Extending CMIS Standard for XML Databases

Mihai Stancu*

**Faculty of Mathematics and Computer Science, Department of Computer Science, University of Craiova, Romania (e-mail: mihai.stancu@yahoo.com)*

Abstract: XML is getting very popular as data exchange standard format in web applications due to its portability and information exchange features. When applications must communicate with each other or must integrate information from several data sources, this standard is very useful. However, querying the XML data raise some efficiency problems because of its hierarchical structure and text representation. Over the last years many researches have been done to establish the optimal way of storing XML data in a relational model for taking advantage of the query processing abilities that a relational model can bring. Thus, XML databases have emerged trying to respond to those needs. When we speak of a higher level application, ECM Systems are the standard for developing applications that manage all the unstructured electronic information inside an organization. Over the past years a new information exchange standard has been developed, called CMIS, that unify the data exchange format between the existing ECM Systems. This paper shows how the CMIS standard can be extended to make possible the information exchange operations between ECM Systems and XML Databases. There is also proposed a CMIS interface implementation for a particular XML database, namely EMC xDB. Some performance tests of this implementation have been done.

Keywords: XML, information exchange, document management.

1. INTRODUCTION

Today, business environment can no longer afford to store essential documents into separate and incompatible repositories. Thus, major ECM vendors, seeking to revolutionize the interoperability of the ECM systems, developed a new standard - CMIS (Content Management Interoperability Services). This standard offers a common way that different applications can use for sharing and exchanging information between multiple ECM systems.

In the last years, the need for data exchange software has grown considerably. XML is the de facto standard for data exchange and many applications rely on XML databases to facilitate data exchange operations. In these circumstances, extending the CMIS standard to support XML databases can be an advantage for such collaborative applications.

The rest of this paper is organized as follows. Section 2 offers a description of the XML databases. Section 3 provides a short overview of ECM systems highlighting the main features of these systems. In section 4 is described the CMIS standard, its constituent parts and how that can be used to achieve information exchange operations. A CMIS standard eXtent for XML databases is described in detail in section 5 which presents a particular case of implementation of CMIS standard for EMC Documentum repository, as the ECM system, and EMC xDB database, as the XML database. Section 6 shows some performance experiments that have been done for the proposed CMIS eXtent.

2. XML DATABASES

There are two major classes of XML database:

- XML-enabled: these map all XML to a traditional database (such as a relational database), accepting XML as input and rendering XML as output. This term implies that the database does the conversion itself (as opposed to relying on middleware).
- Native XML (NXD): the internal model of such databases depends on XML and uses XML documents as the fundamental unit of storage, which is, however, not necessarily stored in the form of text files.

In this study we will focus on Native XML Databases this alternative being more close to the ECM System.

The term "native XML database" (NXD) can lead to confusion. Many NXDs do not function as standalone databases at all, and do not really store the native (text) form. The formal definition from the XML:DB initiative [Borden et al., 2003] states that a native XML database:

- Defines a (logical) model for an XML document - as opposed to the data in that document - and stores and retrieves documents according to that model. At a minimum, the model must include elements, attributes, PCDATA, and document order. Examples of such models include the XPath data model, the XML Infoset, and the models implied by the DOM [Apparao et al., 1998] and the events in SAX 1.0 [Megginson et al., 2000].
- Has an XML document as its fundamental unit of (logical) storage, just as a relational database has a row in

a table as its fundamental unit of (logical) storage.

- Need not have any particular underlying physical storage model. For example, NXDs can use relational, hierarchical, or object-oriented database structures, or use a proprietary storage format (such as indexed, compressed files).

Additionally, many XML databases provide a logical model of grouping documents, called "collections". Databases can set up and manage many collections at one time. In some implementations, a hierarchy of collections can exist, much in the same way that an operating system's directory-structure works.

All XML databases now support at least one form of querying syntax. Minimally, just about all of them support XPath for performing queries against documents or collections of documents. XPath provides a simple pathing system that allows users to identify nodes that match a particular set of criteria.

In addition to XPath, many XML databases support XSLT as a method of transforming documents or query-results retrieved from the database. XSLT provides a declarative language written using an XML grammar. It aims to define a set of XPath filters that can transform documents (in part or in whole) into other formats including Plain text, XML, or HTML.

