

## Using XQuery Language for Retrieving Multimedia Information

**Mihaela BRUT**

*Faculty of Computer Science, "Al. I. Cuza" University of Iasi  
Ge..Berthelot Str., 16, Iasi-700483, Romania  
mihaela@infoiasi.ro*

### **Abstract**

*The wide promotion, with the WWW Consortium support, of the XML as the standard format for the data published on the Web leads to multiple researches in order to define query languages for XML documents. In this paper we present the general framework of the XML information retrieval on Web, with a special lookout to the XQuery language. Also, we shall focus on the different possible techniques for querying the multimedia information in order to obtain the result as XML, SMIL and RDF sequences.*

**Keywords:** *query languages, XML documents, multimedia retrieval.*

### **Introduction**

The broad promotion of XML (Extensible Markup Language) as the standard meta-language applied to define markups for Web documents encouraged multiple researches on promising XML query languages. XML is already used to represent many different kinds of data: web pages, web messages, books, business and banking data and applications, XML representations of relational database tables, programming interfaces, chess games, vector graphics, multimedia presentations, and so on. In addition, some systems offer XML views of non-XML data sources such as relational databases, allowing XML-based processing of data that is not physically represented as XML. Departing from the XML structure of the Web documents, the XML query languages provide different ways of intelligent information retrieval, refining the keywords based search.

In this paper, we shall present the general framework of the querying XML documents problem, followed by a survey on the XQuery language – the WWW Consortium recommendation, currently in process of standardization. We shall discuss the possibility of querying Web multimedia information applying XQuery to SMIL documents, in order to obtain the results in the same format, or as an RDF document containing metadata about temporal behaviour of the multimedia information.

### **The XML querying framework**

The World Wide Web space became an enormous reservoir of information, being a needful work instrument for all researchers. Many scientific data collections are now available on the Web. The problem of the efficient access and management of the growing information became more and more critically. The enormous quantity of available data makes the results of searching a certain subject using a traditional search engine to include hundreds or even thousands Web pages, often provided in an irrelevant order of relevance. The main motive is that the search engines don't take into account the structure of the Web documents containing the requested keywords. The XML query languages and techniques tried to address this deficiency by extrapolating the techniques of querying databases and processing text documents for exploiting the XML documents structure.

As a standard recommended by the Web Consortium, XML (eXtensible Markup Language) [W3C] is strikingly considered as the data format for information interchanging between the sundries Internet applications. The XML popularity is mainly due to its flexibility in the representation of many data types. The use of mark-ups confer to the XML language the possibility of auto description, and its extensible nature makes possible the definition of new document types, with a special destination.

Alongside XML, appeared a series of other Web standards which were adopted, as auxiliaries, in the query languages specifications. For example, XML Schema is a notation for defining new types of elements and documents. XPath language offers a notation for selecting elements from an XML document. XSLT (Extensible Stylesheet Language Transformations) provide a notation for transforming an XML document from a representation to another (for example, from XHTML in SMIL). All this standards are related and intended for XML documents.

Because XML is the standard format for interchanging Web information, it is naturally that queries applied to different types of documents to be expressed as queries on XML data. For this reasons, the necessity of a standard specialized XML query language became stringent.

In December 1998, the Web Consortium organized a conference on the subject of “Query Languages” with two main purposes: establishing the requirements for an XML query language, and providing of concrete solutions. Between the 66 papers of the conference, there were multiple proposals of languages specialized in information extraction from XML documents, as *XML-QL*, *XQL* or *XML-GL* [Cer99, Deu99, Rob98], which modulate the techniques of databases querying and text processing.

For the efficient Web search purpose, there exist sundries proposals of techniques for locating, indexing, querying, and reorganizing the Web sites content. A premier significant direction combined the keywords based search with various techniques adapting the databases query support for the Web space querying [Flo98], as in the case of Web3QL, WebSQL, WebOQL [Aro98], XQL, XML-QL languages – modelled after SQL (Structured Query Language) –, or WebLog – inspired from DataLog language. The techniques of query processing over databases encountered an evolution from the relational databases, to object oriented ones, or to semi structured databases, but the principles keep similarities. The specific problems regard the storage of huge data repositories, the integration of heterogeneous data derived from different sources, the export of new perspectives for inherited data, or the conversion to data formats easy to interchange [Fer99]. The mechanism used in the semi structured databases querying was adopted especially by three XML query languages: XML-QL [Deu99], YALT [Clu98, Clu99] and Lorel [Abi97, Gol99].

Another direction in formalizing XML queries followed the principles specific to the text processing applications, for the operations as searching inside a plain text, querying the structured documents, integration of plain text in the structured queries, derivation of multiple presentation forms from an initial text. Text processing applications are based on different formal model, as the regions algebra [Cla95] for representing the structured text or querying techniques, the principles being modulated to XML documents querying by the XQL language [Ish98].

The query languages for XML documents are based on the following idea: they offer to the user the possibility to formulate a query in a specific manner and generate a new XML document as a pattern for this query. It shall be compared with the target XML document and shall be retained and returned only matched data, under the restructured form of a new XML document. The query languages themselves are implemented as XML documents.

