MILOS: a General Purpose Multimedia Content Management System

Giuseppe Amato, Paolo Bolettieri, Franca Debole,
Fabrizio Falchi, Francesco Furfari, Claudio Gennaro
ISTI-CNR
Pisa, Italy
{giuseppe.amato, paolo.bolettieri, franca.debole
fabrizio.falchi, francesco.furfari, claudio.gennaro}@isti.cnr.it

July 26, 2004

Abstract

This report describes the MILOS Multimedia Content Management System: a general purpose software component tailored to support design and effective implementation of any digital library application. MILOS supports the storage and content based retrieval of any multimedia documents whose descriptions are provided by using arbitrary metadata models represented in XML. MILOS is **flexible** in the management of documents containing different types of data and content descriptions; it is **efficient** and **scalable** in the storage and content based retrieval of these documents. The report illustrates the MILOS system within the Italian Project VICE. However, the solution adopted is able to support the management of different metadata descriptions of multimedia documents in the same repository. For this reason, we illustrate the experiments performed by using the MILOS system to archive documents belonging to four different and heterogenous collections which contain news agencies, scientific papers, and audio/video documentaries.

1 Introduction

In this Report, we present the architecture and the functions of MILOS (MultImedia Learning Object Server) a Repository System intended to efficiently support the distributed storage and retrieval of Multimedia Learning Objects, to be developed by the ISTI-CNR Unit in the context of project VICE.

VICE is a three-year project, starting in 2003, financed by the Italian Ministry of Education, University and Research (MIUR). The objective of the project is to make it possible high quality and effective distance learning, in a cost-effective manner, supporting, in an integrated fashion, teaching/learning activities organized by an authority (it could be an academic institution, or an

enterprise or education provider, etc.) and self-learning (based on self-identified needs and goals), in the context of working activities. The role of our research group is to apply digital library techniques to support the management, retrieval and reuse of Learning Objects, i.e. collection of content/activities, that can be composed according to different needs and different goals.

MILOS has been designed with the aim of constituting an essential component of future infrastructures (distributed, peer-to-peer) for commercially viable e-Learning environments. MILOS is based on powerful multimedia database, able to guarantee advanced features for the persistence, search, and retrieval of Learning Objects written as xml documents and described using W3C xml schema [9]. Since the managed document are in xml format, it will possible to integrate the functionality of SCORM [5] and MPEG-7 [6] (and possible application specific standards), as they are fully supported by the xml schemas. In particular, in the context of project VICE, SCORM will be used to describe Learning Object documents, and MPEG7 will be exploited for enriching multimedia components of the LO. Multimedia components of SCORM objects (raw-data) can be, images, videos, Power Point presentation, MPEG-4 interactive object, etc. (see Section 2 for SCORM and Section 3 for MPEG-7) MILOS is based on a thee-tire architecture (see Section 4), and the search functionality exported by the services of business logic of the system will be able to adapt themselves on the basis of the xml schema of the managed documents. The full implementation of this architecture is expected to be completed within the project. A first prototype, constituting a partial implementation of this architecture, has been completed in the first year of the project and it is described in Section 5. Finally we present several significant digital libraries applications that were implemented by using MILOS, and we show the advantages of the proposed approach in building these specific Digital Library applications, resulting in the simplicity of the implementation and in significant system performance (Section 6).

2 SCORM

The SCORM (Sharable Content Object Reference Model) [5] metadata Information Model is a reference to the IMS Learning Resource metadata Information Model, which itself is based on the IEEE 1484.12.1 LOM (Learning Object metadata) standard. SCORM is developed by the Advanced Distributed Learning (ADL) established by the US Department of Defense (DoD) in 1997 to develop a DoD wide strategy for using learning and information technologies to modernize education and training and to promote cooperation between government, academia and business to develop e learning standardization.

The SCORM defines a Web-based learning "Content Aggregation Model" and "Run-time Environment" for learning objects. The purpose of the SCORM Content Aggregation Model is to provide a common means for composing learning content form discoverable, reusable, sharable and interoperable sources. The SCORM Content Aggregation Model further defines how learning content can

be identified and described, aggregated into a course or portion of a course and moved between systems that may include Learning Management Systems (LMS) and repositories. The purpose of the SCORM Run-time Environment is to provide a means for interoperability between Sharable Content Object-based learning content and LMSs. A requirement of the SCORM is that learning content be interoperable across multiple LMSs regardless of the tools used to create the content. For this to be possible, there must be a common way to start content, a common way for content to communicate with an LMS and predefined data elements that are exchanged between an LMS and content during its execution.

In Appendix A we report: the XML Schema Definition control document (i.e. the IMS Global Learning Consortium, Inc. Learning Resource metadata Specification Version 1.2.1 imsmd_rootv1p2p1.xsd) used to define the grammars and definition on which the SCORM metadata Application Profiles are based; the XML Schema Definition control document used to define specific grammars and definitions of the SCORM Content Packaging Application Profiles (adlcp_rootv1p2.xsd); the XML Schema Definition control document (i.e. IMS Global Learning Consortium, Inc. Content Packaging Specification Version 1.1.2 imscp_rootv1p1p2.zip) used to define the grammars and definition on which the SCORM Content Packaging Application Profiles are based; and the ims_xml.xsd XML schema referred by the IMS XML schemas.

2.1 Content Aggregation Model

One activity of the process of creating and delivering learning experiences involves the creation, discovery and gathering together, or aggregation, of simple assets into more complex learning resources and then organizing the resources into a predefined sequence of delivery. The Content Aggregation Model supports this component and is made up of: Content Model, metadata and Content Packaging.

Content Model is a nomenclature defining the content components of a learning experience. metadata is a mechanism for describing specific instances of the components of the content model. Content Packaging defines how to represent the intended behavior of a learning experience (Content Structure) and how to package learning resources for movement between different environments (Content Packaging). The SCORM Content Model Components is composed of three components: Asset, SCO and Content Aggregation.

A SCORM Asset is a collection of one or more resources (Whole web page, HTML Fragment, JavaScript Functions, Flash Object, JPEG Images, XML Doc) that are appropriate for sharing among SCOs; when packaged, an Asset should contain the appropriate metadata making it searchable in a SCORM repository. Sharable Resource metadata is data contain in the resources manifest file that describes the resource. An Asset can be described with Asset metadata. An Asset metadata is a definition of metadata that can be applied to “raw media” Assets that provides descriptive information about the Asset independent of any usage or potential usage within courseware content. This

metadata is used to facilitate reuse and discoverability principally during content creation, of such Assets within, for example, a content repository.

Assets are not only resources that can be shared among SCOs, they are also resources that have the capability of being discovered in SCORM repositories. SCORM Assets are clearly identified with a unique syntax in the course manifest file. In fact, SCORM allows for packaging Assets in a single manifest file that may be independent of any particular learning object or sharable content object. This allows SCORM repositories to consist of both learning objects and assets. SCOs, Sharable Content Objects, are built using resources, such as web pages, graphics files, etc. In some cases, these resources might need to be shared with other SCOs. Resources that can be shared with other SCOs are called Course Assets, or Sharable Resources. The SCORM documentation defines a Sharable Resource in a broader context with these words: "A Sharable Resource is a specialized instance of a Resource. By definition a Sharable Resource is a resource that is capable of being searched for and discovered within on-line repositories, thereby enhancing opportunities for reuse". This definition essentially consists of three defining characteristics: a SCO is the lowest level component that might be used in another course; a SCO should provide useful learning content by itself; a SCO must be designed to be launched and tracked by a SCORM-compliant LMS. In other words a SCO is a set of related resources that comprise a complete unit of learning content compatible with SCORM run-time requirements. With this definition, a SCO can be extracted from a learning object and used by another learning object. This is essential to achieving the SCORM reusability goal.

A SCO represents a collection of one or more Assets that include a specific launchable asset that utilizes the SCORM Run-Time Environment to communicate with Learning Management System (LMSs). A SCO represents the lowest level or granularity of learning resources that can be tracked by an LMS using the SCORM Run-Time Environment. A SCO is required to adhere to the SCORM Run-Time Environment. This implies that it must have a means to locate an LMS's API Adapter and must contain the minimum API calls (LMSInitialize("") and LMSFinish("")). There is no obligation to implement any of the other API calls as those are optional and depend upon the nature of the content. Participation in the SCORM Run-Tie Environment also means that a SCO may only be launched by an LMS. A SCO may not itself launch other SCOs.

A SCO can be described with SCO metadata. A SCO metadata is a definition of metadata that can be applied to SCOs that provides descriptive information about the content represented in the SCO. This metadata is used to facilitate reuse and discoverability of such content within, for example, a content repository.

A Content Aggregation is a map (content structure) that can be used to aggregate learning resources into a cohesive unit of instruction (e.g. course, chapter, module, etc.), apply structure and associate learning taxonomies. In SCORM "A Learning Object, or SCORM Content Aggregation, is a collection of Sharable Content Objects (SCOs) described by a SCORM manifest file".

A Content Aggregation can reference Content Aggregation metadata. A Content Aggregation metadata is a definition for metadata that describes the content aggregation. The purpose of applying Content Aggregation metadata is to make the content aggregation accessible (enabling discoverability) within, for example, a content repository and to provide descriptive information about the aggregated content represented by the content package as a whole, autonomous unit.

2.2 metadata

The SCORM metadata application profiles directly reference the IEEE LTSC Learning Object metadata (LOM) standard and the IMS Learning Resource metadata XML Binding Specification. The IEEE specification provides roughly 64 metadata elements - more than would be practical for everyday use. SCORM defines which data elements are mandatory in metadata used for tagging assets, sharable content objects, and aggregation (collections) of learning resources. SCORM provides additional specific guidance to IEEE and IMS specifications, for using metadata in SCORM conforming environments.

The IMS Learning Resource metadata Specification allows for communities to place their own namespaced metadata elements throughout an IMS metadata record. All elements that are labeled as container elements in the Information Model allow for the capability to add extensions. A SCORM metadata instance shall not be deemed invalid if it contains proprietary extensions. A conforming system that reads such a metadata instance may ignore XML elements that are not part of the schema defined in SCORM.

AICC/CMI Guidelines for Interoperability, version 3.4, are the basis for the SCORM 1.2 run-time environment and metadata components.

2.3 Content Packaging

The mechanism for binding content aggregations to Content Aggregation metadata is the Content Package. The purpose of Content Packaging is to provide a standardized way to exchange digital learning resources between different systems or tools. Content Packaging also define the structure (or organization) and the intended behavior of a collection of learning resources. Content Packaging defines, among other things: a Manifest file describing the package itself and which contains metadata about the package, an optional Organization section that defines content structure and behavior, a list of references to the resources in the package; how to create an XML-based Manifest; directions for packaging the Manifest and all related physical files into a zip file or on a CD ROM, etc.