Many XML databases also support Xquery [Boag et al., 2003] to perform querying. XQuery includes XPath as a node-selection method, but extends XPath to provide transformational capabilities. Users sometimes refer to its syntax as "FLWOR" (pronounced 'Flower') because the query may include the following clauses: 'for', 'let', 'where', 'order by' and 'return'. Traditional RDBMS vendors (who traditionally had SQL only engines) are now shipping with hybrid SQL and XQuery engines. Hybrid SQL/XQuery engines help to query XML data alongside the relational data, in the same query expression. This approach helps in combining relational and XML data.

Some XML databases support an API called the XML: DB API (or XAPI) as a form of implementation-independent access to the XML data store. In XML databases, XAPI resembles ODBC and JDBC as used with relational databases. On the 24th of June 2009, The Java Community Process released the final version of the XQuery API for Java specification (XQJ) - "a common API that allows an application to submit queries conforming to the W3C XQuery 1.0 specification and to process the results of such queries" [JCP, 2009].

3. ENTERPRISE CONTENT MANAGEMENT

Enterprise Content Management (ECM) is the strategies, methods and tools used to capture, manage, store, preserve, and deliver content and documents related to organizational processes [AIIM, 2001]. ECM tools and strategies allow the management of an organization's unstructured information, wherever that information exists.

Thus, ECM is a system that covers the company’s needs. To accomplish these issues, an ECM system uses its components that can be grouped in the following areas:

- Capture: the capture process involves the conversion of the information from the physical format to the electronic format by using scanning tools;
- Management: depending on the managed content, this process can include components as Document Management (DM), Collaboration, Web content management, Records management, Workflow and business process management (BPM);
- Storage: the storage process deals with the information that is not classified for conservation yet, but it must remain persistent for a certain period of time.
- Conservation: ECM systems provide a long-term safe preservation of the information;

Distribution: in any ECM system there are components responsible for the information export to various external sources.

4. CMIS – AN INTEROPERABILITY STANDARD FOR ECM SYSTEMS

To revolutionize enterprise content management interoperability and address this major business challenge, EMC, IBM, and Microsoft are leading the charge together on developing Content Management Interoperability Services. Content Management Interoperability Services is a proposed standard consisting of a set of web services for sharing information among disparate content repositories that seeks to ensure interoperability for people and applications using multiple content repositories. Working together since late 2006, the three companies were joined in the creation of the Content Management Interoperability Services draft specification by other leading enterprise content management providers, including Alfresco Software, OpenText, Oracle, and SAP. The three companies have jointly submitted the standard to OASIS to initiate the technical development process that leads to standardization.

4.1 Definition

Content Management Interoperability Services (CMIS) is a specification for improving interoperability between Enterprise Content Management systems. OASIS approved CMIS as an OASIS Specification on May 1, 2010.

CMIS uses Web services and Web 2.0 interfaces to enable rich information to be shared across Internet protocols in vendor-neutral formats, among document systems, publishers and repositories, within one enterprise and between companies.

CMIS provides a common data model covering typed files, folders with generic properties that can be set or read. In addition there may be an access control system, and a checkout and version control facility, and the ability to define generic relations. There is a set of generic services for modifying and querying the data model, and several protocol bindings for these services, including SOAP and Representational State Transfer (REST), using the Atom convention. The model is based on common architectures of document management systems.

4.2 Objectives

The main objectives of CMIS are:

- To ensure interoperability for application among disparate content repositories across all platforms
- Decouple repository-access logic from application
- Encapsulate repository-specific logic behind standardized services
- Easy mapping to existing ECM repositories
- Does not require behavioral change in server
- Focus on core capabilities common to most ECM systems
- A single domain model capable to support multiple protocols
- Simple Object Access Protocol (SOAP) / WSDL
- Representational State Transfer (REST) / Atom
- Exploit Web technologies
- Web 2.0, firewall tunneling, Internet scale
- Service-orientation, resource-orientation.

4.3 CMIS components

CMIS standard includes the following components:

- A content management domain-specific data model
- A set of basic services that act on the data model
- Protocol bindings for these basic services
- Simple Object Access Protocol (SOAP) / WSDL
- Representational State Transfer (REST) / Atom

More details about the CMIS components can be found in [AIIM, 2009] [Stancu, 2011a].