We proposed an XML-based query language – called *Web Query Graphical Language* (WQGL) [5] – in order to be used for building graphical interfaces which will assist users to formulate complex queries. The components (widgets) of the graphical interface are defined by *XUL* (*Extensible User-interface Language*) [4, 12] elements. WQGL is designed for queries related to the structure of desired Web pages, and constitute an extension of the *Web Query Graphical Language* (WQFL) [Bur00]. Alongside the query languages, there were developed many software tools which implement them together with other specific retrieval techniques: WebVCR [Anu0], SearchPad [Bha00], COVA [Cha02]. It must be mentioned that there could be used many facilities provided by the existing programming languages (such Perl, C, PHP etc.) to process the Web documents.

In October 1999, the Web Consortium decided the establishing of the XML Query working group [Xml-query], in order to design an XML query language, named XQuery, currently in process of standardization.

Many of the requirements established in December 1998 were kept for the XQuery language. We list below the most important of this:

- data extraction from huge XML documents, by homogeneously processing both the structured part (elements, attributes, values) and the properly text;
- syntax based on other XML standards, such as XPath, XPointer, XSLT;
- XML data transfer between documents having different ontologies (DTDs);
- querying the distributed data, in different XML formats, and the integration of XML results from multiple sources;
- support for standard querying operations: selection, extraction, reduction, reorganization, combination;
- verifying the rightness of query results.

In order to pass towards the “Semantic Web” [Ber01], under the guidance of the Web Consortium, there where developed a series of XML-based languages specialized in the modeling of the knowledge, in the same time appearing specific query languages for documents marked up according to these: DQL (for DAML+OIL, OWL), Quest (for XHTML).

The foundation for the semantically querying of the Web content is provided by the RDF (Resource Description Framework) [2, 10], which allow the attachment of metadata to the XML documents. Thus, for the retrieval of the semantic information on the Web, there where developed a series of query languages for RDF documents: Algae, Squish, WebML, QinetiQ, RQL, DQL, RDQL, RuleML, etc. [Mil03, Pru03].

## **The XQuery Language**

The Web Consortium constituted the XML Query working group in order to design the XQuery language intended for querying the XML documents. The new language is in process of standardization, and was described through a series of working draft documents. The most important is “XQuery 1.0: An XMLQuery Language” [XQuery], containing the syntactic description of the language. The working group also published a list of requirements for XQuery language [Xquery-req], the language data model [Xq-dm], a formal semantic description of it [Xq-sem], a list of functions and operators [Xq-op], and a collection of case studies which illustrates the language appliance. XQuery is a functional language, containing some expression types which can be combined, and being based on the datatype system from XML Schema, so that to be compatible with the related XML standards.

The initial design of the XQuery language is focused on the information retrieval by querying the desired XML documents. After this version shall be finalized, the XML Query working group shall debate the possible extensions of the XQuery so that to provide facilities of modifying the queried XML documents.

We present below a syntactic survey of the XQuery language, followed by a query example applied to a SMIL document.

### *XQuery syntax*

The XQuery syntax was constituted by incorporating the numerous influences [Cha02], among which the most important are the following standards: W3C XPath (XQuery being a superset of XPath) [XPath], XML Schema [Xschema], XSLT [Xsl], and XML itself [W3C]. The general design of XQuery is based on a language proposal named Quilt [Cha00], which is – on its turn – influenced by the functional onset of the OQL (Object Query Language) [ATW96], using the keywords based syntax

of SQL (Structured Query Language) and the previous proposals of query languages, as XQL [Rob98], XML-QL [Deu99], and Lorel [Abi97].

Being a functional language, XQuery is constituted by expressions that return values and do not have side effects. XQuery has several kinds of expressions – in majority, composed from lower-level expressions, combined by operators or keywords. The simplest kind of XQuery expression is a literal, which represents an atomic value, and could be numerical (integer, decimal, double) or strings.

XQuery allow to create atomic values of other types – such elements, attributes, text nodes, processing instructions, comments – by calling constructors. A *constructor* is a function that creates a value of a particular type from a string containing a lexical representation of the desired type. In general, a constructor has the same name as the type it constructs. For example, the following constructor creates a value of type date: date("2003-12-14"). In XQuery could be used variables, their names beginning with a dollar sign.

XQuery provides a core function library, and a mechanism whereby users can define additional functions. For example, the substring() function could be used to extract seven characters from a string, beginning with the fourth one: substring("Tim Berners-Lee", 4, 6).

For referring to a specific XML subelement, there could be used XQuery path expressions, which are based on the syntax of XPath [XPath]. A *path expression* consists of a series of steps, separated by the slash character: doc("books.xml")/bib/book. For selecting the subelements which satisfy certain conditions, XQuery allow the use of *predicates*: doc("books.xml")/bib/book[@year="1999"].

One of the most powerful and common expressions in XQuery are FLWOR expressions, pronounced “flower”. They are similar to the SELECT-FROM-WHERE statements in SQL.

