Search and Delivery of Standardized Learning Resources Based on SOAP Messaging and Native XML Databases

Xin Xiang, Ling Guo, Yuanchun Shi
State Key Lab of Intelligent Technology and Systems,
Computer Science Department, Tsinghua University, Beijing, P.R. China
{xiangx01@mails., guoling02@mails., shiyc@}tsinghua.edu.cn

Abstract

With the progress of Web-based learning technologies, standardized digital repositories and learning management systems are becoming prevalent over time. The heterogeneity in underlying databases and access methods, however, makes it difficult to share and exchange the learning resources between them. In this paper, we propose an architecture for the search and delivery of learning resources. Based on the SOAP transmission protocol, the architecture seeks to improve interoperability between heterogeneous E-Learning implementations. We also present a general-purpose query language as a building block of the architecture. The language provides a unified query interface for resource repositories, thereby shielding the users from the differences in underlying databases and metadata schemas. To highlight our design, an implementation using LOM, native XML database and XPath is presented. The last part of this paper discusses technical and pedagogical issues of concern regarding the launching of contents from within standardized LMSs..
***Keywords:** LOM, Dublin Core, SCORM, Metadata, Content Package, SOAP, XML, XPath*

1. Introduction

The recent approval of LOM (Learning Object Metadata) standard[1] by the IEEE-Standards Association and of Dublin Core Metadata Element Set[2] by ISO marked a new milestone in the field of metadata standardization. It is believed that in the near future, they are to be widely accepted by industry and academia alike.

Whereas metadata and content related E-Learning standards are intended to specify the description of metadata and content, the SCORM (Sharable Content Object Reference Model) model[3][4] extends these standards by further specifying the run-time environment of LMSs (Learning Management Systems), including APIs and data elements needed to launch the SCOs (Sharable Content Objects). With the maturity of these standards, digital repositories and LMSs conforming to them have been developed and deployed all over the world. Though these E-Learning standards are developed to achieve the global accessibility of learning resources, they provide no definition of the communication model for the exchange of learning resources between heterogeneous implementations. As a result, current E-Learning systems are largely isolated applications in terms of interoperability. As stated in the newly published IMS Digital Repositories Interoperability Specification[5], “Finding content, when there are multiple repositories of content to be searched, is a complex problem. The problem is further aggravated when the repositories have heterogeneous representations of meta-data and heterogeneous access methods.”

Investigations have been conducted to address the issue of searching disparate repositories for contents. For the communication between heterogeneous systems, the SOAP (Simple Object Access Protocol)[6] based communication model is considered an efficient and ease-to-use method in a Web-based E-Learning environment. Meanwhile, to hide the heterogeneity of repositories from users, a unified query language tailored to the search of learning resources is indispensable. A recent report on digital repositories released by ADL[7] discusses some standards for networked repository architectures and other important infrastructure technologies that may be useful for managing SCORM compliant contents. [8][9] present a reference model of the digital repository domain and a recommended binding technique using SOAP and XQuery[10]. [11] highlights the effectiveness of SOAP in connecting E-Learning applications. [12] also proposes a SOAP message format and a unified XML query structure used in the access and exchange of learning objects. [13] introduces an unstructured canonical attribute model containing most frequently used attributes in queries. It is used to hide the collection details and is mapped to its native equivalent upon query request.

Based on the reference model in [5], we propose an architecture that partially addresses the aforementioned problems. By using SOAP messaging and a general-purpose query language, the model enables the search and delivery of resources in a metadata representation and query language independent fashion. To highlight our design, an implementation using LOM, native XML database and XPath[14] is presented.

The remainder of this paper is organized as follows: section 2 introduces SOAP messaging and native XML databases, the two supporting technologies of learning resource repositories. Section 3 presents the architecture of the communication model, while the following sections, namely section 4, 5 and 6, cover implementation details,

https://doi.org/10.23106/dcmi.952107319

that is, the translation of query requests, the SOAP based transmission of queries and results, and the delivery of learning resources. Section 7 discusses technical and pedagogical issues of concern with respect to the launching of contents and the learning scenario of SCORM. Section 8 concludes this paper.

2. Supporting Technologies

2.1. Simple Object Access Protocol (SOAP) Messaging

The concept of service presented in [11][15] is very helpful. They propose that each application provides some kind of services, and applications share their services by messaging. Henceforth, aside from the existing metadata and content related E-Learning standards, some communication mechanism is indispensable in connecting standardized learning applications (services).