SCORM packaging adheres strictly to the IMS Content Packaging Specification but provides additional explicit implementation guidance for packaging digital learning resources (Assets, SCO and Content Aggregations).

3 MPEG-7

MPEG-7, formally named Multimedia Content Description Interface, is an ISO/IEC standard developed by MPEG (Moving Picture Experts Group) [6]. The MPEG-7 specify a standard way of describing various type of multimedia information: elementary pieces, complete works and repositories, irrespective of their representation format and storage medium. The objective is to facilitate the quick and efficient identification of interesting and relevant information and the efficient management of that information. These descriptions are both textual (annotations, names of actors etc.) and non textual (statistical features, camera parameters etc.).

Even without MPEG-7, there are many ways to describe multimedia content in use today in various digital asset management systems. Such systems, however, generally do not allow a search across different repositories and do not facilitate content exchange between different databases using different description systems.

MPEG-7 offers a comprehensive set of audiovisual Description Tools (the metadata elements and their structure and relationships, that are defined by the standard in the form of Descriptors an Description Schemes) to create descriptions (i.e., a set of instantiated Description Schemes and their corresponding Descriptors at the users will), which will form the basis for applications enabling the needed effective and efficient access (search, filtering and browsing) to multimedia content. MPEG-7 supports operational requirements that apply, in principle, to both real-time and non real-time as well as push and pull applications.

The main elements of the MPEG-7 standard are: Description Tools (Descriptors and Description Schemes); Description Definition Language (DDL); System tools. Descriptors (D) define the syntax and the semantics of each feature (metadata element). Description Schemes (DS) specify the structure and semantics of the relationships between their components, which may be both Descriptors and Description Schemes. The DDL defines the syntax of the MPEG-7 Description Tools and allows the creation of new Description Schemes and, possibly, Descriptors and allows the extension and modification of existing Description Schemes. System tools support binary coded representation for efficient storage and transmission, transmission mechanisms (both for textual and binary formats), multiplexing of descriptions, synchronization of descriptions with content, management and protection of intellectual property in MPEG-7 descriptions, etc.

The DDL is a schema language to represent the results of modeling audiovisual data, (i.e. descriptors and description schemes) as a set of syntactic, structural and value constraints to which valid MPEG-7 descriptors, description schemes, and descriptions must conform. The purpose of a XML schema is to define a class of XML documents. The purpose of an MPEG-7 schema is to define a class of MPEG-7 documents. MPEG-7 instances are XML documents that conform to a particular MPEG-7 schema (expressed in the DDL) and the described audiovisual content. DDL consists of the following logical

components: XML Schema structural components, XML Schema data types and MPEG-7 specific extensions.

The MPEG-7 specific extensions of the XML Schema Language are: array and matrix data types; built in derived data types. Array and matrix data types can restrict the size of multidimensional matrices to a predefined facet value in a schema or to an attribute at time of instantiation. Built in derived Data Types are: basicTimePoint and basicDuration. An XML Schema parser validating an MPEG-7 XML document using MPEG-7 schemas (expressed in the DDL), just ignore MPEG-7 specific extensions.

The MPEG-7 descriptors are designed for describing the following types of information: low level audiovisual features such as colour, texture, motion, audio energy and so forth; high level features of semantic objects, events and abstract concepts; content management processes; information about the storage media and so forth. It is expected that most descriptors corresponding to low level features will be extracted automatically, whereas human intervention will be required for producing the high level descriptors.

An example of an MPEG 7 document describing low level information of an image is:

```
...
<VisualDescriptor xsi:type="EdgeHistogramType">
   <BinCounts>
      1 0 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 3 2 3
      2 5 3 3 0 2 5 1 0 0 0 1 0 0 0 0 0 2 0 1 0 2 3 0 0 2 1 0
      0 0 0 2 0 0 0 0 0 0 0 0 0 0 1 1 0 1 2 1 1 0 0 1
   </BinCounts>
</VisualDescriptor> <VisualDescriptor xsi:type="ColorLayoutType">
   <YDCCoeff> 3 </YDCCoeff>
   <CbDCCoeff> 30 </CbDCCoeff>
   <CrDCCoeff> 31 </CrDCCoeff>
   <YACCoeff5> 16 16 15 16 15 </YACCoeff5>
   <CbACCoeff2> 16 16 </CbACCoeff2>
   <CrACCoeff2> 16 16 </CrACCoeff2>
</VisualDescriptor> <VisualDescriptor
xsi:type="DominantColorType">
   <SpatialCoherency> 29 </SpatialCoherency>
   <Value>
      <Percentage> 29 </Percentage>
      <Index> 10 10 8 </Index>
      <ColorVariance> 0 0 0 </ColorVariance>
   </Value>
..
```

MPEG-7 defines the mean of the descriptors (i.e. the description) but not the extraction process or the way to use them. For example the distance between two descriptors of the same type is not defined.

Description Schemes expand on the MPEG-7 descriptors by combining individual descriptors and other description schemes within more complex structure and by defining the relationship between the constituent descriptors and description scheme.

The XML schema used to define the MPEG-7 Description Definition Language (ddl-2001.xsd), the standard MPEG-7 schema defined using the DDL (Mpeg7-2001.xsd) and links for the visual-2001.xsd, audio-2001.xsd and msd-2001.xsd documents referred by the MPEG-7 schema, can be found in Appendix B.

4 MILOS LO Repository System Architecture

In this section, we describe the target architecture of MILOS, the Repository System for Learning Objects which constitutes the main contribution of the ISTI-CNR Unit within the VICE project. The full implementation of this architecture is expected to be completed within the project. A first prototype, constituting a partial implementation of this architecture, has been completed in the first year of the project and it is described in Section 4. This first prototype is available for experimentations in the context of project VICE. MILOS is designed to support the storage and retrieval of multimedia Learing Objects (LO). The system is based on a three–tire architecture (Figure 1) composed of five main logical components: metadata Modeler, Interface Logic, Adaptable Repository Service Logic, Storage Layer (composed of the LO Database, and the metadata Database). Note that Figure 1 also illustrates the relationship between the LO Tools (metadata Editors) and the LO Repository. The metadata Modeler is a tool for managing the design of model of metadata for describing LOs. The Interface Logic includes components that allow users to interact with the system on the web, via normal browsers, in order to retrieval and edit the LOs. The Repository Service Logic manages accesses to data stored in the LO repository and metadata database, on behalf of the other two components. In the following, we will give a more detailed description of each of these components. All the components will communicate by means of protocols for distributed systems integration (e.g. SOAP).

4.1 Metadata Modeler

It is a Tool (or a set of Tools) which allows the user to define new model of metadata or enriching the existing model of metadata. For instance, suppose we would like to create a new schema of Learning Object with new functionalities and Ontologies, the output of this design phase will produce one or more xml–schemas, and will be used as input parameter for the Adaptable Repository Service Logic. This component is able to adapt its interface on the basis of the schema imported (these tools will be realized by other Units within project VICE)

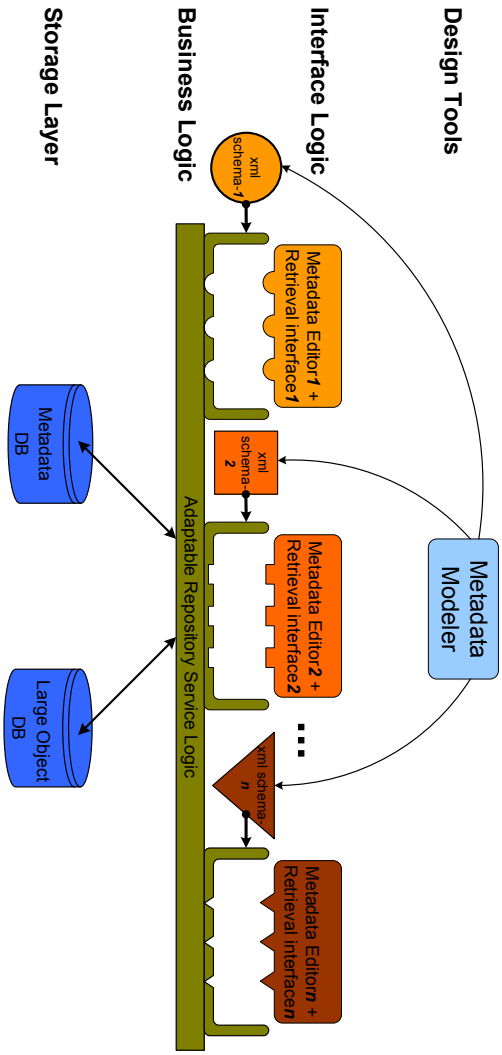

Figure 1: Architecture of Repository Systems for Learning Objects

4.2 Interface Logic

The interface logic includes modules, related to the activities that can be carried out by users of the system. Users will interact with the system through normal web browsers. The metadata Editor Tools, which are part of the authoring tools, allow users to manually edit and review metadata associated with the LOs. The user can either edit automatically generated metadata, as for instance scene boundaries, or can add additional metadata manually. The Retrieval tools are used to search the system LOs. Various possibilities are offered by this interface: users can retrieve documents by performing full text retrieval on the transcript or descriptions associated with LOs, selecting specific fields of the metadata structure, similarity search on multimedia content, or combination of them. As an example of metadata editor, we show the one developed during the ECHO project, a European project intended to manage historical Audio/Video material, outside the scope of project VICE (see Figure 2). An important feature of the metadata editor is that it not hard wired with a particular metadata attributes set, indeed the echo xml metadata schema is used by the editor as configuration file for the metadata model. The advantage of this choice is that it is possible to add/remove fields in the schema of the metadata of the audiovisual document. This is achieved by giving the editor the ability of recognizing a subset of the types available for the XSD schemas, such as: strings, booleans, dates, integers, etc.