The name FLWOR is an acronym, standing for the first letter of the clauses that may occur in a FLWOR expression:

- **for** clauses: associate one or more variables to expressions, creating a tuple stream in which each tuple binds a given variable to one of the items to which its associated expression evaluates;
- **let** clauses: bind variables to the entire result of an expression, adding these bindings to the tuples generated by a for clause, or creating a single tuple to contain these bindings if there is no for clause;
- **where** clauses: filter tuples, retaining only those tuples that satisfy a condition;
- **order by** clauses: sort the tuples in a tuple stream;
- **return** clauses: build the result of the FLWOR expression for a given tuple.

The XQuery language provide, also, an entire set of operators:

- arithmetic operators: +, -, \*, div, mod
- comparison operators: eq, ne, lt, le, gt, ge, =, !=, <, <=, >, >=
- sequence operators: is, is not, <<, >>

For example, if the “books.xml” document contains elements <book> having the attribute “year” and the subelements <title>, <author>, and <publisher>, the next query constructs a list of books containing the “Programming” or “Languages” word inside the title. For each such book, is specified the title as the content of the <book> element, and the year of publication as the attribute.

```
<list>
for $b in doc("http://infoiasi.ro/~mihaela/books.xml")/bib/book
let $e := $b/title[contains(string(.), "Programming ")
                    or contains(string(.), "Application")]
where exists($e)
return
  <book year="{ $b[@year] }">
    { $e }
  </book>
</list>
```

## Using XQuery Constructs for Synchronized Multimedia Retrieval

The XQuery language can be applied to all kinds of XML documents, that is to the documents annotated in any XML-based language. We shall focus in this article on the possibilities of retrieving multimedia information on Web using the XQuery language. Because the SMIL language is a Web Consortium's standard in order to annotate information about the temporal scenario of different Web multimedia objects, we shall first illustrate the querying modalities of SMIL documents using XQuery language, obtaining results in the form of SMIL sequences.

A very important aspect of a multimedia presentation is its temporal scenario, for example in order to coordinate a certain activity with a fragment of a such presentation. We shall apply XQuery for obtaining RDF annotations about the temporal aspects of a SMIL document. In final, we shall present a technique of querying RDF sequences using XQuery.

### Querying SMIL documents

#### *A short presentation of SMIL*

SMIL (Synchronized Multimedia Integration Language) is an XML-based language [12] developed since 1998 by the Web Consortium in order to facilitate the creation of interactive multimedia presentations. SMIL enables authors to describe the temporal behavior of a multimedia presentation, associate hyperlinks with media objects or describe the layout of the presentation on a screen. A presentation is composed from several components, each including different media types, such as audio, video, image or text, and could be executed sequential, parallel or in a combined manner. Control buttons such as stop, fast-forward and rewind allow the user to interrupt the presentation and to move forwards or backwards to another point in the presentation.

SMIL 2.0 – the actual version of the language – is defined as a set of reusable markup (annotation) modules. This allows reuse of SMIL syntax and semantics in other XML-based languages, in particular inside those that need to represent timing and synchronization [14]. For example, SMIL 2.0 components are used for integrating timing into XHTML [13] and into SVG [15]. There are special players for SMIL developed by different companies, such as the RealOne of RealNetworks or Oratrix's GRiNS player and editor. The general trend is to incorporate support for SMIL even in the Web browsers: Internet Explorer 5.5 and up plays XHTML+SMIL [10], Apple's QuickTime version 4.1 or later supports SMIL 1.0 and Adobe's SVG Viewer supports SMIL animation in SVG [16].

Example. The SMIL multimedia presentations are easy to be written and do not require sophisticated authoring tools, because there are simply text XML-based files. As an example, we consider the following SMIL document, called *books.smil*, which contains information about diverse published books and their associated synchronized video-clips. The presentation will split the computer screen into two regions, a movie (in the .avi format) and a text file being displayed in parallel for the movie duration, each in a specific region, and then the movie being replaced by an image for 20 seconds:

```
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 2.0//EN"
  "http://www.w3.org/TR/REC-smil/SMIL20.dtd">
<smil xmlns="http://www.w3.org/2000/SMIL20/CR/Language">
<head>
  <layout type="text/smil-basic-layout">
    <!--we define first the size of the display region -->
    <root-layout width="650" height="375"/>
    <!-- SMIL definitions for the size of each sub-region -->
```

```

    <region id="rvideo" top="35" left="10" height="330" width="300" />
    <region id="rtext" top="35" left="340" height="330" width="300" />
</layout>
</head>
<body>
<seq>
<b:books xmlns:b="http://infoiasi.ro/~mihaela/books.dtd">
  <par>
    <seq>
      <video region="rvideo" begin="0s" end="15s" id="v1" src="../movies/web.avi" />
      
    </seq>
    <text region="rtext" dur="v1.dur+20s">
      <b:title>Web Technologies</b:title>
      <b:author>S. Buraga</b:author>
      <b:publisher>MatrixRom</b:publisher>
      <b:year>2001</b:year>
    </text>
  </par>
<!-- SMIL constructs for presenting other books -->
</b:books>
</body>
</smil>