Considering the fact that Web (HTTP) is and will continue to be an important technological base of E-Learning, and that XML has become the mostly used binding technique of learning technology standards, it is straightforward to adopt the protocol capable of transmitting data encoded in XML through HTTP, that is, SOAP messaging.

As recommended in [5][7], the fundamental SOAP model is comprised of stateless one-way messaging peers in a decentralized distributed environment. SOAP does not specify semantics for the application-specific data it conveys, but rather provides a common framework for enabling application-to-application data exchange[7]. Furthermore, in an IMS DRI[5] compliant learning object repository, SOAP messages with attachments should be used to transmit an IMS compliant content package in fulfilling the "Submit" function. In short, SOAP messaging could address most needs arising in the search and delivery of standardized learning contents in an increasingly distributed environment.

2.2. Native XML Databases and XML Query Language

The SCORM content aggregation specification[3] comprises two models: a metadata model specifying the metadata elements of learning resources, and a content packaging model representing content structure. Both of them are hierarchical, which is convenient for representing data consisting of many elements and sub-elements.

XML is perfectly suited for representing hierarchical models, as exemplified by the LOM and content packaging XML binding specifications published by IMS[16][17], both of which are adopted in SCORM.

Although relational database products today provide built-in XML document and query support, native XML databases are arguably the best choice for metadata storage. The reasons are listed as follows[7]:

- XML documents could be stored in native XML databases in a natural and effective way without any data type mapping;
- Native XML databases preserve the physical structure of the original documents as well as comments, DTDs, etc;
- Native XML databases can store documents without knowing the XML schema or DTD;
- Native XML databases are relatively small, cheap and easy to deploy.

As far as query language is concerned, IMS recommends XQuery, but in current implementations, XPath is more frequently used. Though it is considered that XPath does not have enough expressiveness to function as a database query language, it continues to be popular and could be easily replaced with XQuery whenever necessary. Databases supporting the XPath query language include not only open source native XML databases such as Xindice[18] and 4Suite[19], commercial native XML databases such as Tamino[20], Ipedo[21] and GOXML DB[22], but relational database products like Oracle 8i and 9i as well.

3. Architecture

Basically, there is no one-size-fits-all architecture or framework that could address all the problems in the field of learning resource repository construction and standardized content delivery. The proposed architecture focuses on the effective sharing and exchange of standardized contents among heterogeneous repositories, learning management systems and clients in a decentralized environment.

The proposed architecture comprises three types of participants as depicted in Figure 1:

- *Learning Resource Repositories*

Standardized learning resource repositories provide massive storage for learning resources and metadata, and a uniform interface for query and delivery through SOAP. The repositories deal with two types of requests: the query requests searching for specific metadata, and the delivery requests asking for the actual content. The query requests, expressed in the general-purpose query language, are translated into native equivalents which will then be submitted to the local database. The resulting metadata records will be encoded in XML for SOAP transmission. Similarly, upon a delivery request, the learning resource is packaged in a SOAP message and returned.

- *Learning Management Systems*

LMSs play the role of SOAP clients and application servers simultaneously. As SOAP clients, they request metadata and contents from the repositories via SOAP messaging; and as application servers, they forward the clients' query and delivery requests to repositories, and prepare the returned metadata and contents for clients'