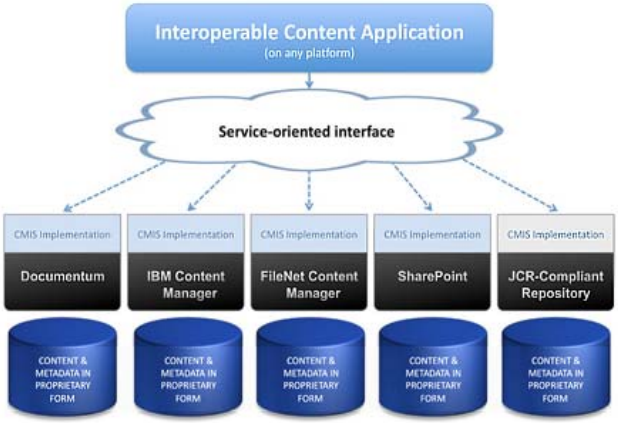


Fig. 1. CMIS collaboration scheme.

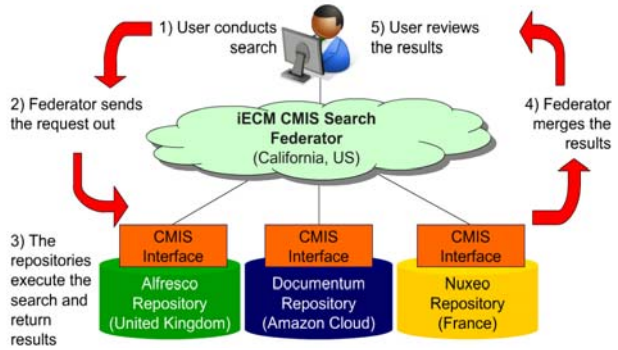


Fig. 2. Informational flow in CMIS Demo application.

4.4 CMIS Demo

The AIIM's iECM Committee validated CMIS standard by implementing a CMIS demo application with support from some of the most important ECM vendors: Alfresco, EMC, IBM, Nuxeo, Exo Platform. They implemented a federated search application that brings documents from different repositories to the end user interface by using CMIS collaboration standard [AIIM, 2009].

5. EXTENDING CMIS FOR XML DATABASES

This section will present a collaboration scenario in which certain ECM repositories can be involve and we will try to add an XML Database that will accessed via CMIS standard.

5.1 Collaboration perspective

As a collaboration case, we have used the federated search application, provided by the AIIM's iECM Committee as a CMIS demo application, and we have try to plug in a new data source that is the EMC xDB XML database. In our sample application, for demonstration purposes, we have simplified the case be using only the EMC Documentum repository as the ECM repository involved in the collaboration scenario.

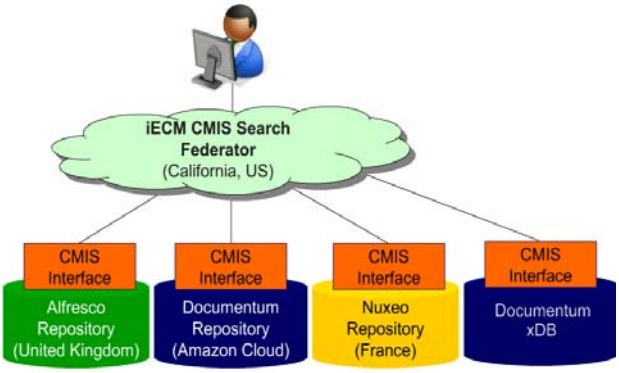


Fig. 3. CMIS collaboration scenario.