```

### *Examples of queries on SMIL documents*

For generating a slide-show containing all title-author pairs, accompanied by the corresponding book image for 20 seconds, we can compose the following XQuery assertion:

```

<seq>
{
  for $b in doc("http://www.infoiasi.ro/~mihaela/books.smil")/body/seq/par
    $t in $b/text/b:title
    $a in $b/text/b:author
  where $a != "et al. "
  return
  <par>
    <text region="rtext" dur="20s">
      <b:title>{$t}<b:title>, <b:author>author: {$a}</b:author>
    </text>
    
  </par>
}
</seq>

```

A part of this query result could be:

```

<seq>
  <par>

```

```

<text region="rtext" dur="20s">
  <b:title> Web Technologies <b:title>,
  <b:author>author: S. Buraga</b:author>
</text>

</par>
<!-- SMIL constructs referring to other authors and books -->
</seq>

```

In the second example, we'll generate a SMIL presentation which includes, for each author, the title and the image of all his books derulated in a sequential manner. The XQuery construct is:

```

<seq>
{
  let $a := doc("http://www.infoiasi.ro/~mihaela/books.smil")/body/b:books/seq/par
  for $name in distinct-values($a/text/b:author)
  where $name != "et al."
  order by $name
  return
    <par dur="2min">
      <text region="rtext"><b:author>{ $name }<b:author></text>
      <seq>
        {
          for $b in doc("http://www.infoiasi.ro/~mihaela/books.smil")/body/b:books/seq/par
          where some $ba in $b/text/b:books satisfies ($ba = $name)
          return { 
            <text region="rtext" dur="indefinite">
              <b:title>{ $ba/b:title }</b:title></text>
            }
        }
      </seq>
    </par>
}
</seq>

```

A sequecne of this query result follows:

```

<seq>
<par dur="2min">
  <text region="rtext"><b:author>S. Buraga<b:author></text>
  <seq>
    
    <text region="rtext" dur="indefinite">
      <b:title>Web Technologies</b:title></text>
    
    <text region="rtext" dur="indefinite">
      <b:title>Bash and Perl Web Programming</b:title></text>
    
  </seq>
</par>

```

```
<!-- SMIL constructs referring other authors and their books -->
</seq>
```

## Obtaining RDF temporal information

### *A short presentation of RDF*

Resource Description Framework (RDF) [2, 10] is a standardized basis for processing metadata. The used metadata format should permit to reason about data. The RDF is intended to be used to capture and state the conceptual structure of information offered in the Web. The RDF assertions can be viewed as a data model for describing machine processable semantics of data to build the infrastructure for Berners-Lee's Semantic Web.

RDF consists of a model for the representation of named properties and property values. RDF properties may be thought of as attributes of resources and in this sense correspond to conventional attribute-value pairs. RDF properties also characterize relationships between resources and therefore a RDF model can resemble an entity-relationship diagram.

To facilitate the definition of metadata, RDF is based on classes. A collection of classes, typically designed for a specific purpose or domain, is called a schema [11]. Through the sharability of schemas, RDF supports the reusability of metadata definitions. The RDF schemas may themselves be written in RDF.

The basic model of RDF consists of three object types [Xml-query]:

- *Resources*: all objects being described by RDF expressions are called resources and they are always named by Uniform Resource Identifiers (URI) plus optional anchor identifiers. Using URI schemas (i.e. http, ftp, or file schemas), every type of resource can be identified in the same manner.
- *Properties*: A property is an explicit aspect, characteristic, attribute, or relation to express a resource. Each property has a specific meaning, defines its permitted values, the type of resources it can state, and its relationship with other properties (via RDF Schema).
- *Statements* : A specific resource together with a named property, plus the value of that property for that resource is an RDF statement. These three individual parts of a statement are named, respectively, the subject, the predicate, and the object. The object of a statement (e.g., the property value) can be another resource or a literal.

Also, RDF specifies three types of container objects:

- Bag – an unordered list of resources or literals,
- Sequence – an ordered list of resources or literals,
- Alternative – a list of resources or literals that represent alternatives for the single value of a property.

The containers may be defined by an URI pattern. RDF can also be used to make statements about other RDF statements (higher-order statements). The RDF data model provides an abstract, conceptual

framework for defining and using metadata. Currently, there are several proposals of model-theoretic semantics for RDF and RDF Schema [17, 18].

The concrete RDF syntax is based on Extensible Markup Language (XML) elements and attributes. The EBNF grammar of the RDF/XML constructs can be found in [2, 10].