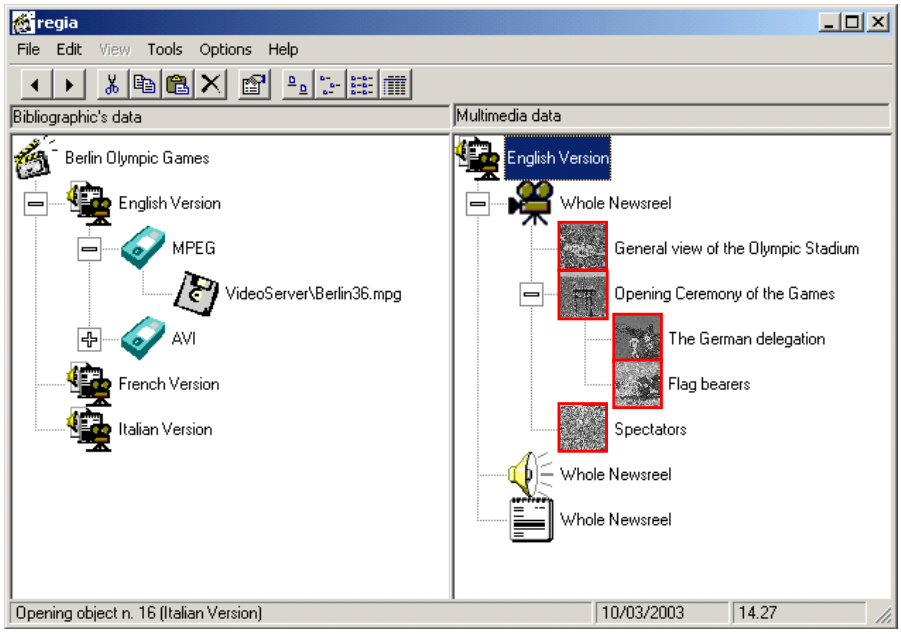

Figure 2: The ECHO metadata editor screen shot

4.3 Adaptable Repository Service Logic (ARLS)

This component manages the accesses to the underlying databases: the LO database, that physically stores Learning Objects managed by the system, and the metadata database, where all metadata associated with the LOs are stored. It manages query processing by integrating and aligning information stored in the two databases. It performs reconciliation of retrieved data by managing ranking. When a new xml schema is introduced, the system will be able to automatically generate an new middleware interface capable to communicate with the editor and retrieval tool of the interface logic. The designer will interact with the ARSL in order to specify which metadata fields are important for the search, and how to map the anthologies, for the retrieval of the LOs.

4.4 Metadata Database

This part of the system manages the metadata associated with the LOs represented here in XML format, using appropriate XML Schema definitions. Therefore, the power of XML structuring and inter-linking will be used in searching LO through the various metadata associated, exploiting also the interconnections between different metadata, at different levels of detail (e.g. information about the content of a scene of an audio/visual presentation, related to the range of technical requirement for display, and minimal background prerequisite for learners).

For this purpose we have designed and implemented an enhanced native XML databases for Digital library application. Our native XML database, in addition to support XML query language standards such as XQuery and XPath, offers advanced search and indexing functionality on arbitrary XML documents. Consider that new generation metadata standard, such as MPEG-7, include in their description also feature automatically extracted from visual documents, such as color histograms, textures, shapes, etc. Thus in addition to index structure that support high performance search and retrieval on heavily structured XML documents [7, 13], our XML database also provides full text search [12], automatic classification [11], and feature similarity search [8] functionality. Specifically, the XML database allows the system administrator to associate specific element names of the XML with special indexes. So, for instance, the tag name `<abstract>` can be associated with a full text index and to an automatic topic classifier that automatically index it with topics chosen from a controlled vocabulary. On the other hand, the MPEG-7 `<VisualDescriptor>` tag might be associated with a similarity search index structure and with a an automatic visual content classifier. To deal easily and transparently with these advanced search and indexing functionalities, we have extended the syntax of the basic XQuery language with new operators that deal with approximate match and ranking.

4.5 LO Database

This repository will be able to deal with various media formats. Different storage strategies will be used in correspondence of different media types and different media usages. XML structures will be used to describe aggregations, at different levels, of elements included in LOs. It will be possible to access an entire LO or to access its specific parts in a selective way. Basic searching functionalities will be provided as well.

5 State of the design and implementation of MILOS

In the context of project VICE, the ISTI-CNR Unit will supply the base framework for the persistence, access and retrieval of the LO repositories.

The framework will be realized on a three–tire architecture based upon a new concept XML engine (Figure 1). The metadata DB is the support for the XML objects persistence and retrieval, which will be used to store LOs. The Large Object DB will be address to store Raw Data associated to the LOs (videos, images, sounds, presentations).

As stated in BPEL4WS Specification, "the goal of the Web Services effort is to achieve universal interoperability between applications by using Web standards. Web Services use a loosely coupled integration model to allow flexible integration of heterogeneous systems in a variety of domains including business-to-consumer, business-to-business and enterprise application integration". With the interoperability issue in mind the implementation of the three–tier architecture has been realized using as middleware the Web Services paradigm. At the present time all the aspects concerning the discovery process and security are not addressed.

Java^{TM} has been selected as reference platform in this development phase mainly for its characteristic of portability . The software infrastructure for the middleware is based on open sources projects developed on the Apache Software Foundation, we use Tomcat as servlet container (web server) and Axis for the Web Services management and deployment. The XML Search Engine, written in java, depends on Berkeley DB library a widely deployed data management engine, available on the most popular platforms in use today. The BerkeleyDB library provides the implementation, based on B+-tree, for the inverted index . In the same way the fulltext search features are based on the support provided by external full-text engines. The integration is realized following a class factory pattern and an implementation based on Microsoft SQLServer is provided by default.

The conceptual architecture delineated in Figure 3 has been mapped in the physical layers reported in Figure 4. The Adaptable Repository Service Logic is implemented by the MultiMedia Digital Library Server (MMDLS) meanwhile the Learning Objects Database is realized by the MultiMedia Server (MMS), finally the metadata DB is represented by the XML Search Engine

(XMLSE) and a DBMS (MS SQLServer by Default). The system can be distributed on six different hosts configuring a file of properties (milos.properties) to indicate the hostname where are deployed the different system components. All the components share the same web application domain (milos) and the WSDL is collected concatenating the service name to the web domain (es: hostname/milos/MMS?MMS.jws).

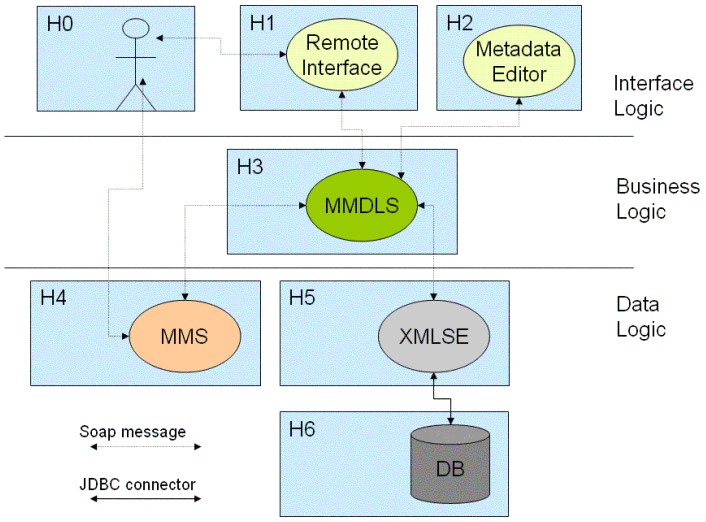

Figure 3: The Milos system architecture

5.1 XML and Ontology mapping

The business logic will be able to provide, to higher layers, an interface for storing XML objects and advanced services to retrieval them. The search services will have to be able, each time, to automatically adapt themselves to the metadata model used by the LOs. To achieve this, the business logic, using the XML schema provided by the higher layers during the system setup, will be able to translate the calls to the search services, in appropriate XQuery [10] calls to the lower layer. At the moment the metadata schema mapping is based on a "properties file" (metadata.properties) in which it is specified, when necessary, the XQuery or the alternative XPath to use for querying a specific element. The metadata.properties file is composed of a set of entries each of one specifying how translate an abstract name of a metadata field (ontology) in the XML Path for addressing a real xml element of the repository. These abstract names are

used during queries as parameters for the search methods. A generic entry of the metadata.properties file has the following structure:

```
metadataType[_Entity][.Attribute] = <XML Path Expression>
```

1. The **metadataType** field specifies the name of the model of metadata we would like to refer to, e.g., DoublinCore, SCORM, etc. In other words, it is the identifier of a dataset of metadata objects.

2. The **Entity** field is the name of the entity of the model we would like to refer to, e.g., Book, Manifest, Lom, etc. If this field is leaved empty, it means that we refer to all the entities instance of the database.

3. The **Attribute** field allows to specify the name of the attribute to search for, e.g., Title, Author, etc. If empty it means that we refer to all the attributes, in a given entity.

For better understanding, consider the following steps of the MMDLS elaboration of the service

```
findFullText(string MetadaType, vector of string value, vector of
string field, string returnField)
```

1. for each triple $< MetadaType, value_i, field_i >$ specified in the query, the MMDLS searches in the metadata.properties the corresponding XPath strings Q_i.

2. for the pair $< MetadaType, returField >$, specified in the query, the MMDLS searches in the metadata.properties the corresponding XPath string R.

3. Finally, the XPath strings Q_i and R are combined in order to form the resulting XQuery to submit to xml repository as in the following:

   for $\$a$ **in** $MetadaType$
   where $\$aQ_1 = value_1$ **and** ... **and** $\$aQ_n = value_n$
   return $\$aR$
   As an example, suppose we have stored in the repository a set of SCORM objects, we could have the following simple metadata.properties file composed of just two entries:

```
scorm_lom.Title=lom/genaral/title/langstring
scorm_lom.Desc=lom/general/description/langstring
```

The first line of the file tells the system that the metadata attribute Title of the conceptual entity called lom of the scorm dataset, is addressed by means of the XPath string *lom/genaral/title/langstring*.

Moreover, suppose, the following LOM object is stored in the repository:

```
<?xml version="1.0"?> <lom
xmlns="http://www.imsglobal.org/xsd/imsmd_rootv1p2p1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.imsglobal.org/xsd/imsmd_rootv1p2p1
imsmd_rootv1p2p1.xsd">
    <general>
        <title>
            <langstring>Vessel Aground Lighting</langstring>
        </title>
        <description>
            <langstring>Picture showing lighting requirments for inland
            vessel less than 50 meters in length in an aground condition.</langstring>
        </description>
        <keyword>
            <langstring>Vessel</langstring>
        </keyword>
        <keyword>
            <langstring>Lighting</langstring>
        </keyword>
        <keyword>
            <langstring>Aground</langstring>
        </keyword>
    </general>
    ...
</lom>
```

The call to the method

```
findFullText(‘‘scorm_lom’’,‘‘Vessel Aground
Lighting’’,‘‘Title’’ ‘‘Desc’’)
```

will return the description content of the above lom xml file. Through an ontology scheme, in future, the business logic will be able to provide tools for transparent search DL at the higher layers.