https://doi.org/10.23106/dcmi.952107319

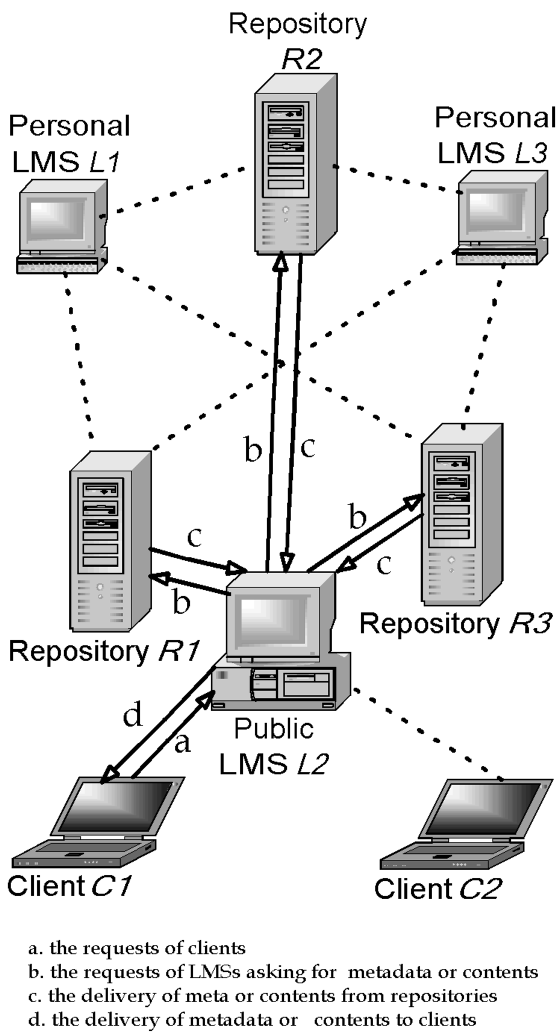

Figure 1. The Architecture of Content Search and Delivery.

browsing.

- *Clients*

Clients use common Web browsers to view the metadata information and the launched SCOs through HTTP sessions.

It is possible for a client and an LMS to reside in the same host, e.g. a PC may have a lightweight personal LMS (such as L1 or L3 in Figure 1) capable of requesting contents, launching SCOs, and providing application service to a local browser.

A typical learning scenario taking place in this architecture is described as below:

1. Client C1 logs on to LMS L2;
2. Client C1 issues a request for conformant contents on *data structure in computer science*;
3. LMS L2 forwards the request to repository R1, R2 and R3 in turn through SOAP messaging;
4. Metadata records satisfying the request are returned from the repositories, also through SOAP messaging. After being returned, they are cached in LMS L2;
5. Client C1 looks through the resulting metadata of contents and issues a request to download one of them;
6. LMS L2 forwards the request to the repository storing that content;
7. The repository delivers the requested content to LMS L2 by SOAP massaging with attachments;
8. LMS L2 launches the SCOs packaged in the delivered content from within its SCORM compliant run-time environment.

This architecture to some extent integrates existing heterogeneous learning resource repositories. Regardless of the underlying database and metadata schema of a repository, the only modification needed is to add a SOAP server and a translator tailored to its internal data structure, for the architecture, the unified query language, the messaging mechanism, the metadata information and the packaged contents are all platform and database neutral.

The LMSs play a key role in the scenario. It is the LMSs where most of the application logics of learning activities are performed, including search, evaluation, delivery and launch. To make a SCORM compliant LMS work in this scenario, the SOAP massaging mechanism needs to be implemented in addition to standard APIs.

The LMSs may vary in terms of size, performance, functionality and scalability. The lightweight LMSs, such as LMS L1 and LMS L3 in Figure 1, could be referred to as *Personal Learning Management Systems*. They are for personal use, functionally and structurally compact, and could be easily deployed on a home PC. On the other hand, *Public Learning Management Systems*, such as LMS L2 in Figure 1, have the full functionality of an LMS and are able to serve a group of users of varying sizes.

4. Translation of Query Requests

In [7], XQuery is the recommended query language to express query requests. Considering the heterogeneity in existing learning resource repositories, however, practitioners would rather have a unified query language than having to digest one for each implementation. In our architecture, the repository is responsible for translating the general query expression into its native equivalent, e.g. SQL for a relational database, and XPath or XQuery for a native XML database depending on the query language supported.

The process of handling query and delivery requests in a repository with a native XML database is depicted in Figure 2.

The general-purpose query language could be used to express simple queries with searchable attributes such as “title”, “language”, “description” and “keyword”. Logical and relational operations, for example “and”, “or”, “equals”, “not equals”, “greater than”, “less than”, “greater than or equals”, and “less than or equals”, are supported along with “contains”, an important keyword in query expressions.