The declarative languages for querying different RDF resources and related schemes are in the proposal phase for the moment being. These languages are very necessary in the context of the large scale semantic Web applications, such *Knowledge Portals* or *E-Marketplaces*, which have to manage voluminous RDF description databases. For example, in the knowledge portals as Open Directory

Project (ODP), CNET, XMLTree [KPex], the information resources such Web sites, scientific articles, etc. are pieced together and classified in enormous hierarchies of thematic categories and subjects. These descriptions are processed in order to personalize the access to portal using standards as RDF Site Summary [Beg00]. In addition, the entire portal catalog can be exported in RDF format. At the moment being, the search in the portal catalogs is limited to the keywords-based retrieval or to the thematic navigation.[Kar02]

*Examples of queries on SMIL documents resulting an RDF sequence*

For example, the following RDF document, named temporal.rdf, contains the temporal information of the previous document books.smil:

```

<!--the list of the multimedia resources having associated temporal attributes -->
<rdf:Bag id="multimedia_resources">
  <rdf:li>
    <rdf:Description about="../movies/web.avi">
      <d:meta id="v1">
        <d:type elem="video">video/avi</d:type>
        <d:begin>0s</d:begin>
        <d:end>15s</d:end>
        <d:dur>15s</d:dur>
      </d:meta>
    </rdf:Description>
  </rdf:li>
  <rdf:li>
    <rdf:Description about="../images/web.jpg">
      <d:meta>
        <d:type elem="img">image/jpeg</d:type>
        <d:begin>v1.end</d:begin>
        <d:dur>20s</d:dur>
      </d:meta>
    </rdf:Description>
  </rdf:li>
  <!--other similar RDF constructs about the other books -->
</rdf:Bag>
<!--the source file containing the multimedia resources collection -->
<rdf:Description about="#multimedia_resources">
  <source href="http://www.infoiasi.ro/~mihaela/books.smil" />
</rdf:Description>

```

This RDF sequence could be obtained applying the following XQuery construct to the book.smil document:

```

<rdf:Bag id="multimedia_resources">
{
for $b in doc("http://www.infoiasi.ro/~mihaela/books.smil")/body/seq/par/seq
  $v in $b/video
  $i in $b/img
return
{
  <rdf:li>

```

```

<rdf:Description about="$v[@src]">
  <d:meta id="$v[@id]">
    <d:type elem="video">video/avi</d:type>
    <d:begin>{$v[@begin]}</d:begin>
    <d:end>{$v[@end]}</d:end>
    <d:dur>{$v[@dur]}</d:dur>
  </d:meta>
</rdf:Description>
</rdf:li>
<rdf:li>
  <rdf:Description about="$i[@img]">
    <d:meta>
      <d:type elem="img">image/jpeg</d:type>
      <d:begin>{$i[@begin]}</d:begin>
      <d:dur>{$i[@dur]}</d:dur>
    </d:meta>
  </rdf:Description>
</rdf:li>
}
}
</rdf:Bag>
<rdf:Description about="#multimedia_resources">
  <source href="{ doc()}" />
</rdf:Description>

```

The above RDF sequence could be enhanced with meta-information relating to the content of the multimedia object. This information could be extracted from the “title”, “alt”, and “longdesc” attributes, which associate a title, an alternative text, and, respectively, a long description for the corresponding multimedia element. It is very important to specify these attributes for a multimedia element, because they provide the main modality for describing the content of a multimedia object. Thus, inside <d:meta> element could also appear:

```

<d:meta>
  <d:title>{$i[@title]}</d:title>
  <d:alt>{$i[@alt]}</d:alt>
  <d:longdesc>{$i[@longdesc]}</d:longdesc>
</d:meta>

```

From another XML document, there could be obtained the following supplementary RDF metadata about the creators of multimedia object, which are in the same time the authors of the books. We call this document books.rdf:

```

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description about="../movies/web.avi">
    <s:Creator rdf:resource="http://www.infoiasi.ro/~busaco" />
  </rdf:Description>
  <rdf:Description about="http://www.infoiasi.ro/~busaco">
    <v:name>Sabin Buraga</v:name>
    <v:email>busaco@infoiasi.ro</v:email>
  </rdf:Description>
</rdf:RDF>

```

## Querying RDF metadata

There isn't established yet a standardized RDF query language, but there are many proposals, and even implementations in use, for RDF query languages. The reason of late for this problem is open discussion on the RDF query formalization.

A RDF document can be represented as a oriented graph, which nodes are represented by resources or their values, and which arches illustrates the correspondings properties. The result of querying a such document can be considered as a sub-graph, extracted according to the query requirements. So that, many RDF query languages identify a query with a RDF graph with parts missing, assign those parts variable names, and get a series of bindings to those variables: RDFdb QL, Algae, Squish, RDQL, RDFql. Some of these languages (especially RQL) have a specific syntax for accesing information from the RDF Schemas (subclasses, properties, etc.), while others have a XML Path-like structure that can recursively match subgraphs (like Versa language) [Mil03, Pru03].

Because Xquery is now in the final phase of standardization, and is considered a universal query language for all the data sources that can be represented in a universal data format, XML, it could be applied for querying RDF metadata. It must be retained that it is not recommended to use the XQuery language for querying RDF documents without a preliminary preprocessing. The main reason is that RDF has many syntactic forms for representing the same logical content, so it is difficult to apply a XML-based query format like XQuery or XSLT to a certain RDF sequence.