Another important system tuning is obtained configuring the kind of index to use for specific elements. By means of a properties file (index.properties) is possible to specify for the content of an element the using of a PathIndex or a full-text index. Finally, the Multimedia Server is now based on very simple data storage mechanism in future we will use adapters towards legacy server or we will be based on a more robust delivery protocol. Figure 4 reports a summary of the implemented interfaces

6 Field Trials

In order to verify and demonstrate the flexibility and efficiency of MILOS in managing different heterogeneous digital library applications, we have taken

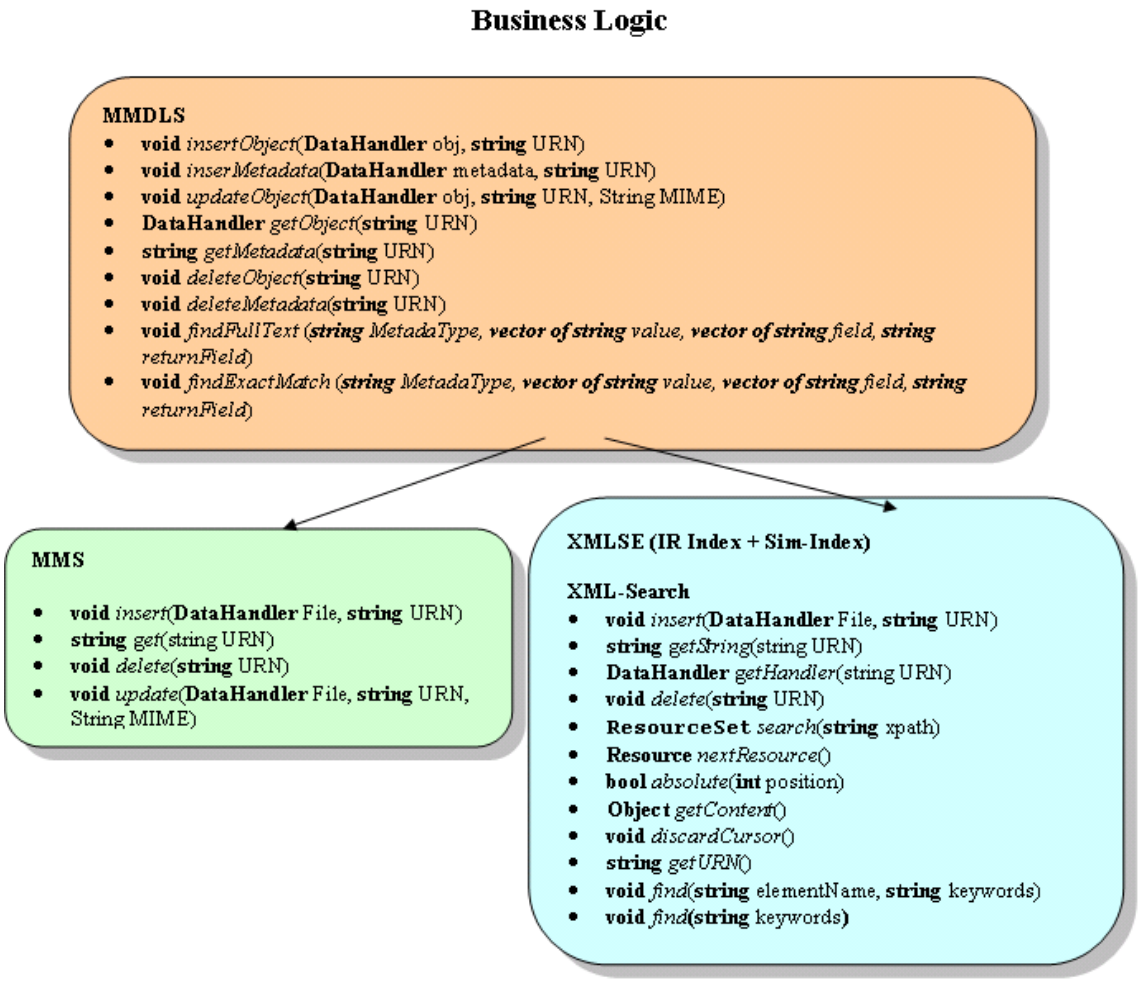

Figure 4: MILOS interfaces summary

four data sets used by four different digital libraries and we have built from scratch the corresponding digital library application on top of MILOS. The data sets that we have considered consist of documents and metadata of very different nature: the Reuters data set [3], the ACM Sigmod Record dataset [4], the DBLP data set [1], and the ECHO data set [2]. These data sets and the corresponding MILOS powered digital library applications are described in next sections.

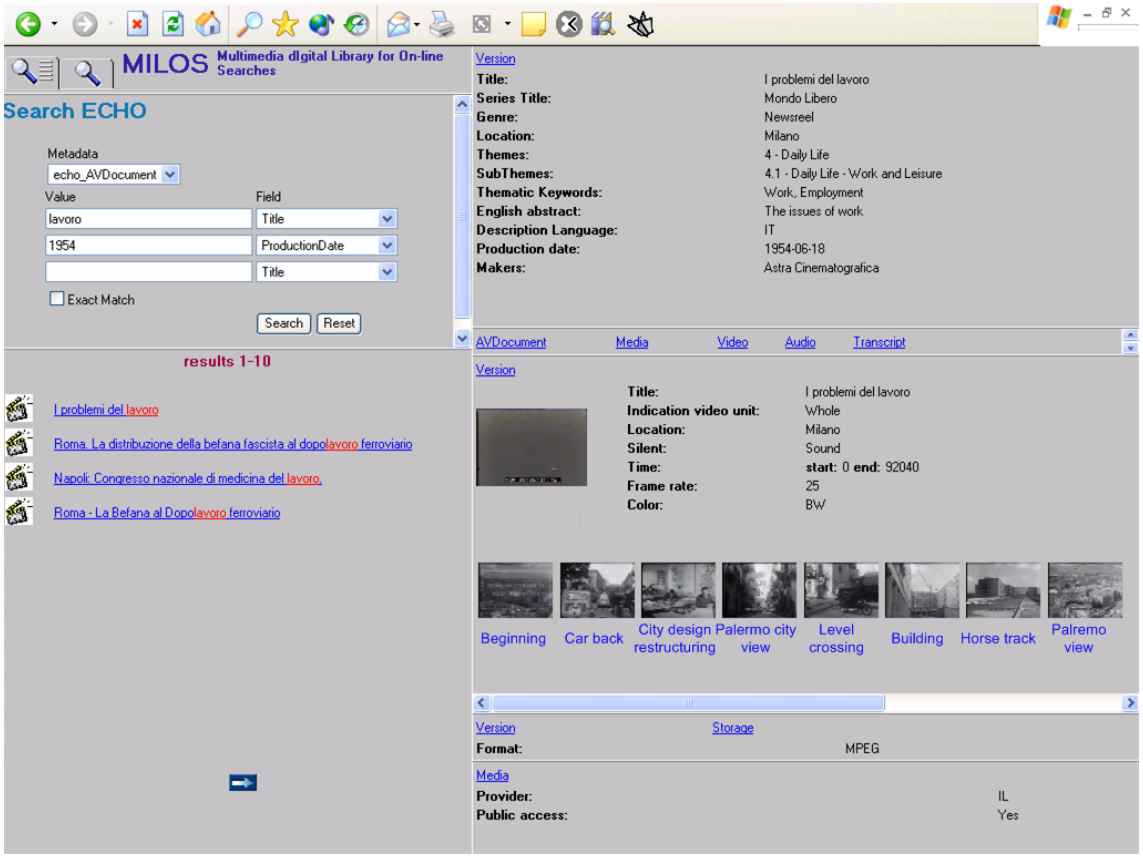

Figure 5: The ECHO retrieval interfaces implemented in MILOS

The digital library applications that we have built use the same MILOS instance and all data sets were stored together. The functionality of MILOS allows individual application to selectively access just data and metadata of their interest. Each digital library application consists of a specific search and browsing interface (built according to the data managed) and a bulk import tool. The search and browse interfaces were built as web applications using Java Server Pages (JSP). The bulk import tool was a simple java application.

