obtained in XML format from the client layer as illustrated in figure 7.

```xml
<query version="1.0" encoding="UTF-8"/>
<query>
  <essay>
    <tag name="author" value="Karakoidas" next="AND"/>
    <tag name="author" value="Papadakis"/>
  </essay>
</query>
```

**FIGURE 7:** A simple query.xml file

The message in figure 7 describes a query for a deliverable that was submitted from authors “Papadakis” and “Karakoidas”. The translator module translates the queries in SQL and addresses the transformed message to the DocML database. The deriving results are embedded in XML files and associated with adequate XSL stylesheets. An XSL engine transforms the XML files in HTML format according to the directives included in their associated stylesheets. The resulting HTML files are finally transmitted through the employment of a dedicated Web server to the Web browser that triggered the initial request.

As it has already been mentioned, the universities.xml and sareas.xml files contain information about the participating Universities and the identified scientific areas respectively. Thus, the former file mainly contains administrative information about each university while the latter one is actually a list of the scientific areas that may be associated with each course described in the back-end repository structure. A typical sareas.xml file has been presented in figure 4. Finally, the DocML database keeps the metadata information of the back-end repository structure in a searchable Relational Database Management System (RDBMS). Consistency with the XML files
residing at the back-end is provided from the merger module, by periodically visiting the digital library’s nodes and updating the DocML database.

Thin Client

Interaction between the digital library and its users is performed through the Web. Thus, the Web server at the middleware accepts and forwards requests that are being issued from users to other components of the middleware. Users interact with the system from thin client workstations. Thin clients consist of a standard Web browser capable of rendering HTML. According to this strategy, the proposed digital library can be accessed from any workstation that incorporates a Web browser, irrespective of the underlying operating system.

SUPPORTED SERVICES

User Interface

The diverse nature of computer systems from which DocML can be accessed, dictates that special attention should be given to compatibility issues that arise from the employment of Web browsers coming from different vendors and/or operating systems. Thus, the user interface service is implemented in HTML (ver. 3.2), which is supported from all the major Web browsers currently in the market.

The core functionality of this service is provided by java servlets and scripts implemented in Javascript, which in turn facilitate information retrieval and content delivery for the proposed digital library, as described in the following sections. The employment of such a combination of server and client side programming techniques provides best performance for the described service.

Resource Discovery

Iannella et al. state in (Iannella, 1996) that "Resource Discovery is the term commonly used to refer to the exercise of locating, accessing, retrieving and managing relevant resources for a user". In this context, DocML facilitates resource discovery through searching, browsing and content delivery services.

Searching

Searching can be performed by submitting either a simple or an advanced query that follows certain criteria like author, date of submission, etc. The corresponding query terms are then matched against content-specific fields of the DocML database like scientific area, author and subject. The ability to maintain such information in the proposed digital library derives from the fact that the overall repository structure is based on XML documents. Such documents encapsulate content-specific metadata that are mapped to the middleware’s DocML database, facilitating this way the retrieval of more relevant search results. Following, the translator module creates a search results list in XML format. The deriving list is transformed to plain HTML through the employment of an adequate XSL stylesheet and delivered to the client. XSL technology is also employed to provide extended functionality to the search results list. Thus, users of the digital library are able to interact with the search results and define their own personalized way according to which they wish the retrieved documents to be filtered or ranked. For example, a user may decide to place at the top of the list the most recent documents instead of the most relevant ones. Finally, each document reference illustrated as a hyperlink within
the HTML output can be activated to lead (using XPath technology [2]) to its associated resource(es) residing at the back-end, as it has been shown in figure 2. The employment of XPath technology is boosting the digital library's interaction with its users. Some of the features supported by this technology are links that lead to multiple destinations, bi-directional linking, attached roles, descriptive titles to the various ends a link attaches and so on. The only drawback of such an approach is the lack of dominating tools on the market that will automate the procedure of creating and managing such interactive documents. However, it should be taken under consideration that although XML is a new and consequently not stable technology, it is already adopted from virtually all major companies that produce Web products.

**Browsing**

Users of the proposed digital library may browse the underlying document collection in two ways. According to the first strategy, links may be followed that correspond to the various levels of the digital library’s repository hierarchy, as it has been illustrated in figure 2. For example, in order to browse through the documents that belong to the course “Compilers”, users must initially visit the “University of Piraeus” link, then click on the “Computer Science” department and finally select the “Compilers” course to view the corresponding documents.