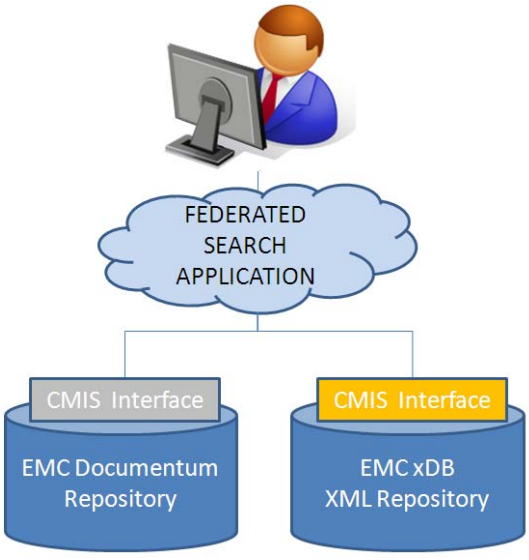


Fig. 4. Documentum-xDB collaboration perspective

For each ECM repository, a CMIS interface is provided by the ECM vendor, so the new XML Database will need also a CMIS interface that will provide the data access in the CMIS format. This CMIS interface was created by implementing the Web Services described by the CMIS standard. This Web Services can respond to some specific CMIS SOAP requests, run XML repository specific code to access the data and return the information as SOAP response to the CMIS client.

5.2 Components model

In order to properly respond to CMIS requests, the CMIS interface have to rely on some existing data model from XML Repository. Internal to the CMIS interface, a DataMapper is used to map the CMIS data model to the repository specific data model. After the data model is obtained, eventually, a QueryTranslator is used to transform the CMIS specific query into XQuery format. A sample of queryable data in xDB database is shown in figure 6. This data will have to be validated against the dtd file shown in Fig. 7.

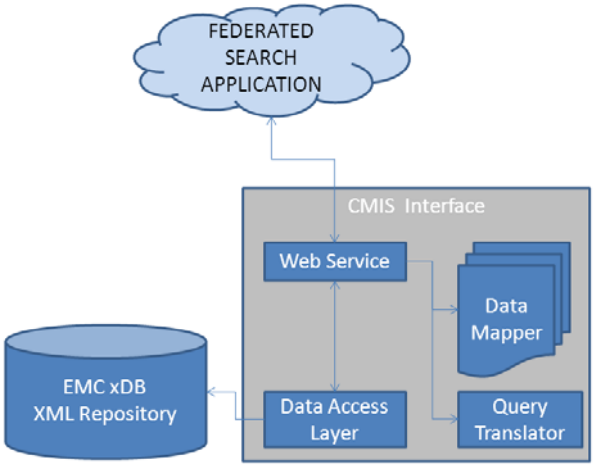


Fig. 5. Main components of the CMIS interface.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ecm-documents SYSTEM "ecm-xdb.dtd">
<ecm-documents>
    <ecm-document ecmObjectId="123">
        <ecmObjectTypeId>ecm-document</ecmObjectTypeId>
        <ecmCreatedBy>test_user</ecmCreatedBy>
        <ecmCreationDate>2011-08-10 09:10:12</ecmCreationDate>
        <ecmLastModifiedBy>test_user</ecmLastModifiedBy>
        <ecmLastModificationDate>2011-08-10 09:10:12</ecmLastM
        <ecmName>Sample doc 1</ecmName>
        <ecmIsImmutable>false</ecmIsImmutable>
        <ecmVersionLabel>
            <labels>
                <label>1.0</label>
            </labels>
        </ecmVersionLabel>
        <ecmVersionSeriesId>123</ecmVersionSeriesId>
        <ecmVersionSeriesCheckedOutBy/>
        <ecmContentStreamLength>0</ecmContentStreamLength>
        <ecmContentStreamMimeType/>
        <ecmContentStreamId/>
    </ecm-document>
    <ecm-document ecmObjectId="234">
        ...
```

Fig. 6. Queriable XML data in xDB.

The data-mapping xml file for xDB map the CMIS types to specific xml elements from repository and within each type mapping, an attribute mapping is used to for each attribute from CMIS type. These mappings are described in figure 8. Along with these data type mappings, there are configured also the capabilities supported or not by the CMIS interface. From the main mapping configured in this file we can mention

<config:cmisType> - data type mapping definition
<cmis:id> - specify the CMIS element identifier
<cmis:localName> - name of the data type / property from xDB
Capabilities descriptors: <cmis:queryable>, <cmis:fileable>
etc.

However, in the XML repository may exist some XML documents that can be expose to CMIS exchange layer but those documents do not share the same format described above. In such cases, additional transformation layer can be used in order to transform the existing XML documents from repository into some XML documents that have the CMIS compliant structure. Such a transformation layer can be easily implemented by using XSL transformation files. Once the CMIS interface is build, existing XML documents from repository that have various structures can be exposed to CMIS exchange layer by plugging in the corresponding XSL transformation files.

The prototype platform of the CMIS interface with support for xDB database provides also a web application as the interface for data querying. Figure 9 shows the federated search form and the returned results.