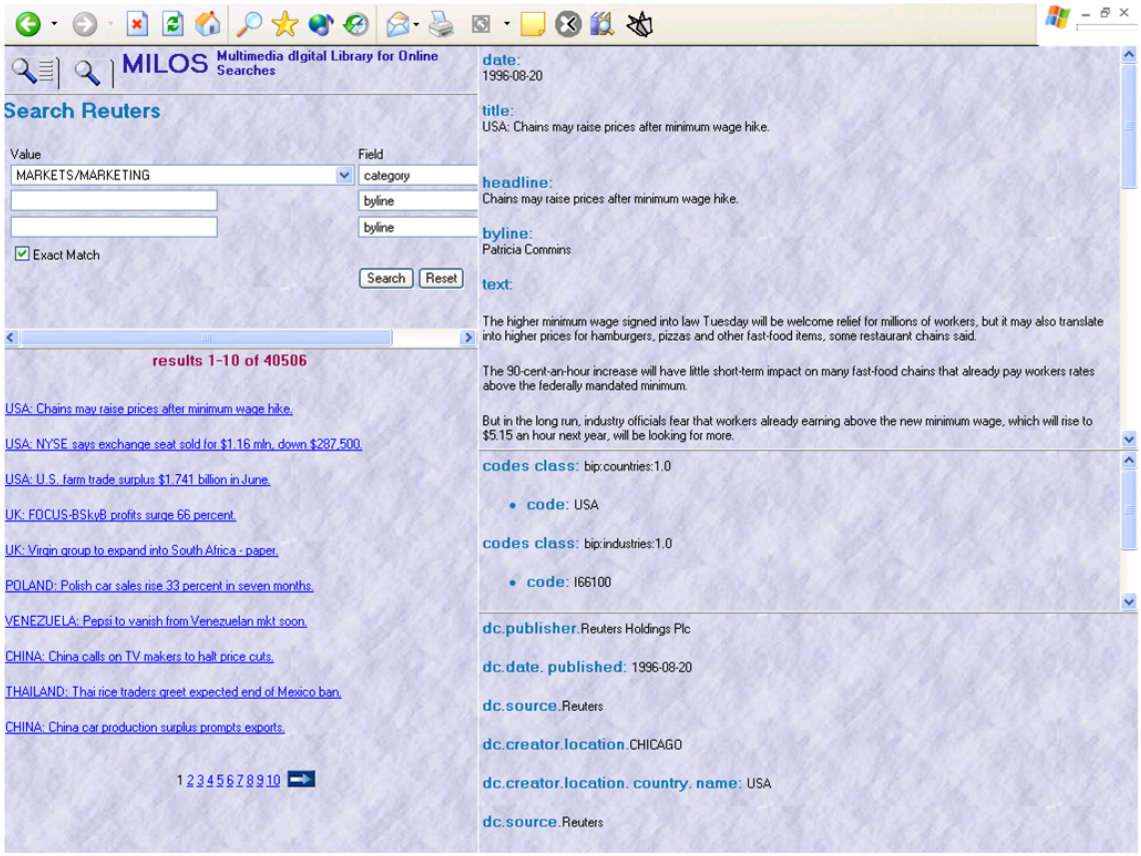

Figure 6: The Reuters retrieval interfaces implemented in MILOS

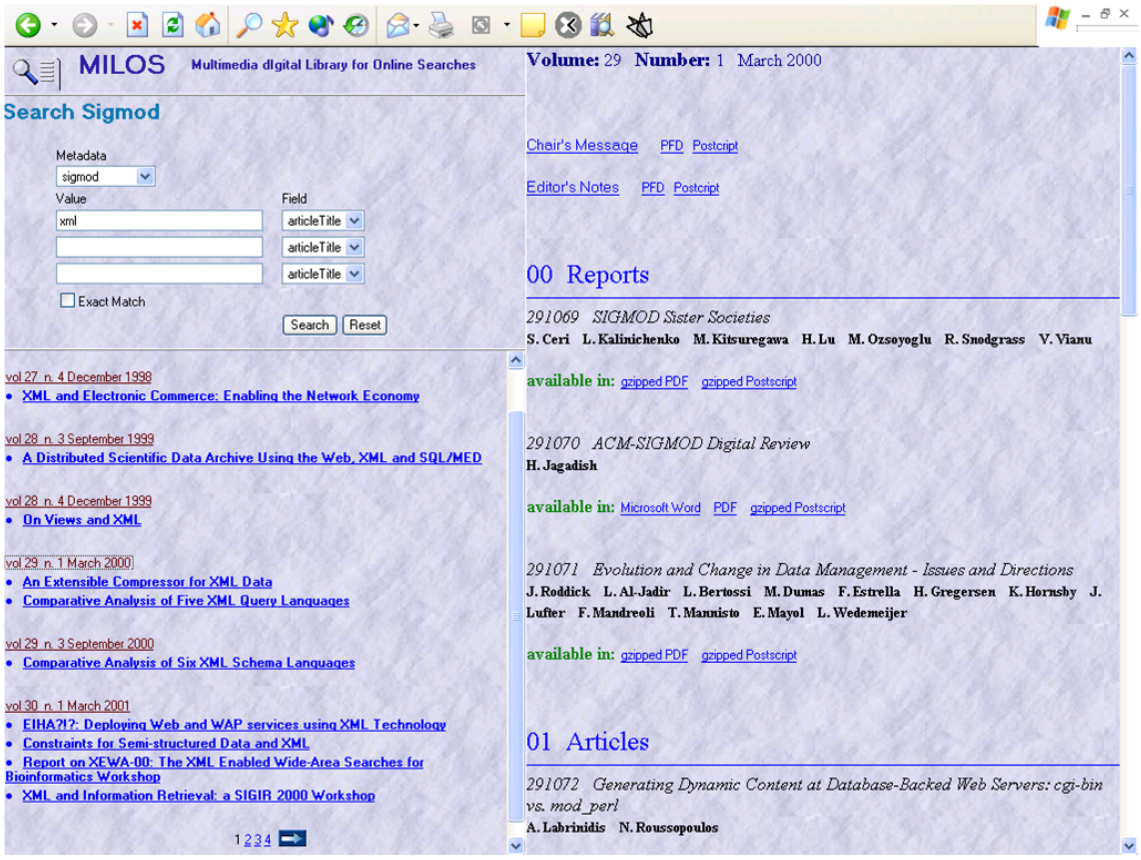

Figure 7: The Sigmod/DBLP retrieval interfaces implemented in MILOS

On average, the effort required to build each of these application from scratch was one week of work of a single skilled person. This, we believe, is really a little effort compared to the cost that would have been required to build from scratch a digital library, without general purpose tools, or the cost that would have been required to translate and adapt the data and metadata to cope with the requirements and restrictions of an existing digital library systems.

Notice that we built from scratch the browse and retrieval interface. However, we are currently working at designing and developing some tools for automatic generation of browse and retrieval interface in correspondence of data and metadata managed by MILOS. This will contribute to further reduce the cost of building digital library applications.

All the applications resulted to be very efficient. We have installed the system, the applications, and the data on a single computer equipped with a Pentium 1.8 GHz and 1 Gb of RAM, running Win 2000 server. We have used JAX-RPC as SOAP application server to run MILOS. We have asked 30 persons to use the digital library at the same time from remote workstations, and to execute a predefined search intensive job. On average the response time of the system was below 1 second. Notice also that for more intensive use of the system, the underlying Web Service technology, offers plenty of solutions to guarantee scalability exploiting techniques of replication, load balancing, resource/connection pooling etc.

The screen shots of the digital library applications can be seen in the Figures 5, 6, and 7.

6.1 Reuters data set

The Reuters data set [3] contains text news agencies and corresponding metadata. There are two types of metadata: Reuters specific metadata including titles, authors, topic categories, and extended Dublin Core metadata.

The Reuters data set contains 810,000 news agencies (2.6 Gb) where text and metadata are both encoded in XML. We have associated the full text index and the automatic topic classifier to the elements containing the body, the title, and the headline of the news. Other value indexes where associated with elements corresponding to frequently searched metadata, like locations, dates, countries.

The search interface allows user to perform integrated, text, category, and exact match search. The bulk import tool simply takes all XML file, corresponding to the news, contained in a specified directory, associates them with a unique URN, and inserts them in MILOS. The bulk import of the Reuters data set (with corresponding indexing of data and metadata) took 20 hours.

6.2 ACM Sigmod Record and DBLP data sets

Both the ACM Sigmod Record data-set [4] and the DBLP data-set [1] consist of metadata corresponding to description of scientific publication in the computer science domain. The ACM Sigmod record is relatively small. It is composed of 46 XML files (1Mb), while the DBLP data-set is composed of just one huge

(187Mb) XML file. Their structure is completely different even if they contain information describing similar objects.

For these two datasets we built just one digital library application from which both data are accessed. We exploited the mapping functionality of MILOS for having the requests of the application correctly translated for the two schemas. We have associated a full text index to the elements containing the titles of the articles, and we have associated other value indexes to other frequently searched elements, such as the authors, the dates, the years, etc.

The search and browse interface, allows users to search for articles by various combinations full text and exact/partial match of elements. In addition it allows user to browse results by navigating trough links (and implicitly submitting new queries to MILOS) related to the authors, journals, conferences, etc.

The bulk import tool simply takes the XML files corresponding to the metadata and directly insert them in the system, after having associated them with unique URNs. The bulk import of these data sets, including their indexing, required about 5 hours.

6.3 ECHO data set

The ECHO data set [2] includes historical audio/visual document and corresponding metadata. ECHO is the most significant example of the capability of MILOS to support the management of arbitrary metadata schemas. The metadata model adopted in ECHO, based on IFLA/FRBR model, is rather complex and strongly structured. It is used for representing the audio-visual contents of the archive and includes among others, the description of videos in English and in the original language, specific metadata fields such as Title, Producer, year, etc., the boundaries of the scene detected (associated with a textual descriptions), the audio segmentation (distinguishing among noise, music, speech, etc.), the Speech Transcripts, and visual features for supporting similarity search.

The collection is composed of about 8,000 documents for 50 hours of video described by 43,000 XML files (36 Mb). In addition each scene detected is associated with a JPEG encoded key frame for a total of 21GB of MPEG-1 and JPEG files. Full text indexes where associated to textual descriptive fields, similarity search index where associated with elements containing image (key frames) features, and other value indexes where associated with frequently searched elements.

The search and retrieval interface allows users to find videos by combining full text, image similarity, and exact/partial match search. Users can browse among scenes, and corresponding metadata, of retrieved videos. They can play specific scenes of a retrieved video. We simply used the file system as a repository for the videos. However, as previously discussed, moving to more complex multimedia storage services is straightforward and transparent, from the system and application perspective. The original ECHO digital library application was built using a relational database, and translating all metadata in a relational schema. Even simple searches required several seconds to be processed. With

MILOS we had a dramatic improvement of performance, being able to serve requests in less than one second even with several users accessing the system.

The bulk import tool takes the videos and the key frame images, and insert them in the system. Each of these document is associated with an unique identifier, which is also added in the ECHO metadata to correctly refer the document from the corresponding metadata. Then the updated metadata are also inserted in the system. The bulk import of data and metadata and their indexing required almost one hour.

References

[1] DBLP computer science bibliography. http://www.informatik.uni-trier.de/ ley/db/.

[2] European chronicles on-line. Available at http://pc-erato2.iei.pi.cnr.it/echo/.

[3] Reuters corpus. http://about.reuters.com/researchandstandards/corpus/.

[4] Sigmod record, xml edition. http://www.acm.org/sigs/sigmod/record/xml/.

[5] Shareable content object reference model initiative (scorm), the xml cover pages, October 2001. http://xml.coverpages.org/scorm.html.

[6] Mpeg requirements group, mpeg-7 overview, 2003. Doc. ISO/IEC JTC1/SC29/WG11N5525.

[7] G. Amato, F. Debole, F. Rabitti, and P. Zezula. YAPI: Yet another path index for XML searching. In *ECDL 2003, 7th European Conference on Research and Advanced Technology for Digital Libraries, Trondheim, Norway, August 17-22, 2003*, 2003.

[8] C. Böhm, S. Berchtold, and D. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys*, 33(3):322–373, September 2001.

[9] W. W. W. Consortium. XML path language (XPath), version 1.0, W3C. Recommendation, November 1999.

[10] W. W. W. Consortium. XQuery 1.0: An XML query language. W3C Working Draft, November 2002. http://www.w3.org/TR/xquery.

[11] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambidge University Press, 2000.

[12] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company, 1983.

[13] P. Zezula, G. Amato, F. Debole, and F. Rabitti. Tree signatures for xml querying and navigation. In *Database and XML Technologies, First International XML Database Symposium, XSym 2003*, volume 2824 of *Lecture Notes in Computer Science*, pages 149–163. Springer, 2003.