For example, the previous RDF sequence can be transformed in the following two RDF equivalent representations:

```
<rdf:RDF>
  <rdf:Description about="../movies/web.avi">
    <s:Creator>
      <rdf:Description about="http://www.infoiasi.ro/~busaco">
        <v:Name>Sabin Buraga</v:Name>
        <v:Email>busaco@infoiasi.ro</v:Email>
      </rdf:Description>
    </s:Creator>
  </rdf:Description>
</rdf:RDF>
```

The next representation use attributes:

```
<rdf:RDF>
  <rdf:Description about="../movies/web.avi">
    <s:Creator rdf:resource="http://www.infoiasi.ro/~busaco"
      v:Name="Sabin Buraga"
      v:Email="busaco@infoiasi.ro" />
  </rdf:Description>
</rdf:RDF>
```

An RDF parser which implements the RDF specification make no difference between the above three forms, but not the same happens with the XQuery or XSLT querying tools. There where many attempts in order to normalize a RDF sequence into a different XML syntax which can be queried with these tools, but the RDF Core working group haven't adopted a standard solution yet. For example, Max Froumentin developed a XSLT RDF parser which first normalize a RDF sequence before querying it with XSLT or XQuery [Fro01].

We use below the solution gived by Jonathan Robie [Rob01]: in order to query an RDF document using XQuery, the RDF assertions are first converted to a normal form that eliminates the syntactic

and structural variation. In this normal form, every set of RDF assertions is presented as a list of statements, and each statement has one subject, one object, and one predicate. The previous three syntactic equivalent RDF sequences are, also, equivalent with the following RDF normal form:

```
<statement>
  <subject>../movies/web.avi </subject>
  <predicate>s:Creator</predicate>
  <object>http://www.infoiasi.ro/~busaco</object>
</statement>
<statement>
  <subject>http://www.infoiasi.ro/~busaco</subject>
  <predicate>v:Name</predicate>
  <object>v:Sabin Buraga</object>
</statement>
<statement>
  <subject>http://www.infoiasi.ro/~busaco</subject>
  <predicate>v:Email</predicate>
  <object>busaco@infoiasi.ro</object>
</statement>
```

The XQuery construct which generate this statements containing triples from the first syntactic form follows:

```
for $d in doc("books.rdf")/rdf:RDF/rdf:Description
  let $about = $d[@about]
  for $s in distinct-values ($d/*)
  returns statement
  {
    <statement>
      <subject>{$about}</subject>
      <predicate>{name($s)}</predicate>
      <object>{string($s)}</object>
    </statement>
  }
```

It can be observed that in the example below there are two statements relating to the same subject, <http://www.infoiasi.ro/~busaco>. In a such case the statements could be joined in a <rdf:Description> sequence containing the predicates as elements and the objects as content:

```
<rdf:description rdf:about=" http://www.infoiasi.ro/~busaco">
  <v:Name> v:Sabin Buraga </v:Name>
  < v:Email > busaco@infoiasi.ro</ v:Email>
</rdf:description>
```

This transformation could be obtained through the following XQuery construct, which unify the statements having the same subject:

```
<rdf>
  {
    for $t in distinct(document("books-triples.rdf")//statement/subject)
    return
      <rdf:description about={ $t/text() }>
      {
        for $s in document("books-triples.rdf")//statement
        where $s/subject=$t
```

```

return
  <{$s/predicate/text()}>
  {
    $s/object/text()
  }
  </>
}
</rdf:description>
}
</rdf>

```

It can be observed that the first syntactic RDF sequence, named books.rdf, corresponds to normalized RDF sequence having the statements optimized. So, for an homogeneous parse of the RDF metadata, we consider that it would be suitable to store metadata related to a subject inside an <rdf:description> element, having the subject as the value of “about” attribute. The name of sub-elements will be predicates for the subject, and their content – the corresponding objects.

Using XQuery language we could search, for example, all temporal multimedia objects belonging to a certain author. For this, it must be simultaneous queried the temporal.rdf and the books.rdf documents:

```

Define function retrieve-multimedia(charstring $author)
return rdf:Description
{ rdf:Description[@about=$author]
<rdf:Bag>
{for $d1 in doc(“books.rdf”)/rdf:RDF/rdf:Description
  where exists text(v:Name) = $author
  for $d2 in doc(“books.rdf”)/rdf:RDF/rdf:Description
    where exists $d2/s:Creator[@rdf:Resource]= $d1[@about]
    let $r=$d2[@about]
    for $t in doc(“temporal.rdf”)/rdf:Bag/rdf:Description
      if $t[@about] = $r
      then return
        { <rdf:li>$t</rdf:li>}
}
</rdf:Bag>

```

### *Implementation*

The XML processing techniques use both DOM (Document Object Model) [2, Xml-query] and SAX (Simple API for XML) [2] models, implemented in PHP. In order to process XQuery constructs, the *XQuery Lite* [7] PHP library is used. To view SMIL presentations, different players can be used. Our tests adopted GRiNS and RealOne players.

### **Conclusion**

In this paper, we presented first a short survey on XML-based query languages and their applications. The paper focused on XQuery language – proposed recommendation of the World-Wide Web Consortium.

