XML QUERY OBJECT MODEL: A GENERAL VIEW

Fábio Ribeiro Silvestre
Fundação Visconde de Cairu - CEPPEV
Rua do Salete 50, Barris - Salvador/BA - Brasil

Hernane Borges de Barros Pereira
Fundação Visconde de Cairu - CEPPEV and
Universidade Estadual de Feira de Santana - DEXA
Rua da Brisa 42, Ap. 1701 Pituba - Salvador/BA - Brasil

Keywords: Querying Process, Semistructured Data, XQOM Schema, Framework Technology.

Abstract: We have observed an increase in document search needs in organizations. This increase in document search demand has required better efficiency on search software languages. The current technologies are using specific patterns to search documents. The most usual pattern, called XML, has been proposed by W3C and some languages (e.g. XQL, XML-QL etc.) were created to recover XML documents. The implementation of a XML query server, which abstracts the language used, has been an emerging necessity. This paper surveys technology used to develop a XML query server, called XQOM Schema, from the modelling to the implementation. In an organization setting, we have adopted new abstraction forms of searching for XML documents without the usage limitation established by a specific language. XQOM Schema may contribute to the creation of an universal document searching pattern in XML format.

1 INTRODUCTION

Currently, the demand for documents search systems has increased in organizations. The Internet's growth brings a some practical issues. For instance, how to recover documents more efficiently? However, there are some technologies to solve this problem that use specific patterns to search documents, such as XML (eXtensible Markup Language), proposed by W3C (Consortium World Wide Web) (Bray et al., 1997).

Some XML query languages have been proposed to W3C. Three of them are more usual: XQL (functional pattern), XML-QL (relational pattern) and Lorel (object oriented pattern). We propose a query server to semi-structured documents based on XML language, which abstracts the query language employed. We have chosen the XQL and XML-QL languages to test the proposed server. These languages use a driver developed by researchers at the Berkley University for XQL and a driver developed by researchers at the AT&T for XML-QL. (Robie et al., 1998; Research, 1999).

Despite the fact that investments (from the time and money perspectives) in researches on query languages are being carried out, there is a lack of products and processes related to the information and management knowledge, inside the organization. In general, organizations have a well-defined and almost inflexible way for internal data structure. This, intuitively, implies the use of conventional databases (e.g. relational or object-oriented databases). However, the processing of semi-structured data in these databases types is inefficient since semi-structured data has a predefined framework of data storing. On the other hand, the semi-structured documents have a highly flexible and irregular structure that makes the conventional databases inappropriate, disabling the databases with respect to the semi-structured data storing (Graves, 2001).

The main goal of this paper is to survey the XML Query Object Model that consists of a server (framework), called XQOM Schema, that allows the creation of several XML document search applications used in an organizational context (e.g. to be worked in an intranet). This server uses metadata to carry out searches in electronic documents, aiming more precise results.

2 XML QUERY OBJECT MODEL

XML Query Object Model (XQOM) is a query server to semi-structured data. Some function paradigms must be observed by applications that aspire to use the

Ribeiro Silvestre F. and Borges de Barros Pereira H. (2006).
XML QUERY OBJECT MODEL: A GENERAL VIEW.
In *Proceedings of WEBIST 2006 - Second International Conference on Web Information Systems and Technologies - Internet Technology / Web Interface and Applications*, pages 469-472
DOI: 10.5220/0001244904690472

XQOM Schema. Two steps are essential. First step determines that there will be an individual responsible for the creation of conceptual documents in a specific category. The second one implies that there will be a mapping between conceptual model and stored XML documents. These XML documents have been created through XQOM Schema.

XQOM Schema is basically composed of two modules: administration module and query module. Thus, there will be corresponding profiles: administrator and end-user.

2.1 XQOM Architecture

In this section, we present the XQOM Schema functional processes. Figure 1 depicts the XQOM architecture and its functionality is detailed as follows.

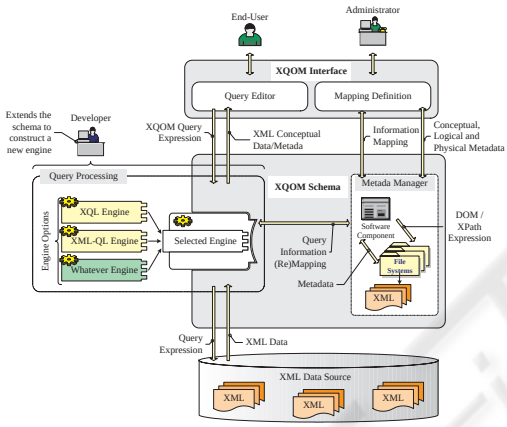

Figure 1: XQOM Architecture.