```
<?xml encoding="UTF-8"?>
<!ELEMENT ecm-documents (ecm-document)+>
<!ATTLIST ecm-documents xmlns CDATA #FIXED ''>

<!ELEMENT ecm-document (ecmObjectTypeId,ecmCreatedBy,
    ecmCreationDate,ecmLastModifiedBy,
    ecmLastModificationDate,ecmName,
    ecmIsImmutable,ecmVersionLabel,
    ecmVersionSeriesId,ecmVersionSeriesCheckedOutBy,
    ecmContentStreamLength,ecmContentStreamMimeType,
    ecmContentStreamId)>
<!ATTLIST ecm-document
  xmlns CDATA #FIXED ''
  ecmObjectId CDATA #REQUIRED>

<!ELEMENT ecmObjectTypeId (#PCDATA)>
<!ATTLIST ecmObjectTypeId xmlns CDATA #FIXED ''>

<!ELEMENT ecmCreatedBy (#PCDATA)>
<!ATTLIST ecmCreatedBy xmlns CDATA #FIXED ''>

<!ELEMENT ecmCreationDate (#PCDATA)>
<!ATTLIST ecmCreationDate xmlns CDATA #FIXED ''>

<!ELEMENT ecmLastModifiedBy (#PCDATA)>
<!ATTLIST ecmLastModifiedBy xmlns CDATA #FIXED ''>

<!ELEMENT ecmLastModificationDate (#PCDATA)>
<!ATTLIST ecmLastModificationDate xmlns CDATA #FIXED ''>
```

Fig. 7. DTD file describing XML documents from xDB.

```
<config:cmisType
    xsi:type="cmis:cmisTypeDocumentDefinitionType">
    <cmis:id>cmis:document</cmis:id>
    <cmis:localName>dm_document</cmis:localName>
    <cmis:displayName>Document</cmis:displayName>
    <cmis:queryName>cmis:document</cmis:queryName>
    <cmis:description>Document</cmis:description>
    <cmis:baseId>cmis:document</cmis:baseId>
    <cmis:creatable>true</cmis:creatable>
    <cmis:fileable>true</cmis:fileable>
    <cmis:queryable>true</cmis:queryable>
    <cmis:fulltextIndexed>true</cmis:fulltextIndexed>
    <cmis:includedInSupertypeQuery>true</cmis:includedIn
    <cmis:controllablePolicy>false</cmis:controllablePol
    <cmis:controllableACL>true</cmis:controllableACL>
    <cmis:versionable>true</cmis:versionable>
    <cmis:contentStreamAllowed>allowed</cmis:contentStre
    <cmis:propertyIdDefinition>
        <cmis:id>cmis:objectId</cmis:id>
        <cmis:localName>r_object_id</cmis:localName>
        <cmis:displayName>Object Id</cmis:displayName>
        <cmis:queryName>cmis:objectId</cmis:queryName>
        <cmis:description>Object Id</cmis:description>
        <cmis:propertyType>id</cmis:propertyType>
        <cmis:cardinality>single</cmis:cardinality>
        <cmis:updatability>readonly</cmis:updatability>
```

Fig. 8. CMIS - xDB mapping configurations.

However, in the XML repository may exist some XML documents that can be expose to CMIS exchange layer but those documents do not share the same format described above. In such cases, additional transformation layer can be used in order to transform the existing XML documents from repository into some XML documents that have the CMIS compliant structure. Such a transformation layer can be easily implemented by using XSL transformation files. Once the CMIS interface is build, existing XML documents from repository that have various structures can be exposed to CMIS exchange layer by plugging in the corresponding XSL transformation files.

The prototype platform of the CMIS interface with support for xDB database provides also a web application as the interface for data querying. Figure 9 shows the federated search form and the returned results.

6. PERFORMANCE STUDIES

In these experiments was used the Neoload performance tool with a population of 10 virtual users added gradually, two users at every 12 seconds. The experimental scenario included three CMIS queries called in ascending order of the complexity, as can be seen in Figure 10.

ID	Nume	Titlu	Subiect	Autor	Data crearii	Tip repository
ADCB1287	CaietSar-2011-05-07	Caiet de sarcini - mai 2011	docCS	User1	07.05.2011 15:27	xDB
FDBB7639	SpecFunc-2011-06-10	Specificatii functionale - iunie 2011	docSF	User1	10.06.2011 15:29	xDB
BADD2538	SolDes-2011-07-11	Solution Design - iulie 2011	docSD	User1	11.07.2011 15:30	xDB
FACB8440	ManInst-2011-08-10	Manual de instalare - august 2011	docMan	User2	10.08.2011 15:34	xDB
DACB3285	ManUtil-2011-08-10	Manual de utilizare - august 2011	docMan	User2	10.08.2011 15:36	xDB
DDFB3519	ManAdm-2011-08-10	Manual de administrare - august 2011	docMan	User2	10.08.2011 15:37	xDB
0900000280	Bilant 2010	Bilant economic 2010	Economie	User1	12.10.2011 17:09	Documentum
0900000280	Bilant 2011	Bilant economic 2011	Economie	User1	12.10.2011 17:09	Documentum
0900000280	AdvVenit2011	Adeverinta venit - 2011	Economie	User1	31.10.2011 15:43	Documentum
0900000280	AdvVenit2010	Adeverinta venit - 2010	Economie	User1	31.10.2011 15:44	Documentum

Fig. 9. Federated search form and the returned results.