https://doi.org/10.23106/dcmi.952107319

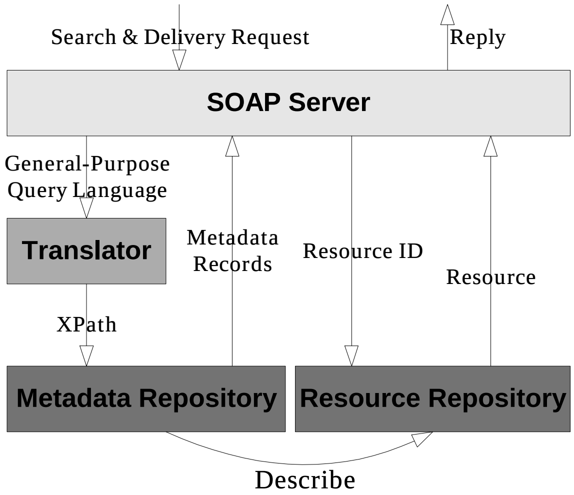

Figure 2. The Process of Handling Search & Delivery Requests in a Repository with a Native XML Database.

```
QUERY       ::= SUM EOL | EOL | EOF
SUM         ::= TERM ("OR" TERM)*
TERM        ::= UNARY ("AND" UNARY)*
UNARY        ::= "NOT" ELEMENT | ELEMENT
ELEMENT     ::= ATTRIBUTE
               ("EQUALS" | "CONTAINS" |
               "NEQ" | "LT" | "GT" |
               "LE" | "GE") VALUE |
                "(" SUM ")"
ATTRIBUTE ::= "CATALOG" | "ENTRY" |
               "TITLE" | "LANGUAGE" |
               "DESCRIPTION" |
               "KEYWORD" | "CREATOR" |
               "DATE" |
               "METADATASCHEMA" |
               "METADATALANGUAGE" |
                "FORMAT" |
               "LEARNINGRESOURCETYPE" |
               "COVERAGE" |
               "SUBJECT"
VALUE      ::= "'" (LETTER |
               ["0" - "9", " "])+ "'" |
                "\\"" (LETTER |
               ["0" - "9", " "])+ "\\""
LETTER     ::= ["-", "\\u0024",
                "\\u0041"-"\\u005a",
                 "\\u005f",
                "\\u0061"-"\\u007a",
                "\\u00c0"-"\\u00d6",
                "\\u00d8"-"\\u00f6",
                "\\u00f8"-"\\u00ff",
                "\\u0100"-"\\u1fff",
                "\\u3040"-"\\u318f",
                "\\u3300"-"\\u337f",
                "\\u3400"-"\\u3d2d",
                "\\u4e00"-"\\u9fff",
                "\\uf900"-"\\ufaff"]
EOL         ::= "\"
```

Figure 3. Formal Grammar of a General-Purpose Query Language.

The formal grammar of the general-purpose query language is given in Figure 3 using BNF notation.

As can be seen from the declaration of the token "ATTRIBUTE", this query language is designed to be schema-neutral by extracting the mostly used searchable elements defined in existing metadata schemas, such as Dublin Core, LOM and the core element set of CELTS 3.1 (a subset of the IEEE LOM model)[23].

The translation from this query language to XPath has been implemented through a LL(1) parser generated by JavaCC[24]. The rudimentary rule of the translation is to produce an XPath language element upon each derivation during the process of parsing. By this kind of translation, a query expression in this general-purpose query language:

```
TITLE CONTAINS “DATA STRUCTURE”
```

will produce something like the XPath expression below in a LOM conformant repository:

```
/lom[contains(general/title/langstrin
g, “DATA STRUCTURE”)]
```

and (as per the Dublin Core XML guidelines[25]) will generate the following expression in a repository complying with Dublin Core.

```
/dc:metadata[contains(dc:title, “DATA
STRUCTURE”)]
```

Similarly, by

```
DESCRIPTION CONTAINS “UNIX” AND LANGU
AGE EQUALS “en”
```

we will get

```
/lom[contains(general/description/lan
gstring, “UNIX”) and general/language=”e
n”]
```

in a LOM conformant repository, and

```
/dc:metadata[contains(dc:description,
“UNIX”) and dc:language=”en”]
```

in a DC conformant one.

Now that we have successfully implemented the XPath translator, its SQL and XQuery counterparts could be implemented in a similar approach.

This query language is certainly not a fully qualified database query language for it does not allow for joins or sorting of the query results, nor does it support recursive query. But the initiative is to equip LMSs with a lightweight and understandable query language that could

https://doi.org/10.23106/dcmi.952107319

be easily translated into widely used query languages such as SQL and XPath. It is relatively natural and powerful in comparison with the XML query structure specified in [12], and is intended to bridge the gap between a database-specific query language and the natural language. A query expression in this language could be conveniently composed through trivial combination of the input of a well-structured HTML form in an LMS.