One of the interesting problems regarding data retrieval is the multimedia retrieval. For this, the paper presented different techniques of generating SMIL presentations starting from existing (semi-) structured synchronized multimedia information. These techniques used XQuery constructs. Also, we

focused on different possible techniques for querying the multimedia information in order to generate RDF assertions that can be used to associate metadata for multimedia resources.

As a further work, we intend to investigate different approaches in order to automatically formulate XQuery constructs from a set of querying templates. Another direction is to study the possibilities of querying RDF sequences using specialized RDF-query languages (which are now in process of standardization), and the possibilities of combining these results with those obtained using XQuery language.

## References

- [Aro98] G. Arocena, A. Mendelzon, G. Mihaila, „Query Languages for the Web”, Query Languages W3C Conference, Boston, 1998: <http://www.w3.org/TandS/QL/QL98/wql.html>
- [Atw96] T. Atwood, D. Barry, J. Duhl et al., „The Object Database Standard: ODMG-93, Release 1.2”, R. G. C. Catell, Editor, Morgan Kaufmann Publishers, San Francisco, CA, 1996
- [14] J. L. Beckham, G. Fabbriozio, N. Klarlund, Towards SMIL as a Foundation for Multimodal, Multimedia Applications, W3C’s Multimodal Interaction Activity Group, 2002: <http://www.w3.org/2002/mmi/2002/smil-rexx.pdf>
- [Beg00] G. Begeed-Dov, D. Brickley et al., “Site Summary Specification Protocol (RSS 1.0)”, August 2000
- [Ber01] T. Berners-Lee, J. Hendler, O. Lassila “The Semantic Web”, Scientific American, May 2001
- [Bha00] K. Bharat, “SearchPad: Explicit Capture of Search Context to Support Web Search”, Proceedings of the Eight International World Wide Web Conference WWW9, ACM Press, 2000.
- [XQuery] S. Boag et al., “XQuery 1.0: An XML Query Language”, W3C Working Draft 12 November 2003: <http://www.w3.org/TR/xquery/>
- [15] J. Bowler, Scalable Vector Graphics (SVG) 1.0 Specification, W3C Recommendation, Boston, Sept. 2001: <http://www.w3.org/TR/SVG>
- [11] D. Brickley, R. Guha (eds.), *Resource Description Framework (RDF) Schema Specification*, W3C Candidate Recommendation, Boston. 2000: <http://www.w3.org/TR/rdf-schema>
- [Bur00] S. C. Buraga, T. Rusu, “An XML-based Query Language Used in Structural Search Activity on Web”, in The Fourth Conference on Technical Informatics – CONTI 2000 Proceedings, Timisoara, 2000.
- [2] Buraga, S. (2001) *Web Technologies*, Matrix Rom, Bucharest.
- [3] Buraga, S. (2002) *Modeling Relations Between Web Resources*, Transactions on Automatic Control and Computer Science, vol.47 (61), No.2, Politehnica Press, Timisoara.
- [4] Buraga, S., Brut, M. (2002) *Different XML-based Search Techniques on Web*, Transactions on Automatic Control and Computer Science, vol. 47 (61), No.2, Politehnica Press, Timisoara.
- [5] Buraga, S., Brut, M. (2001) *A Proposal for a Web Structural Search Language Based on XML Technologies*, Scientific Annals of the "A.I.Cuza" University of Iasi – Computer Science Section, Tome X, 2001, "A.I.Cuza" University Press House, Iasi.
- [Cer99] S. Ceri et al., “XML-GL: A Graphical Language for Querying and Restructuring XML Documents”, Proceedings of the Eight International World Wide Web Conference WWW8, Toronto, 1999.