```
1. SELECT cmis:objectId, cmis:name, cmis:contentStreamLength, cmis:createdBy,
cmis:creationDate, cmis:title, cmis:subject FROM cmis:document WHERE
cmis:createdBy LIKE 'User%'
2. SELECT cmis:objectId, cmis:name, cmis:contentStreamLength, cmis:createdBy,
cmis:creationDate, cmis:title, cmis:subject FROM cmis:document WHERE
cmis:createdBy LIKE 'User%' AND cmis:creationDate >= TIMESTAMP '2011-06-
30T00:00:00.000+03:00' AND cmis:creationDate <= TIMESTAMP '2011-11-
01T23:59:59.999+02:00'
SELECT cmis:objectId, cmis:name, cmis:contentStreamLength, cmis:createdBy,
cmis:creationDate, cmis:title, cmis:subject FROM cmis:document WHERE cmis:title
LIKE 'Bi%' AND cmis:createdBy LIKE 'User%' AND cmis:creationDate >= TIMESTAMP
'2011-06-30T00:00:00.000+03:00' AND cmis:creationDate <= TIMESTAMP '2011-11-
01T23:59:59.999+02:00
```

Fig. 10. Experimental CMIS queries.

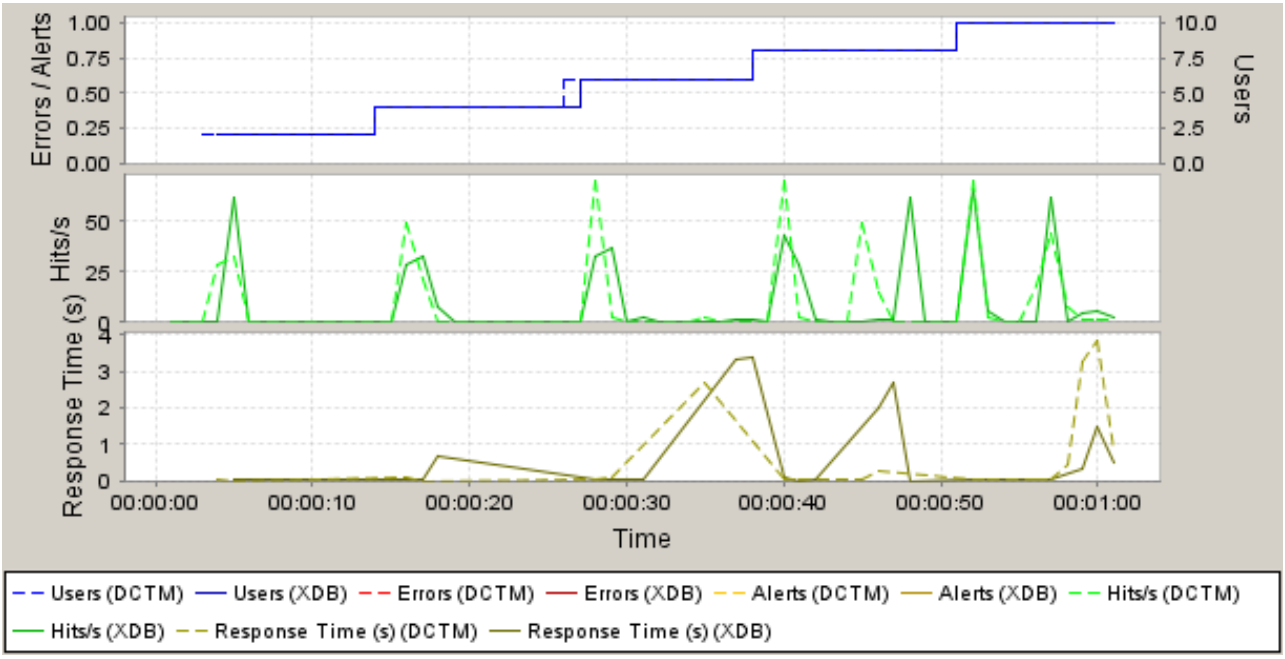


Fig. 11. Statistics summary.

This experimental scenario was performed separately for both the Documentum repository and xDB XML database to highlight the differences between these two CMIS interfaces. The statistics summary computed in this experiment are shown in figure 11. At the top of the graph is represented the virtual users load during the test. The central part of the graph represents the number of requests per second and the bottom of the graph describes the response times obtained in these experiments. The response time against load is shown in Fig. 12.

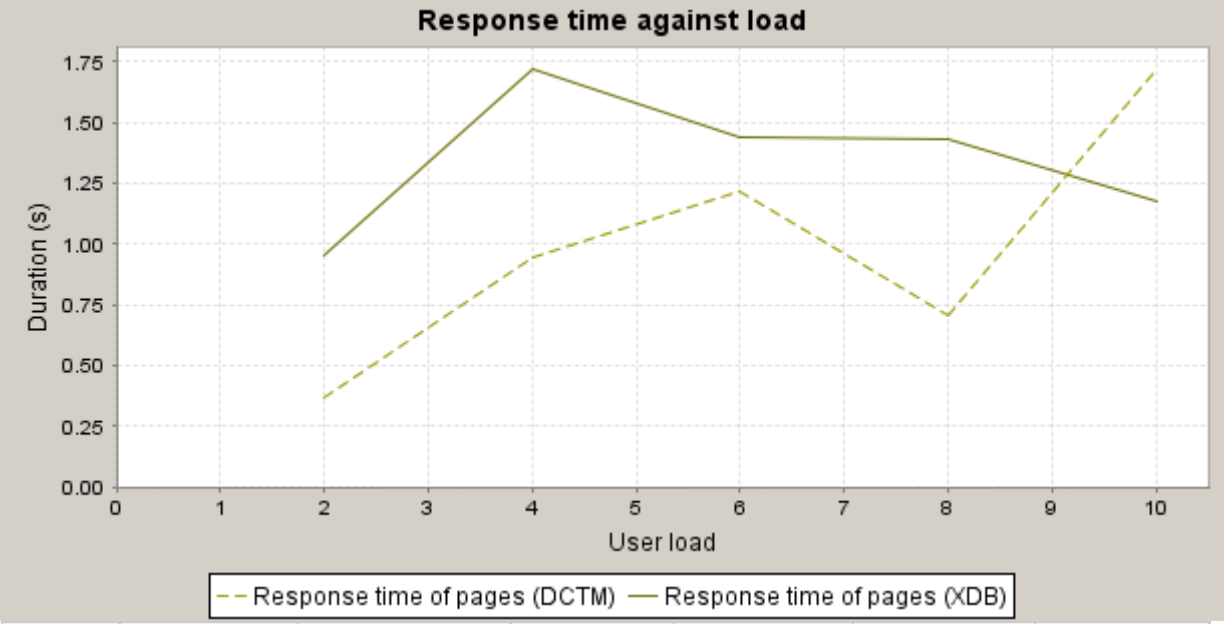
From these experimental studies we can see that CMIS interface for Documentum provides better response times, but the difference is not significant. ECM systems like EMC Documentum receive the support from the underlying relational databases, taking advantage of the advanced indexing mechanism used by the relational systems. The experiments done by using the native access to the two data sources provides similar response times, which indicates that the implementation of CMIS for xDB is optimal in relation to the implementation of CMIS provided by EMC for Documentum repositories.