A SCORM XML-Schemas

A.1 adlcp_rootv1p2.xsd

We report the XML Schema Definition control document used to define specific grammars and definitions of the SCORM Content Packaging Application Profiles.

```
<?xml version="1.0"?> <!-- filename=adlcp_rootv1p2.xsd --> <!--
Conforms to w3c http://www.w3.org/TR/xmlschema-1/ 2000-10-24-->
<xsd:schema
targetNamespace="http://www.adlnet.org/xsd/adlcp_rootv1p2"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:imscp="http://www.imsproject.org/xsd/imscp_rootv1p1p2"
xmlns="http://www.adlnet.org/xsd/adlcp_rootv1p2"
elementFormDefault="unqualified" version="ADL Version 1.2">
    <xsd:import namespace="http://www.imsproject.org/xsd/imscp_rootv1p1p2"
    schemaLocation="imscp_rootv1p1p2.xsd"/>
    <xsd:element name="location" type="locationType"/>
    <xsd:element name="prerequisites" type="prerequisitesType"/>
    <xsd:element name="maxtimeallowed" type="maxtimeallowedType"/>
    <xsd:element name="timelimitaction" type="timelimitactionType"/>
    <xsd:element name="datafromlms" type="datafromlmsType"/>
    <xsd:element name="masteryscore" type="masteryscoreType"/>
    <xsd:element name="schema" type="newSchemaType"/>
    <xsd:simpleType name="newSchemaType">
        <xsd:restriction base="imscp:schemaType">
            <xsd:enumeration value="ADL SCORM"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:element name="schemaversion" type="newSchemaversionType"/>
    <xsd:simpleType name="newSchemaversionType">
        <xsd:restriction base="imscp:schemaversionType">
            <xsd:enumeration value="1.2"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:attribute name="scormtype">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="asset"/>
                <xsd:enumeration value="sco"/>
```

```
        </xsd:restriction>
    </xsd:simpleType>
</xsd:attribute>
<xsd:simpleType name="locationType">
    <xsd:restriction base="xsd:string">
        <xsd:maxLength value="2000"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="prerequisitesType">
    <xsd:simpleContent>
        <xsd:extension base="prerequisiteStringType">
            <xsd:attributeGroup ref="attr.prerequisitetype"/>
        </xsd:extension>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:attributeGroup name="attr.prerequisitetype">
    <xsd:attribute name="type" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="aicc_script"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:simpleType name="maxtimeallowedType">
    <xsd:restriction base="xsd:string">
        <xsd:maxLength value="13"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="timelimitactionType">
    <xsd:restriction base="stringType">
        <xsd:enumeration value="exit,no message"/>
        <xsd:enumeration value="exit,message"/>
        <xsd:enumeration value="continue,no message"/>
        <xsd:enumeration value="continue,message"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="datafromlmsType">
    <xsd:restriction base="xsd:string">
        <xsd:maxLength value="255"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="masteryscoreType">
    <xsd:restriction base="xsd:string">
        <xsd:maxLength value="200"/>
    </xsd:restriction>
```

```
    </xsd:simpleType>
    <xsd:simpleType name="stringType">
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="prerequisiteStringType">
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="200"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:schema>
```

A.2 imscp_rootv1p1p2.xsd

We report the XML Schema Definition control document (i.e. IMS Global Learning Consortium, Inc. Content Packaging Specification Version 1.1.2 imscp_rootv1p1p2.zip) used to define the grammars and definition on which the SCORM Content Packaging Application Profiles are based.

```
<?xml version="1.0"?> <!-- edited with XML Spy v3.5
(http://www.xmlspy.com) by Thomas Wason (private) --> <!--
filename=ims_cp_rootv1p1p2.xsd --> <!-- Copyright (2) 2001 IMS
Global Learning Consortium, Inc. --> <!-- edited by Thomas Wason
--> <!-- Conforms to w3c http://www.w3.org/TR/xmlschema-1/
2000-10-24--> <xsd:schema
targetNamespace="http://www.imsproject.org/xsd/imscp_rootv1p1p2"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.imsproject.org/xsd/imscp_rootv1p1p2"
elementFormDefault="unqualified" version="IMS CP 1.1.2">
    <!-- ******************** -->
    <!-- ** Change History ** -->
    <!-- ******************** -->
    <xsd:annotation>
        <xsd:documentation xml:lang="en">DRAFT XSD for
        IMS Content Packaging version 1.1 DRAFT</xsd:documentation>
        <xsd:documentation> Copyright (c) 2001 IMS GLC,
        Inc. </xsd:documentation>
        <xsd:documentation>2000-04-21, Adjustments by T.D.
        Wason from CP 1.0.</xsd:documentation>
        <xsd:documentation>2001-02-22, T.D.Wason: Modify for
        2000-10-24 XML-Schema version.  Modified to support
        extension.</xsd:documentation>
        <xsd:documentation>2001-03-12, T.D.Wason: Change filename,
        target and metadata namespaces and metadata fielname.
        Add metadata to itemType, fileType and organizationType.</xsd:documentation>
        <xsd:documentation>Do not define namespaces for xml in
        XML instances generated from this xsd.</xsd:documentation>
```

```
    <xsd:documentation>Imports IMS metadata xsd, lower case
    element names.         </xsd:documentation>
    <xsd:documentation>This XSD provides a reference
    to the IMS metadata root element as imsmd:record</xsd:documentation>
    <xsd:documentation>If the IMS metadata is to be used
    in the XML instance then the instance must define an
    IMS metadata prefix with a namespace.  The metadata
    targetNamespace should be used.  </xsd:documentation>
    <xsd:documentation>2001-03-20, Thor Anderson: Remove
    manifestref, change resourceref back to identifierref,
    change manifest back to contained by manifest. --Tom
    Wason: manifest may contain _none_ or more manifests.</xsd:documentation>
    <xsd:documentation>2001-04-13 Tom Wason: corrected
    attirbute name structure.  Was misnamed type.  </xsd:documentation>
    <xsd:documentation>2001-05-14 Schawn Thropp: Made all
    complexType extensible with the group.any</xsd:documentation>
    <xsd:documentation>Added the anyAttribute to all
    complexTypes. Changed the href attribute on the
    fileType and resourceType to xsd:string</xsd:documentation>
    <xsd:documentation>Changed the maxLength of the
    href, identifierref, parameters, structure attributes to
    match the Information model.</xsd:documentation>
    <xsd:documentation>2001-07-25 Schawn Thropp:
    Changed the namespace for the Schema of Schemas
    to the 5/2/2001 W3C XML Schema</xsd:documentation>
    <xsd:documentation>Recommendation. attributeGroup
    attr.imsmd deleted, was not used anywhere.
    Any attribute declarations that have</xsd:documentation>
    <xsd:documentation>use = "default" changed to
    use="optional" - attr.structure.req.</xsd:documentation>
    <xsd:documentation>Any attribute declarations
    that have value="somevalue" changed to default="somevalue",</xsd:documentation>
    <xsd:documentation>attr.structure.req (hierarchical).
    Removed references to IMS MD Version 1.1.</xsd:documentation>
    <xsd:documentation>Modified attribute group
    "attr.resourcetype.req" to change use from optional</xsd:documentation>
    <xsd:documentation>to required to match the information model.
    As a result the default value also needed to be removed</xsd:documentation>
    <xsd:documentation>Name change for XSD.  Changed to match version of CP Spec
</xsd:annotation>
<xsd:annotation>
    <xsd:documentation>Inclusions and Imports</xsd:documentation>
</xsd:annotation>
<xsd:import namespace="http://www.w3.org/XML/1998/namespace"
schemaLocation="ims_xml.xsd"/>
<xsd:annotation>
```

```
    <xsd:documentation>Attribute Declarations</xsd:documentation>
</xsd:annotation>
<!-- ************************** -->
<!-- ** Attribute Declarations ** -->
<!-- ************************** -->
<xsd:attributeGroup name="attr.base">
    <xsd:attribute ref="xml:base" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.default">
    <xsd:attribute name="default" type="xsd:IDREF" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.href">
    <xsd:attribute name="href" use="optional">
        <xsd:simpleType>
            <xsd:restriction base="xsd:anyURI">
                <xsd:maxLength value="2000"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.href.req">
    <xsd:attribute name="href" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:anyURI">
                <xsd:maxLength value="2000"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.identifier.req">
    <xsd:attribute name="identifier" type="xsd:ID" use="required"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.identifier">
    <xsd:attribute name="identifier" type="xsd:ID" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.isvisible">
    <xsd:attribute name="isvisible" type="xsd:boolean" use="optional"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.parameters">
    <xsd:attribute name="parameters" use="optional">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:maxLength value="1000"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
```

```
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.identifierref">
    <xsd:attribute name="identifierref" use="optional">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:maxLength value="2000"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.identifierref.req">
    <xsd:attribute name="identifierref" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:maxLength value="2000"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.resourcetype.req">
    <xsd:attribute name="type" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:maxLength value="1000"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.structure.req">
    <xsd:attribute name="structure" use="optional" default="hierarchical">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:maxLength value="200"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:attributeGroup name="attr.version">
    <xsd:attribute name="version" use="optional">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:maxLength value="20"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
```

```
<xsd:annotation>
    <xsd:documentation>element groups</xsd:documentation>
</xsd:annotation>
<xsd:group name="grp.any">
    <xsd:annotation>
        <xsd:documentation>Any namespaced element from any
        namespace may be included within an &quot;any&quot; element.
        The namespace for the imported element must be defined in the
        instance, and the schema must be imported.  </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:any namespace="##other" processContents="strict"
        minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:group>
<!-- ************************* -->
<!-- ** Element Declarations ** -->
<!-- ************************* -->
<xsd:element name="dependency" type="dependencyType"/>
<xsd:element name="file" type="fileType"/>
<xsd:element name="item" type="itemType"/>
<xsd:element name="manifest" type="manifestType"/>
<xsd:element name="metadata" type="metadataType"/>
<xsd:element name="organization" type="organizationType"/>
<xsd:element name="organizations" type="organizationsType"/>
<xsd:element name="resource" type="resourceType"/>
<xsd:element name="resources" type="resourcesType"/>
<xsd:element name="schema" type="schemaType"/>
<xsd:element name="schemaversion" type="schemaversionType"/>
<xsd:element name="title" type="titleType"/>
<!-- ******************* -->
<!-- ** Complex Types ** -->
<!-- ******************* -->
<!-- **************** -->
<!-- ** dependency ** -->
<!-- **************** -->
<xsd:complexType name="dependencyType">
    <xsd:sequence>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="attr.identifierref.req"/>
    <xsd:anyAttribute namespace="##other" processContents="strict"/>
</xsd:complexType>
<!-- ********** -->
<!-- ** file ** -->
<!-- ********** -->
```

```
<xsd:complexType name="fileType">
    <xsd:sequence>
        <xsd:element ref="metadata" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="attr.href.req"/>
    <xsd:anyAttribute namespace="##other" processContents="strict"/>
</xsd:complexType>
<!-- ********** -->
<!-- ** item ** -->
<!-- ********** -->
<xsd:complexType name="itemType">
    <xsd:sequence>
        <xsd:element ref="title" minOccurs="0"/>
        <xsd:element ref="item" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="metadata" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="attr.identifier.req"/>
    <xsd:attributeGroup ref="attr.identifierref"/>
    <xsd:attributeGroup ref="attr.isvisible"/>
    <xsd:attributeGroup ref="attr.parameters"/>
    <xsd:anyAttribute namespace="##other" processContents="strict"/>
</xsd:complexType>
<!-- ************** -->
<!-- ** manifest ** -->
<!-- ************** -->
<xsd:complexType name="manifestType">
    <xsd:sequence>
        <xsd:element ref="metadata" minOccurs="0"/>
        <xsd:element ref="organizations"/>
        <xsd:element ref="resources"/>
        <xsd:element ref="manifest" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="attr.identifier.req"/>
    <xsd:attributeGroup ref="attr.version"/>
    <xsd:attribute ref="xml:base"/>
    <xsd:anyAttribute namespace="##other" processContents="strict"/>
</xsd:complexType>
<!-- ************** -->
<!-- ** metadata ** -->
<!-- ************** -->
<xsd:complexType name="metadataType">
    <xsd:sequence>
        <xsd:element ref="schema" minOccurs="0"/>
```

```
        <xsd:element ref="schemaversion" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<!-- ******************* -->
<!-- ** organizations ** -->
<!-- ******************* -->
<xsd:complexType name="organizationsType">
    <xsd:sequence>
        <xsd:element ref="organization" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="attr.default"/>
    <xsd:anyAttribute namespace="##other" processContents="strict"/>
</xsd:complexType>
<!-- ****************** -->
<!-- ** organization ** -->
<!-- ****************** -->
<xsd:complexType name="organizationType">
    <xsd:sequence>
        <xsd:element ref="title" minOccurs="0"/>
        <xsd:element ref="item" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="metadata" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="attr.identifier.req"/>
    <xsd:attributeGroup ref="attr.structure.req"/>
    <xsd:anyAttribute namespace="##other" processContents="strict"/>
</xsd:complexType>
<!-- ***************** -->
<!-- ** resources ** -->
<!-- ***************** -->
<xsd:complexType name="resourcesType">
    <xsd:sequence>
        <xsd:element ref="resource" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
    <xsd:attributeGroup ref="attr.base"/>
    <xsd:anyAttribute namespace="##other" processContents="strict"/>
</xsd:complexType>
<!-- ***************** -->
<!-- ** resource ** -->
<!-- ***************** -->
<xsd:complexType name="resourceType">
    <xsd:sequence>
        <xsd:element ref="metadata" minOccurs="0"/>
```

```
            <xsd:element ref="file" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element ref="dependency" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:group ref="grp.any"/>
        </xsd:sequence>
        <xsd:attributeGroup ref="attr.identifier.req"/>
        <xsd:attributeGroup ref="attr.resourcetype.req"/>
        <xsd:attributeGroup ref="attr.base"/>
        <xsd:attributeGroup ref="attr.href"/>
        <xsd:anyAttribute namespace="##other" processContents="strict"/>
    </xsd:complexType>
    <!-- ****************** -->
    <!-- ** Simple Types ** -->
    <!-- ****************** -->
    <!-- ************ -->
    <!-- ** schema ** -->
    <!-- ************ -->
    <xsd:simpleType name="schemaType">
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="100"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- ******************** -->
    <!-- ** schemaversion ** -->
    <!-- ******************** -->
    <xsd:simpleType name="schemaversionType">
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="20"/>
        </xsd:restriction>
    </xsd:simpleType>
    <!-- *********** -->
    <!-- ** title ** -->
    <!-- *********** -->
    <xsd:simpleType name="titleType">
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="200"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:schema>
```