Given its simple and scalable grammar, the general-purpose query language could easily be extended in terms of expressiveness.

5. SOAP Based Query Requests and Result Transmission

A query request expressed in the general-purpose query language is packaged in a SOAP envelope. As discussed before, it is left to the repository to perform the translation, so as to preserve the generality of its query interface.

Figure 4 shows a sample request.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://
schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <query>
      TITLE CONTAINS "Data Structure"
    </query>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 4. A SOAP Message Containing the Query Request.

Like the query requests, the resulting metadata records are also delivered in a SOAP envelope as demonstrated in Figure 5.

6. Delivery of Learning Resources

After having retrieved the resulting metadata records, the user may go over the metadata information, and ask for one or more resources of interest.

Since practicable global identification systems of digital learning resources are outside the scope of this paper, a locally unique identifier is used here for demonstration. If possible, it can be replaced by any global identifier in use.

A user requests a learning resource by sending the locally unique ID to the LMS which forwards the request to the repository containing the requested resource through SOAP messaging as depicted in Figure 6.

The requested content is delivered through SOAP massaging with attachments.

The reasons for choosing SOAP for content delivery as opposed to using HTTP or FTP proposed in IMS DRI –

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://
schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <lom xmlns:src="http://
     xml.apache.org/xindice/Query"
     src:col="/db/metadata"
src:key="54466ea634ce5a11000000f4ce48647
4">
      <general>
        <identifier>
          <catalog>Local ID</catalog>
          <entry><langstring>
            4
          </langstring></entry>
        </identifier>
        <title>
          <langstring xml:lang="en">
            Data Structure in C++
          </langstring>
        </title>
        <language>en</language>
        <description>
          <langstring
           xml:lang="en">
            A Comprehensive C++ Data
Structures Programming Course.
          </langstring>
        </description>
        <keyword>
          <langstring xml:lang="en">
            Programming Language
          </langstring>
        </keyword>
        <keyword>
          <langstring xml:lang="en">
            Data Structure
          </langstring>
        </keyword>
      </general>
      ......
    </lom>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 5. A SOAP Message Containing the Resulting Metadata Record.

Core Functions Best Practice Guide Specification[9] are stated as follows:

1. SOAP is based on HTTP, hence is easy to implement, platform independent and firewall-friendly;
2. The SOAP implementation hides the detail of the repositories from the LMSs, thereby providing a uniform access method of learning contents;
3. As metadata and contents are transferred in a common mechanism, system consistency and compactness are enhanced.

https://doi.org/10.23106/dcmi.952107319

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://
schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:request
     soapenv:encodingStyle="http://
     schemas.xmlsoap.org/soap/encoding/"
     xmlns:ns1="urn:DeliverService">
      <source xsi:type="ns1:String">
        4
      </source>
    </ns1:request>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 6. A SOAP Message Containing the Requested Resource ID.

A SOAP message snippet containing the delivered content is shown in Figure 7.

```
------=_Part_2_3521712.1052464544911
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: binary
Content-Id:
<3FC4689165C63FB9F92319D360373000>

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://
 schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <ns1:requestResponse
     soapenv:encodingStyle="http://
     schemas.xmlsoap.org/soap/encoding/"
     xmlns:ns1="urn:DeliverService">
      <returnqname
href="cid:48613F8DDDF74EB4237538A2DA647F
77" xmlns:ns2="http://xml.apache.org/
xml-soap"/>
    </ns1:requestResponse>
  </soapenv:Body>
</soapenv:Envelope>
------=_Part_2_3521712.1052464544911
Content-Type: application/octet-stream
Content-Transfer-Encoding: binary
Content-Id:
<48613F8DDDF74EB4237538A2DA647F77>

......CONTENT......

------=_Part_2_3521712.1052464544911--
```

Figure 7. A SOAP Message Containing the Content.

The two bold parts of the snippet represent the SOAP message body and the attached content respectively.

After being delivered, the standard content is unzipped by the requesting LMS. If the content is a SCORM compliant courseware package, a persistent object (database or disk file) will be built to maintain the persistence of run-time environment data elements implemented by the LMS. Moreover, the LMS will extract the course structure by parsing the standard manifest file wrapped in the content package, and the sequencing method will be determined depending on the sequencing mechanism implemented by the LMS.