Alternatively, users may choose to classify the underlying document collection prior to browsing according to a set of predefined scientific areas. Such a "content centered" classification is facilitated through the employment of the sareas.xml file, which has been previously presented in figure 4. This XML structure contains the scientific areas that form the University's scientific horizon. Thus, each course of every department within the Universities is associated with one or more scientific areas. By selecting a specific scientific area, users actually focus resource discovery on a subset of the underlying document repositories. This way, the browsing process becomes faster and easier while at the same time, the searching process results in more accurate results. The ability to classify the document collection in a different way from the one that corresponds to the collection's physical structure derives from the fact that XML allows the existence of multiple views of the same data. Specifically, every XML document corresponds to a XML tree that can be manipulated in various ways using appropriate XSL stylesheets.

**Content delivery**

According to the nature of the documents that are referenced in the search results list, users can either view the corresponding files from their original location or, download them to their workstation for further process (figure 8). Files in XML format are immediately transformed to HTML from adequate XSL stylesheets upon submission and stored to the document repository. This way, abuse of system resources at the back-end is avoided. Such problems could arise in the case of on-the-fly transformations upon request of XML files to HTML from adequate XSL processors.
ADMINISTRATIVE ISSUES

A central authority (CA) is responsible for the admission of new members to the DocML digital library. Initially, the University wishing to participate in the digital library must send an appropriate request form to the CA. Upon acceptance, the CA sends a software application that is employed to setup the new node. The CA will then register the University by updating the universities.xml file.

At this point, it should be mentioned that both the universities.xml and the sareas.xml (corresponding to the scientific areas list) files are maintained by the CA at the middleware. For performance reasons, copies of these files are kept in each back-end node separately.

Consistency between the content of the DocML database at the middleware and the individual repository nodes at the back-end is provided from the merger module. Thus, the merger module updates the DocML database by periodically contacting each node and collecting information about new and/or updated local documents. This pull model (Marshai, 1999) that is applied for updating the DocML database is preferred from an equivalent push model because it is more efficient. Thus, the updating procedure can be scheduled to begin at times when the network traffic of the digital library is reduced. On the other hand, if some nodes of the digital library are unavailable at the time of the transaction, they won't be contacted.

The maintenance of each node of the digital library is performed locally by the University’s administrator with custom tools provided by the CA. Specifically, the administrator may add, remove and update documents in the local repository structure.

The programming language that is best suited to manage XML data and consequently implement the functionality of the DocML digital library is Java (JDK 1.3). Its platform-independent nature as well as its specialization to Web applications, renders Java as the most appropriate language for such a task. Another argument in favor of the employment of Java for the implementation of the DocML digital library is the existence of a number of Java-based XML-capable tools, which gives the ability to choose the most appropriate one for each specialized task.
CONCLUSIONS

We have presented DocML, a Web-based digital library system capable of handling student deliverables in a University environment. The proposed digital library manages documents of diverse content that are stored in a distributed repository structure in an efficient and scalable way. DocML faces common issues that are posed from the Web and digital library environment from an academic’s point of view. Documents within the digital library are described through a distributed metadata structure that is based on the XML standard. Such information is mapped to a centralized database (i.e. DocML database) facilitating this way more efficient information retrieval. At the University of Piraeus in Greece, we are currently in the process of implementing the proposed digital library. Our primary future work is to integrate DocML with an access control service as well as addressing a number of security issues.

However, building a digital library is not just a matter of employing the right technologies or the most sophisticated software components. Thus, the success of such a project depends highly on the involvement of communities that will benefit from its use. For example, administration and maintenance of DocML should be undertaken from students in the context of course assignments.

NOTES


2. XML Path Language (XPath) Version 1.0, Nov. 1999, URL: http://www.w3.org/TR/xpath

3. Fundamental design principles for the Web. W3C Consortium. URL: http://www.w3.org/Consortium/#web-design

4. Quoted labels imply fixed names while non-quoted labels are substituted from adequate names. For example, the label “university” may refer to “University of Piraeus”, the label “department” may refer to the “Computer Science” department and the “course” label may refer to the “Digital Libraries” course.


REFERENCES