In these experiments the EMC Documentum 6.5 platform was installed on SQL Server 2008 R2 Standard. On the other hand, the xDB database, being a native XML database, has its own storage system. The user interface was implemented as Web application deployed in Apache Tomcat 6.0.29 and uses J2EE technologies such as JSF and IceFaces.

6. CONCLUSIONS

In this paper is presented, firstly, Enterprise Content Management (ECM) and how it can improve the organization and storage of documents in the organizational processes.

Following the development of ECM systems, in Section 4 was presented a new standard, CMIS (Content Management Interoperability Services) offered by AIIM Standards Organization [AIIM, 2009]. This new standard aims to improve interoperability between ECM systems. In section 5 it is proposed a *new model to extend the



	Graph Min	Average	Graph Max	Graph Median	Graph Avg-90%	Graph Std. Dev.
DCTM	0.368	0.991	1.716	0.946	0.809	0.513
XDB	0.953	1.344	1.723	1.428	1.249	0.292
%	+159%	+35.7%	+0.4%	+51%	+54.4%	-43%

Fig. 12. Response time against load.

standard in order to take advantage of the CMIS interoperability features in the information exchange processes between ECM systems and XML databases. Also, this section describes the implementation of such an extension that enables interoperability between an EMC Documentum repository and a XML-native database EMC xDB.

FUTURE WORK

The future work will include the implementation and testing of a CMIS interface that access data from a native XML database that supports all the capabilities of an ECM system. Also, this implementation can make use of a wider range of native XML databases: eXist, BaseX, MonetDB etc.