2.1.1 XQOM Administration Module

The administrator defines conceptual, logical and physical worlds, explained below, which must be disposed to query through the XQOM Schema. These worlds are controlled by XQOM mapping process. The administrator also defines the mapping between conceptual world and several documents (i.e. physical documents). All documents involved in the XQOM mapping process are stored in a local file system to future searches. Mapping module is performed by a XQOM functionality, called XQOM Map Engine, that is essential for the (re)mapping of XQOM metadata in both administration and query modules.

We understand conceptual world as a common set of definitions or nomenclatures created by administrator. This set evolves specific domains of rules in an environment and it is collectively created. Conceptual models are just XML documents which represent these domains through its hierarchical structure.

Physical world denotes documents created by some people (e.g. paper, journal, magazine, book etc.). The end-user must structure his/her documents taking into account XML markups. After all, the document must be sent to the administrator to storage in XQOM Schema.

Logical world evolves mapping between conceptual document and physical document(s). The administrator (i.e. a specialist in that information area) must associate conceptual world metadata with physical world metadata. We emphasize that physical documents are created by different users and, thus, they can represent them with different XML markups. This occurs because users are completely free to invent their own markup nomenclatures.

The administrator's role is fundamental to promote a good operation of XQOM Schema. It is extremely important that this individual be a specialist in the area being mapped.

Metadata management is responsible for the control of administration module. Its goal is to provide mapping information as well to generate conceptual, logical and physical documents, storing them in the local file system. The physical document storage is defined by the application that extends XQOM Schema. The same process is valuable to search models or physical documents.

When a solicitation of mapping information is sent to metadata manager process, a software component is activated to attend this request. This software component constructs Xpath/DOM expressions that will be applied to XML metadata documents distributed among the directories of the local file system. The results of mapping metadata are returned and redirected to the metadata manager that sends them to the application solicitant.

Using the mapping information as a starting point, the administrator can create new models or update them. These models are redirected to the XQOM Schema by the administrator. Then metadata manager process updates the conceptual and logical documents in directories of file system.

2.1.2 XQOM Query Module

The end-user can execute searches to conceptual documents using XQOM Query Module. He or she performs the queries through the XQOM objects specification process. The language generation process (i.e. XQOM Interpreter) uses it to generate new query languages. The XQOM mapping process is used by XQOM Interpreter to generate XML query languages to document of the "physical world". These languages are submitted to the XML, which returns a response. Each response is mapped again to the end-user "conceptual world".

XQOM Specification is another functionality of

XQOM Schema. It is also represented through XQOM query objects that constitute the status of search syntaxes to semi-structured data. The grammatical representation of the search, performed by the end-user through the conceptual models, is stored in XQOM object specification. These objects represent a query grammar to semi-structured data.

XQOM Map Engine is responsible by mapping the query objects of XQOM Specification to the corresponding XQOM query objects that represent the physical document markups. XQOM Interpreter will perform queries using these objects and it will return a XML document. This will be transferred to XQOM Map Engine to map it in corresponding conceptual model to end-user.

XQOM Interpreter is responsible for the query language generation through the query objects transferred from the XQOM mapping process. It transforms these objects in a XML query language. From this point, the process continues as discussed before.

We have observed that XQOM Map Engine and XQOM Interpreter are the XQOM Schema functionalities that it may be activated several times in a query. This is possible because conceptual model can be associated to more than one physical document through logical document.

The end-user is responsible for execution of the query module. He or she accesses XQOM Schema through the client-server application or the web application. The query machine is the process responsible for the query module management.

The end-user constructs XQOM expressions using a query editor (i.e. XQOM Interface). The query machine process is activated to attend the request and it calls the metadata manager to perform the XQOM expression mapping.

In this process, n XQOM query objects are transferred to the application that extends XQOM Schema, which will generate n XML query languages. These will be transferred to a class responsible for the linking of the query language to a XML query processing software or to a XML database.

The resultant XML documents are returned to the query machine, which will require the metadata manager a new mapping of XML physical documents in the corresponding conceptual model. This model is returned to query machine and, after all, to the initial application. The end-user only will see a conceptual result.

2.2 XQOM Query Objects

XQOM query objects are represented by grammatical graph objects in which its status denotes syntaxes of a query language to semi-structured data (Figure 2).

XQOM Schema has been inspired on DOM model (Document Object Model), but XML documents are not converted into XML objects defined by DOM specification. The objects in XQOM Schema have values defined by end-user and compiles them to a specific XML query language.

In case of change in the persistence layer, it is only necessary a driver change to generate a new query language through the XQOM Schema. This driver is defined through a class that compiles the objects of the XQOM Schema to the corresponding database language. The XQOM Schema implicitly calls this class and its change will not affect the application that uses the proposed schema.

Figure 2 represents the XML query object model through the oriented and labeled graph. This representation is based on the deterministic finite automata (Aho et al., 1974; Aho et al., 1995; Du and KO, 2001; Hopcroft et al., 2001).