- [Cha00] D. Chamberlin, "XQuery: An XML Query Language", IBM Systems Journal, Vol. 41, No. 4, 2002
- [17] W. Conen, R. Klapsing, *A Logical Interpretation of RDF*, Linköping Electronic Articles in Computer and Information Science, 5, 2000: <http://www.ida.liu.se/ext/epa/cis/2000/013/tcover.html>
- [Cha02] Guang-Ho Cha, "COVA: A System for Content-Based Distance Learning", in CDROM Proceedings of the 11<sup>th</sup> International World Wide Web Conference, Hawaii, USA, May 2002: <http://www2002.org/CDROM/alternate/683/>
- [Cla95] C. L. A. Clarke, G. V. Cormack, F. J. Burkowski, "An algebra for structured text search and a framework for its implementation", in The Computer Journal, 38(1), 1995;
- [Clu98] S. Cluet, C. Delobel, J. Siméon, K. Smaga, "Your Mediators Need Data Conversion!", in Proceedings of ACM-SIGMOD International Conference on Management of Data, pp. 177-188, 1998:  
<http://cosmos.inria.fr:8080/cgi-bin/publisverso?what=abstract&query=138>
- [Clu99] S. Cluet, S. Jacqmin, J. Siméon, "The New YATL: Design and Specifications", Technical Report, INRIA, 1999;
- [Deu99] A. Deutsch et al., "XML-QL: A Query Language for XML", in Proceedings of QL'98 – The Query Languages Workshop, Cambridge, Mass., 1998: <http://www.w3.org/TR/1998/NOTE-xml-ql-19980819/>; also, in Proceedings of the International World Wide Web Conference, 1999: <http://www.research.att.com/~mff/files/final.html>
- [Fer99] M. Fernandez, J. Siméon, P. Wadler, "XML Query Language: Experiences and Exemplars": <http://www-db.research.bell-labs.com/user/simeon/xquery.html>
- [Flo98] D. Florescu, A. Levy, A. Mendelzon, "Database Techniques for the WWW: a Survey", SIGMOD Record, ACM, 1998.
- [Fro01] M. Froumentin, <http://www.idealliance.org/papers/xml2001/papers/html/03-01-04.html>  
<http://www.w3.org/2001/12/rubyrdf/xsltrdf/README.html>
- [Gol99] R. Goldman, J. McHugh, J. Widom, "From semistructured data to XML: Migrating the Lore data model and query language", in Proceedings of the 2nd International Workshop on the Web and Databases (WebDB '99), Philadelphia, Pennsylvania, June 1999;
- [18] P. Hayes (ed.), *RDF Model Theory*, W3C Working Draft, Boston, 2002:  
<http://www.w3.org/TR/rdf-mt/>
- [Ish98] H. Ishikawa, K. Kubota, Y. Kanemasa, "XQL: A Query Language for XML Data", Query Languages W3C Conference, Boston, 1998: <http://www.w3.org/TandS/QL/QL98/flab.txt>
- [Kar02] G. Karvounarakis, S. Alexaki et al., "RQL: A Declarative Query Language for RDF", W3C Conference, Hawaii, SUA, 2002: <http://www2002.org/CDROM/refereed/329/>
- [10] O. Lassila, R. Swick (eds.), *Resource Description Framework (RDF) Model and Syntax Specification*, W3C Recommendation, Boston, 1999: <http://www.w3.org/TR/REC-rdf-syntax>
- [Mil03] L. Miller, "Databases, Query, API, Interfaces report on Query languages", Public report for Semantic Web Advanced Development for Europe (SWAD-Europe) Project, 2003:  
[http://www.w3.org/2001/sw/Europe/reports/rdf\\_ql\\_comparison\\_report](http://www.w3.org/2001/sw/Europe/reports/rdf_ql_comparison_report)

- [12] J. van Ossenbruggen, L. Rutledge, et al., Synchronized Multimedia Integration Language (SMIL 2.0) Specification, Aug. 2001: <http://www.w3.org/TR/smil20/>
- [13] S. Pemberton et al., XHTML 1.0 - The Extensible HyperText Markup Language, W3C Recommendation, Boston, Jan-Aug 2002: <http://www.w3.org/TR/xhtml1/>
- [Pru03] E. Prud'hommeaux, B. Grosz, „RDF Query and Rules: A Framework and Survey”, W3C overview, 2003: <http://www.w3.org/2001/11/13-RDF-Query-Rules/>
- [Rob98] J. Robie, J. Lapp, D. Schach, “XML Query Language (XQL)”, in Proceedings of QL'98 – The Query Languages Workshop, Cambridge, Mass., 1998: <http://www.w3.org/TandS/QL/QL98/pp/xql.html>
- [Rob01] Jonathan Robie, “The Syntactic Web. Syntax and Semantics on the Web”, Proceedings of the “XML Conference & Exposition 2001”, Orlando, Florida, December 2001: <http://www.idealliance.org/papers/xml2001/papers/html/03-01-04.html>
- [Xq-dm] M. Fernandez, J. Robie, „XML Query Data Model”, W3C Working Draft 15 February 2001: <http://www.w3.org/TR/2001/WD-query-datamodel-20010215/>  
 ”XQuery 1.0 and XPath 2.0 Data Model”, W3C Working Draft, 16 August 2002: <http://www.w3.org/TR/querydatamodel>
- [Xq-sem] „XQuery 1.0 Formal Semantics”, W3C Working Draft, 16 August 2002: <http://www.w3.org/TR/query-semantics>.
- [Xq-op] „XQuery 1.0 and XPath 2.0 Functions and Operators”, W3C Working Draft, 16 August 2002: <http://www.w3.org/TR/xquery-operators>
- [Xsl] <http://www.w3.org/TR/WD-xsl>
- [XPath] <http://www.w3.org/TR/XPath>
- [Xschema] <http://www.w3.org/TR/XSchema>
- [KPex] \* \* \*, <http://www.dmoz.org/>, <http://home.cnet.com/>, <http://www.xmltree.com/>
- [7] \* \* \* (2003) *XQuery Lite*: <http://sourceforge.net/projects/phpxmlclasses/>
- [16] \* \* \*, W3C, Synchronized Multimedia: <http://www.w3.org/AudioVideo/>
- [Xml-query] \* \* \* (2003) *World-Wide Web Consortium's Activity on Query Languages*:
- [W3C] \* \* \* (2003) *World-Wide Web Consortium's Technical Reports*: <http://www.w3.org/TR/>  
<http://www.w3.org/XML/Query>