A.3 imsmd_rootv1p2p1.xsd

We report the XML Schema Definition control document (i.e. the IMS Global Learning Consortium, Inc. Learning Resource metadata Specification Version 1.2.1 imsmd_rootv1p2p1.xsd) used to define the grammars and definition on which the SCORM metadata Application Profiles are based.

```
<?xml version="1.0" encoding="UTF-8"?> <!-edited by Thomas Wason
' <xsd:schema
targetNamespace="http://www.imsglobal.org/xsd/imsmd_rootv1p2p1"
xmlns="http://www.imsglobal.org/xsd/imsmd_rootv1p2p1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" version="1.2:1.1 IMS:MD1.2">
    <xsd:import namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="ims_xml.xsd"/>
    <!-- ******************** '
    <!-- ** Change History ** '
    <!-- ******************** '
    <xsd:annotation>
        <xsd:documentation>2001-04-26 T.D.Wason. IMS metadata 1.2 XML-Schema.
                                        </xsd:documentation>
        <xsd:documentation>2001-06-07 S.E.Thropp. Changed the
        multiplicity on all elements to match the         </xsd:documentation>
        <xsd:documentation>Final 1.2 Binding Specification.
                                                    </xsd:documentation>
        <xsd:documentation>Changed all elements that use the
        langstringType to a multiplicy of 1 or more        </xsd:documentation>
        <xsd:documentation>Changed centity in the contribute
        element to have a multiplicity of 0 or more.       </xsd:documentation>
        <xsd:documentation>Changed the requirement element to
        have a multiplicity of 0 or more.                 </xsd:documentation>
        <xsd:documentation> 2001-07-25 Schawn Thropp.
        Updates to bring the XSD up to speed with the W3C        </xsd:documentation>
        <xsd:documentation> XML Schema Recommendation.
        The following changes were made: Change the             </xsd:documentation>
        <xsd:documentation> namespace to reference the 5/2/2001
        W3C XML Schema Recommendation,the base          </xsd:documentation>
        <xsd:documentation> type for the durtimeType, simpleType,
        was changed from timeDuration to duration.    </xsd:documentation>
        <xsd:documentation> Any attribute declarations that
        have use="default" had to change to use="optional"  </xsd:documentation>
        <xsd:documentation> - attr.type.  Any attribute
        declarations that have value ="somevalue" had to change </xsd:documentation>
        <xsd:documentation> to default = "somevalue" -
        attr.type (URI)                                      </xsd:documentation>
        <xsd:documentation> 2001-09-04 Schawn Thropp
                                                             </xsd:documentation>
        <xsd:documentation> Changed the targetNamespace and
        namespace of schema to reflect version change       </xsd:documentation>
    </xsd:annotation>
    <!-- ************************** '
    <!-- ** Attribute Declaration ** '
```

```
<!-- ************************** ’
<xsd:attributeGroup name="attr.type">
    <xsd:attribute name="type" use="optional" default="URI">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="URI"/>
                <xsd:enumeration value="TEXT"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:attributeGroup>
<xsd:group name="grp.any">
    <xsd:annotation>
        <xsd:documentation>Any namespaced element from any
        namespace may be used for an &quot;any&quot; element.
        The namespace for the imported element must be defined
        in the instance, and the schema must be imported.  </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:any namespace="##any" processContents="strict"
        minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:group>
<!-- ************************** ’
<!-- ** Element Declaration ** ’
<!-- ************************** ’
<xsd:element name="aggregationlevel" type="aggregationlevelType"/>
<xsd:element name="annotation" type="annotationType"/>
<xsd:element name="catalogentry" type="catalogentryType"/>
<xsd:element name="catalog" type="catalogType"/>
<xsd:element name="centity" type="centityType"/>
<xsd:element name="classification" type="classificationType"/>
<xsd:element name="context" type="contextType"/>
<xsd:element name="contribute" type="contributeType"/>
<xsd:element name="copyrightandotherrestrictions"
type="copyrightandotherrestrictionsType"/>
<xsd:element name="cost" type="costType"/>
<xsd:element name="coverage" type="coverageType"/>
<xsd:element name="date" type="dateType"/>
<xsd:element name="datetime" type="datetimeType"/>
<xsd:element name="description" type="descriptionType"/>
<xsd:element name="difficulty" type="difficultyType"/>
<xsd:element name="educational" type="educationalType"/>
<xsd:element name="entry" type="entryType"/>
<xsd:element name="format" type="formatType"/>
<xsd:element name="general" type="generalType"/>
```

```
<xsd:element name="identifier" type="xsd:string"/>
<xsd:element name="intendedenduserrole" type="intendedenduserroleType"/>
<xsd:element name="interactivitylevel" type="interactivitylevelType"/>
<xsd:element name="interactivitytype" type="interactivitytypeType"/>
<xsd:element name="keyword" type="keywordType"/>
<xsd:element name="kind" type="kindType"/>
<xsd:element name="langstring" type="langstringType"/>
<xsd:element name="language" type="xsd:string"/>
<xsd:element name="learningresourcetype" type="learningresourcetypeType"/>
<xsd:element name="lifecycle" type="lifecycleType"/>
<xsd:element name="location" type="locationType"/>
<xsd:element name="lom" type="lomType"/>
<xsd:element name="maximumversion" type="minimumversionType"/>
<xsd:element name="metadatascheme" type="metadataschemeType"/>
<xsd:element name="metametadata" type="metametadataType"/>
<xsd:element name="minimumversion" type="maximumversionType"/>
<xsd:element name="name" type="nameType"/>
<xsd:element name="purpose" type="purposeType"/>
<xsd:element name="relation" type="relationType"/>
<xsd:element name="requirement" type="requirementType"/>
<xsd:element name="resource" type="resourceType"/>
<xsd:element name="rights" type="rightsType"/>
<xsd:element name="role" type="roleType"/>
<xsd:element name="semanticdensity" type="semanticdensityType"/>
<xsd:element name="size" type="sizeType"/>
<xsd:element name="source" type="sourceType"/>
<xsd:element name="status" type="statusType"/>
<xsd:element name="structure" type="structureType"/>
<xsd:element name="taxon" type="taxonType"/>
<xsd:element name="taxonpath" type="taxonpathType"/>
<xsd:element name="technical" type="technicalType"/>
<xsd:element name="title" type="titleType"/>
<xsd:element name="type" type="typeType"/>
<xsd:element name="typicalagerange" type="typicalagerangeType"/>
<xsd:element name="typicallearningtime" type="typicallearningtimeType"/>
<xsd:element name="value" type="valueType"/>
<xsd:element name="person" type="personType"/>
<xsd:element name="vcard" type="xsd:string"/>
<xsd:element name="version" type="versionType"/>
<xsd:element name="installationremarks" type="installationremarksType"/>
<xsd:element name="otherplatformrequirements"
type="otherplatformrequirementsType"/>
<xsd:element name="duration" type="durationType"/>
<xsd:element name="id" type="idType"/>
<!-- ******************* '
<!-- ** Complex Types ** '
```

```
<!-- ******************** ’
<xsd:complexType name="aggregationlevelType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="annotationType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="person" minOccurs="0"/>
        <xsd:element ref="date" minOccurs="0"/>
        <xsd:element ref="description" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="catalogentryType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="catalog"/>
        <xsd:element ref="entry"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="centityType">
    <xsd:sequence>
        <xsd:element ref="vcard"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="classificationType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="purpose" minOccurs="0"/>
        <xsd:element ref="taxonpath" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="description" minOccurs="0"/>
        <xsd:element ref="keyword" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="contextType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="contributeType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="role"/>
        <xsd:element ref="centity" minOccurs="0" maxOccurs="unbounded"/>
```

```
        <xsd:element ref="date" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="copyrightandotherrestrictionsType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="costType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="coverageType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="dateType">
    <xsd:sequence>
        <xsd:element ref="datetime" minOccurs="0"/>
        <xsd:element ref="description" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="descriptionType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="difficultyType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="durationType">
    <xsd:sequence>
        <xsd:element ref="datetime" minOccurs="0"/>
        <xsd:element ref="description" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="educationalType" mixed="true">
    <xsd:sequence>
```

```
        <xsd:element ref="interactivitytype" minOccurs="0"/>
        <xsd:element ref="learningresourcetype" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="interactivitylevel" minOccurs="0"/>
        <xsd:element ref="semanticdensity" minOccurs="0"/>
        <xsd:element ref="intendededenduserrole" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="context" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="typicalagerange" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="difficulty" minOccurs="0"/>
        <xsd:element ref="typicallearningtime" minOccurs="0"/>
        <xsd:element ref="description" minOccurs="0"/>
        <xsd:element ref="language" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="entryType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="generalType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="identifier" minOccurs="0"/>
        <xsd:element ref="title" minOccurs="0"/>
        <xsd:element ref="catalogentry" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="language" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="keyword" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="coverage" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="structure" minOccurs="0"/>