Also, the development of graphics software tools for defining the types of correspondence between CMIS and native XML database, with optional generation of XSL transformations, may be a goal of future researches.

Implementing a semantic caching system for a CMIS client is also another future research direction [Chen et al., 2002] [Stancu, 2011a].

REFERENCES

AIIM – Association for Information and Image Management. (2009). CMIS, Content Management Interoperable Services. Available at http://www.aiim.org/Resources/Standards/Articles/CMIS

AIIM – Association for Information and Image Management. (2001). What is ECM Enterprise Content Management. Available at http://www.aiim.org/What-is-ECM-Enterprise-Content-Management

Apparao, V. et al. (1998). Document Object Model (DOM) level 1 specification, W3C Recommendation. http://www.w3.org/TR/REC-DOM-Level-1/.

Boag, S. et al. (2003). XQuery 1.0: An XML query language. W3C Working Draft. http://www.w3.org/TR/xquery/.

Borden, J., Martin, L., Staken, K. (2003). XML:DB Initiative for XML Databases. Available at http://xmldb-org.sourceforge.net/index.html

Chen, L., Rundensteiner, E., A. (2002). ACE-XQ: A CachE-aware XQuery Answering System . *Proceedings of the 5th International Workshop on the Web and Databases (WebDB)*, Madison, WI, p. 31–36.

Clark, J. (1999). XSL transformations (XSLT) specification. available from the W3C Consortium, http://www.w3.org/TR/WD-xslt.

JCP – Java Community Process. (2009). JSR 225: XQuery API for Java (XQJ). Available at http://jcp.org/en/jsr/summary?id=225

Megginson, D. et al. (2000). Simple API for XML (SAX). Available at http://www.saxproject.org/

Stancu, M. (2011). Semantic Cache - Optimizing the Semistructured Data Queries. *Annals of the University of Craiova - Mathematics and Computer Science* Series, ISSN 1223-6934 Vol. 38, No. 3.

Stancu, M. (2011). CMIS eXtent: a bridge from ECM systems to XML Databases. *Journal of Knowledge, Communications and Computing Technologies*, ISSN 2067-0958, Vol. 3, No. 2