On completion of the actions above the course is ready for future launch.

7. Launching of SCOs from within SCORM Compliant Run-Time Environment in Public and Private Learning Management Systems

Upon the request of launching, the relevant SCO is launched as per the SCORM run-time environment specification[4].

A comprehensive public learning management system should implement the following function regarding SCO launching:

- *Fully Functional API Adapter*

The LMS should supply an API adapter that implements the required functionality described in [4].

- *Comprehensive Data Elements*

The LMS should implement most data elements defined in [4], including elements describing student information, learner performance and comment data. By doing so, it enables full support of the SCORM run-time environment data model and utmost trackability of information about the launched SCOs.

- *Persistence Through Database*

Database-enabled persistence mechanism should be provided to maintain the persistence of the complex data elements implemented by the LMS. The database could extensively and efficiently record the learner information, SCO information and learner performance, thereby fulfilling the responsibility of a public LMS.

A personal learning management system, however, need not implement as much functionality as a public learning management system does.

- *Simple API Adapter*

A personal LMS should supply a simple API adapter that implements the basic functionality during launching of SCOs, for instance, the *LMSInitialize* and *LMSFinish* methods described in [4].

- *Reduced Data Elements Set*

As proposed in [26], the set of data elements implemented by a personal LMS might be selectively reduced in order to improve the performance of LMS while maintaining its basic functionality.

- *Lightweight Persistence Mechanism*

https://doi.org/10.23106/dcmi.952107319

Since a personal LMS cares little about the management of the learner information, the detailed SCO information and the evaluation of learner performance, it is recommended that the basic disk file-based persistence mechanism be implemented, making the LMS small, efficient and easy to deploy.

Although SCORM claims to be pedagogically neutral, the learning scenario of launching SCOs in a Web-based run-time environment has been deemed individual-centric, even in a pubic learning management system environment. The purpose of the differentiation of personal LMSs from public ones is to adapt the LMSs in different learning environments, thus avoiding a one-size-fits-all solution in this rapidly changing E-Learning application market. And it is the responsibility of LMS vendors to tune their products, not only to fit training use, but also to meet the progressive need for collaborative learning.

8. Conclusion and Prospect

In this paper, we propose a distributed learning resource search and delivery architecture using SOAP messaging. In the architecture, a general-purpose query language is designed as a common query interface of learning resource repositories. Both the architecture and the query language are independent of platform, database and metadata schema, seeking to achieve interoperability between heterogeneous repositories found in current practice of E-Learning applications.

At this time, a campus-wide E-Learning application conforming to the proposed architecture, with Xindice as its XML repository, Axis[27] as the SOAP implementation, and WebLogic as the application server, is being developed in Tsinghua university. A public learning management system with extended data element set based on Oracle 8i database has been implemented to fit the need of college education.

The architecture may be improved in the following aspects:

- *Introduction of Subscription/Alert Mechanism*

As a possible future direction of the IMS DRI project, the subscription/alert function is an effective method to facilitate search and exchange of learning resources. However, the simple communication model might be complicated by the introduction of a new protocol (for example, SMTP) used to implement the function.

- *Adoption of DOI*

The Digital Object Identifier System[28], as proposed in IMS DRI Specification[9], is a possible future location service of digital learning resources. It could be used as a globally unique identifier system for resolving the location of the learning objects in a network of distributed repositories as opposed to the much more limited locally unique identifier system demonstrated in section 6.

- *Support for XQuery Language*

As a matter of fact, XPath is currently the mainstream native XML database query language. However, it is believed that in the near future, the relatively powerful and advanced XQuery language is to be supported by major database vendors, learning resource repository developers and academia. Hence a translator will be implemented to turn the general-purpose query language into XQuery.

Acknowledgements

This material is based upon work supported by the Ministry of Education and the National Natural Science Foundation in China.

References

[1] IEEE Learning Technology Standard Committee Working Group 12, "Final 1484.12.1 LOM draft standard", http://ltsc.ieee.org/doc/wg12/LOM_1484_12_1_v1_Final_Draft.pdf, September 2002.

[2] Dublin Core Metadata Initiative, "Dublin Core Metadata Element Set, Version 1.1", http://dublincore.org/usage/terms/dc/current-elements/, October 2002.