        <xsd:element ref="aggregationlevel" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="installationremarksType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="intendededenduserroleType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="interactivitylevelType">
    <xsd:sequence>
```

```
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="interactivitytypeType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="keywordType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="kindType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="langstringType">
    <xsd:simpleContent>
        <xsd:extension base="xsd:string">
            <xsd:attribute ref="xml:lang"/>
        </xsd:extension>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="learningresourcetypeType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="lifecycleType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="version" minOccurs="0"/>
        <xsd:element ref="status" minOccurs="0"/>
        <xsd:element ref="contribute" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="locationType">
    <xsd:simpleContent>
        <xsd:extension base="xsd:string">
            <xsd:attributeGroup ref="attr.type"/>
```

```
        </xsd:extension>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="lomType">
    <xsd:sequence>
        <xsd:element ref="general" minOccurs="0"/>
        <xsd:element ref="lifecycle" minOccurs="0"/>
        <xsd:element ref="metametadata" minOccurs="0"/>
        <xsd:element ref="technical" minOccurs="0"/>
        <xsd:element ref="educational" minOccurs="0"/>
        <xsd:element ref="rights" minOccurs="0"/>
        <xsd:element ref="relation" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="annotation" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="classification" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="metametadataType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="identifier" minOccurs="0"/>
        <xsd:element ref="catalogentry" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="contribute" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="metadatascheme" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="language" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="nameType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="otherplatformrequirementsType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="personType">
    <xsd:sequence>
        <xsd:element ref="vcard"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="purposeType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
```

```
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="relationType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="kind" minOccurs="0"/>
        <xsd:element ref="resource" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="requirementType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="type" minOccurs="0"/>
        <xsd:element ref="name" minOccurs="0"/>
        <xsd:element ref="minimumversion" minOccurs="0"/>
        <xsd:element ref="maximumversion" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="resourceType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="identifier" minOccurs="0"/>
        <xsd:element ref="description" minOccurs="0"/>
        <xsd:element ref="catalogentry" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="rightsType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="cost" minOccurs="0"/>
        <xsd:element ref="copyrightandotherrestrictions" minOccurs="0"/>
        <xsd:element ref="description" minOccurs="0"/>
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="roleType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="semanticdensityType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
```

```
<xsd:complexType name="sourceType">
    <xsd:sequence>
        <xsd:element ref="langstring"/>
    </xsd: equenze>
</xsd:complexType>
<xsd:complexType name="statusType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="stringType">
    <xsd:simpleContent>
        <xsd:extension base="xsd:string">
            <xsd:attribute ref="xml:lang"/>
        </xsd:extension>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="structureType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="taxonpathType">
    <xsd:sequence>
        <xsd:element ref="source" minOccurs="0"/>
        <xsd:element ref="taxon" minOccurs="0"/>
    </xsd: equenze>
</xsd:complexType>
<xsd:complexType name="taxonType">
    <xsd:sequence>
        <xsd:element ref="id" minOccurs="0"/>
        <xsd:element ref="entry" minOccurs="0"/>
        <xsd:element ref="taxon" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="technicalType" mixed="true">
    <xsd:sequence>
        <xsd:element ref="format" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="size" minOccurs="0"/>
        <xsd:element ref="location" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="requirement" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element ref="installationremarks" minOccurs="0"/>
        <xsd:element ref="otherplatformrequirements" minOccurs="0"/>
        <xsd:element ref="duration" minOccurs="0"/>
```

```
        <xsd:group ref="grp.any"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="titleType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="typeType">
    <xsd:sequence>
        <xsd:element ref="source"/>
        <xsd:element ref="value"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="typicalagerangeType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="typicallearningtimeType">
    <xsd: equenze>
        <xsd:element ref="datetime" minOccurs="0"/>
        <xsd:element ref="description" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="valueType">
    <xsd:sequence>
        <xsd:element ref="langstring"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="versionType">
    <xsd:sequence>
        <xsd:element ref="langstring" maxOccurs="unbounded"/>
    </xsd: equenze>
</xsd:complexType>
<!-- ****************** '
<!-- ** Simple Types ** '
<!-- ****************** '
<xsd:simpleType name="formatType">
    <xsd:restriction base="xsd:string"/>
</xsd:simpleType>
<xsd:simpleType name="sizeType">
    <xsd:restriction base="xsd:int"/>
</xsd:simpleType>
<xsd:simpleType name="datetimeType">
    <xsd:restriction base="xsd:string"/>
```

```
    </xsd:simpleType>
    <xsd:simpleType name="idType">
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="metadataschemeType">
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="catalogType">
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="minimumversionType">
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
    <xsd:simpleType name="maximumversionType">
        <xsd:restriction base="xsd:string"/>
    </xsd:simpleType>
</xsd:schema>
```

A.4 SCORM: ims_xml.xsd

We report the XML schema referred by the IMS Global Learning Consortium, Inc. Learning Resource metadata Specification and Content Packaging Specification.

```
<?xml version="1.0" encoding="UTF-8"?> <!-- filename=ims_xml.xsd
--> <xsd:schema
targetNamespace="http://www.w3.org/XML/1998/namespace"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.w3.org/XML/1998/namespace"
elementFormDefault="qualified">
    <!-- 2001-02-22 edited by Thomas Wason IMS Global Learning Consortium, Inc. -->
    <xsd:annotation>
        <xsd:documentation>In namespace-aware XML processors,
        the &quot;xml&quot; prefix is bound to the namespace
        name http://www.w3.org/XML/1998/namespace.</xsd:documentation>
        <xsd:documentation>Do not reference this file in XML
        instances</xsd:documentation>
        <xsd:documentation>Schawn Thropp: Changed the uriReference
        type to string type</xsd:documentation>
    </xsd:annotation>
    <xsd:attribute name="lang" type="xsd:language">
        <xsd:annotation>
            <xsd:documentation>Refers to universal  XML 1.0 lang
            attribute</xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
```

```
    <xsd:attribute name="base" type="xsd:string">
        <xsd:annotation>
            <xsd:documentation>Refers to XML Base:
            http://www.w3.org/TR/xmlbase</xsd:documentation>
        </xsd:annotation>
    </xsd:attribute>
    <xsd:attribute name="link" type="xsd:string"/>
</xsd:schema>
```

B MPEG-7 XML-Schemas

B.1 ddl-2001.xsd

We report the XML schema used to define the MPEG-7 Description Definition Language.

```
<?xml version="1.0" encoding="UTF-8"?> <!--
################################################################################
--> <!-- ISO/IEC 15938 Information Technology - Multimedia Content
Description Interface  --> <!-- Part 2: Description Definition
Language (ISO/IEC 15938-2)                                  --> <!--
################################################################################
--> <schema targetNamespace="urn:mpeg:mpeg7:schema:2001"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2001"
xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
    <annotation>
        <documentation>
            This document contains tools defined as MPEG-7 specific extention of
            XML Schema in ISO/IEC 15938-2
        </documentation>
    </annotation>
    <!-- ################################################ -->
    <!-- Definition of 'mpeg7:dim" for Matrix Datatype                  -->
    <!-- ################################################ -->
    <!-- definition of listOfPositiveIntegerForDim datatype -->
    <simpleType name="listOfPositiveIntegerForDim">
        <list itemType="positiveInteger"/>
    </simpleType>
    <!-- definition of mpeg7:dim attribute -->
    <attribute name="dim">
        <simpleType>
            <restriction base="mpeg7:listOfPositiveIntegerForDim">
                <minLength value="1"/>
            </restriction>
        </simpleType>
```

```
    </attribute>
    <!-- ################################################ -->
    <!-- Definition of MPEG-7 Datatype Extensions                    -->
    <!-- ################################################ -->
    <!-- definition of basicTimePoint dataype -->
    <simpleType name="basicTimePointType">
        <restriction base="string">
            <pattern value="\-?(\d+(\-\d{2}(\-\d{2})?)?)?(T\d{2}(:\d{2}(:\d{2}
            (:\d+(\.\d{2})?)?)?)?)?(F\d+)?((\-|\+)\d{2}:\d{2})?"/>
        </restriction>
    </simpleType>
    <!-- definition of basicDuration dataype -->
    <simpleType name="basicDurationType">
        <restriction base="string">
            <pattern value="\-?P(\d+D)?(T(\d+H)?(\d+M)?(\d+S)?(\d+N)?
            (\d{2}f)?)?(\d+F)?((\-|\+)\d{2}:\d{2}Z)?"/>
        </restriction>
    </simpleType>
</schema>
```