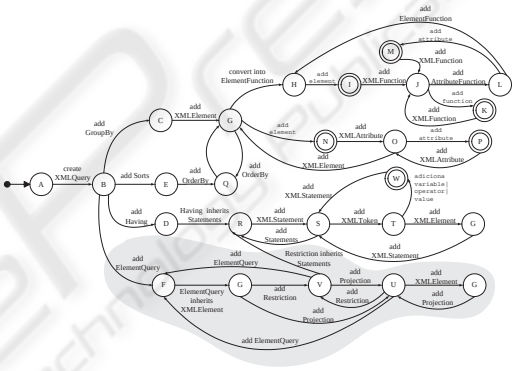

Figure 2: XQOM Query Objects used to test the XQOM Schema.

The gray area of Figure 2 represents the query objects used to test the query languages within the XQOM Schema.

2.3 XQOM Algorithm

The following XQOM Algorithm describes the search carried out through the XQOM Schema from the conceptual model (or document) input and its query filters to the result based on XML document that also consists of a conceptual document:

1. Define the structure of the conceptual model;
2. Establish the structure of the physical models;
3. Define the mapping between the conceptual and physical models;
4. Load the filters;
5. Configure the lexical analyzer based on the filters sent by XQOM Interface;
6. While there exists physical model associated to logical model do:
   (a) Run the lexical analyzer based on the filters sent by XQOM Interface;
   (b) Run the query languages and store the XML results;

7. Remap to only one conceptual model the XML results based on the definition of the logical models.

3 SIMULATION AND RESULTS

In this section, we present simulations' performances that have collaborated to validate the XQOM Schema functioning process. These results have partially been obtained through a non-visual query environment for the queries homologation carried out in the XQOM Schema.

The communication among the conceptual, logical and physical models is composed of four layers (i.e. interface, conceptual metadata, logical metadata and physical metadata), which illustrates the XQOM mapping idea (Silvestre et al., 2005) .

The XQOM mapping process, depicted in Figure 3, is basically described as an association between conceptual metadata and physical metadata (i.e. XML Files).

Figure 3 presents, in a general view, an instance of the XQOM mapping used to simulate and evaluate the XQOM Schema: three physical documents and a conceptual model. They are create and stored by the administrator. The associations between conceptual and physical models, presented in Figure 3, represent the logical model.

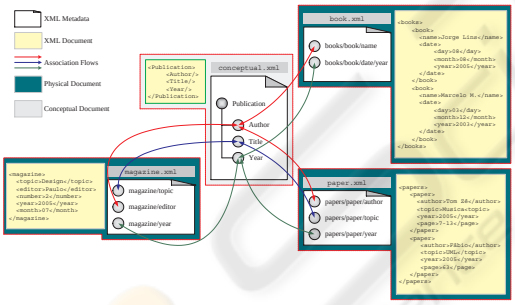

Figure 3: XQOM Mapping: Association between conceptual metadata and physical metadata (i.e. XML files).

4 CONCLUDING REMARKS

Some benefits from XQOM Schema are pointed out. It will facilitate to query official documents, journals, libraries, magazines etc. It will also contribute to the growing up of Document Electronic Management applications used in organizations that needs to offer their documental heritage for searching. These benefits can be possible through the use of conceptual models such as ontologies, which will facilitate the search and retrieval of electronic documents.

As well, it is important to highlight that XQOM Schema does not use any particular XML query language, despite it can generates any language. This fact turns it portable to other databases. Its portability is granted among operational systems because it has been developed in Java.

We point out a XQOM Schema limitation. There is a great responsibility granted to the administrator and end-user as well. They create the markups for the conceptual documents and physical documents, respectively. This can generate incorrect markups or physical documents. As a consequence, wrong results can be returned. Finally, it is suggested as a future work the realization of an interface evaluation in order to improve the usability of the XQOM Interface.

REFERENCES

Aho, A. V., Hopcroft, J. E., and Ullman, J. D. (1974). *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading.

Aho, A. V., Sethi, R., and Ullman, J. D. (1995). *Compiladores Princípios, Técnicas e Ferramentas*. LTC, Rio de Janeiro.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., and Yergeau, F. (1997). Extensible Markup Language (XML). on-line. XML Specification by W3C.

Du, D.-Z. and KO, K.-I. (2001). *Problem Solving in Automata, Languages, and Complexity*. John Wiley and Sons, New York.

Graves, M. (2001). *Designing XML Databases*. Prentice Hall.

Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2001). *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, 2nd edition.

Research, A. L. (1999). XML-QL: A Query Language for XML. on-line. Source AT&T for the XML-QL.

Robie, J., Lapp, J., and Schach, D. (1998). XML Query Language (XQL). on-line. XQL Tutorial of the W3C.

Silvestre, F. R., Pereira, H. B. B., and Jorge, E. M. F. (2005). Esquema XQOM de Consultas a Documentos XML: Análise Teórica. In *SUCESU'2005 - Congresso Nacional de Tecnologia de Informação e Comunicação*, Belo Horizonte, Brazil.