[3] Advanced Distributed Learning, "The SCORM Content Aggregation Model, Version 1.2", http://www.adlnet.org/ADLDOCS/Documents/SCORM_1.2_CAM.pdf, October 2001.

[4] Advanced Distributed Learning, "The SCORM Run-Time Environment, Version 1.2", http://www.adlnet.org/ADLDOCS/Document/SCORM_1.2_RunTimeEnv.pdf, October 2001.

[5] IMS Global Learning Consortium, "IMS Digital Repositories Interoperability – Core Functions Information Model Version 1.0 Final Specification", http://www.imsglobal.org/digitalrepositories/driv1p0/imsdri_infov1p0.html, January 2003.

[6] Box Don, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, D. Winer, "Simple Object Access Protocol (SOAP) 1.1", http://www.w3.org/TR/SOAP/, May 2000.

[7] Advanced Distributed Learning, "The Emerging and Enabling Technologies for the Design of Learning Object Repositories Report", http://www.adlnet.org/adldocs/Documents/ADLRepository TechnicalInvestigationReport.doc, November 2002.

[8] IMS Global Learning Consortium, "IMS Digital Repositories Interoperability – Core Functions XML Binding Version 1.0 Final Specification", http://www.imsglobal.org/digitalrepositories/driv1p0/imsdri_bindv1p0.html, January 2003.

[9] IMS Global Learning Consortium, "IMS Digital Repositories Interoperability – Core Functions Best Practice Guide Version 1.0 Final Specification", http://www.imsglobal.org/digitalrepositories/driv1p0/imsdri_bestv1p0.html, January 2003.

[10] Boag, S., D. Chamberlin, M. F. Fernandez, D. Florescu, J. Robie, and J. Siméon, "XQuery 1.0: An XML Query Language", http://www.w3.org/TR/xquery/, May 2003.

https://doi.org/10.23106/dcmi.952107319

[11] Wilson Scott, “Gluing learning applications together with SOAP”, http://www.cetis.ac.uk/content/20011015153128, October 2001.
[12] The Learning Federation, “Learning Object Repository Access and Exchange Web Service Specification”, http://www.thelearningfederation.edu.au/repo/cms2/tlf/published/3859/docs/SOAP_Specification_V0_4.pdf, January 2003.
[13] Michelle Baldonado, Chen-Chuan K. Chang, Luis Gravano, Andreas Paepcke, “The Stanford Digital Library metadata architecture”, *International Journal on Digital Libraries, Volume 1, Number 2*, September, 1997.
[14] Clark James, S. DeRose, “XML Path Language (XPath) Version 1.0”, http://www.w3.org/TR/xpath/, November 1999.
[15] Wilson Scott, “The next big thing? Three architectural frameworks for learning technologies”, http://www.cetis.ac.uk/content/20010828163808, August 2001.
[16] IMS Global Learning Consortium, “IMS Learning Resource Meta-data XML Binding Version 1.2.1 Final Specification”, http://www.imsglobal.org/metadata/imsmdv1p2p1/imsmd_bindv1p2p1.html, September 2001.
[17] IMS Global Learning Consortium, “IMS Content Packaging XML Binding Version 1.1.2 Final Specification”, http//www.imsglobal.org/content/packaging/cpv1p1p2/imscp_bindv1p1p2.html, August 2001.
[18] http://xml.apache.org/xindice/
[19] http://4suite.org/
[20] http://www.softwareag.com/tamino/
[21] http://www.ipedo.com/
[22] http://www.xmlglobal.com/prod/db/
[23] Xin Xiang, Zhongnan Shen, Ling Guo, Yuanchun Shi, “Introduction of the Core Elements Set in Localized LOM Model”, http://lttf.ieee.org/learn_tech/issues/january2003/index.html#6, *IEEE Learning Technology Newsletter*, Volume 5, Issue 1, January 2003.
[24] http://www.webgain.com/products/java_cc/
[25] http://dublincore.org/documents/dc-xml-guidelines/
[26] Qu Changtao, Nejdl Wolfgang, “Towards Interoperability and Reusability of Learning Resources: a SCORM conformant Courseware for Computer Science Education”, *Proc. of 2nd IEEE International Conference on Advanced Learning Technologies*, September 2002.
[27] http://ws.apache.org/axis/
[28] http://www.doi.org

https://doi.org/10.23106/dcmi.952107319