Managing Branch Versioning in Versioned/Temporal XML Documents*

Luis J. Arévalo Rosado, Antonio Polo Márquez, and Jorge Martínez Gil

University of Extremadura, Department of Computer Science
Avda. de la Universidad s/n 10071 Caceres (Spain)
{ljarevalo,polo,jmargil}@unex.es

Abstract. Due to the linear nature of time, XML timestamped solutions for the management of XML versions have difficulty in supporting non-lineal versioning. Following up on our previous work, which dealt with a new technique for the management of non-lineal versions of XML graph documents, called versionstamp, we have gone a step forward by adding temporal information to each version included in the document. Not only does it allow us to query the vDocuments on a temporal and version level but also we can manage branch versioning in the temporal axis. Moreover, to check its functionality, we have compared our technique to a timestamped XML solution and a set of Web services has been developed. The easy management of multiple versioning, the large number of queries in different XML standard query languages and its implementation by using only XML technology, are some of the advantages of the proposed technique.

1 Introduction

In this collaborative society information flows through all forms of computing, however nobody looks at it in a static way because it changes throughout time and its management becomes necessary to query past information, to retrieve documents belonging to a specific version and to monitor the changes, etc. Document management has been used for years in such environments like collaborative software development, file share resources, etc and more recently, with the appearance of XML [1], it has become necessary also to manage these documents.

Versions of an XML document can be managed through traditional procedures like CVS [2] or subversion [3], the traditional adapted procedures based on XML operations change (delta XML) [4,5] or integrate the different versions into a single XML file using temporal [8,11,12,13,14] or version [9,15] technique. We consider that whatever XML versioning system should have the following main features: it should be able to, validate all XML versions of the document to its schema (the first two solutions do not take into account this fact), support branch versioning (temporal solutions do not do this) and, have the possibility to

* This work has been financed by Spanish CICYT projects "TIN2005-09098-C05-05" and "TIN2005-25882-E".

D. Barbosa et al. (Eds.): XSym 2007, LNCS 4704, pp. 107–121, 2007.

query the XML versioned documents using some XML standard query languages such as XQuery and XPath (the first solution does not do this).

To get these characteristics, we have used the technique shown in [9] that consists of marking the document with a versionstamp instead of using a timestamp. In this work we have gone even further by adding temporal information to each version allowing us to query the document either on temporal or/and version level. We have also defined the basic updated operations common to whatever XML document, describing them by means of an XML document called *XML transactional document* which allows us to manage changes for any markup language based on the XML specification. Moreover, to check its functionality, we have compared our technique to a timestamped XML solution as well as developing a set of Web Services.

The remainder of this paper is organized as follows: we begin by summarizing the current solutions for the management of XML versions. Then, we continue showing the foundations of this paper based on [9], extending it with temporal information and describing later the basic updated operations. We then follow up this by showing several queries made on a temporal and version level. After that, some implementation details and the achieved results are discussed and finally, we offer our conclusions and a look at our future work.

2 State-of-the-Art

The problem of XML document version management combines the issues of document version management [4,5,6,7] and temporal databases [22]. Document version management has been used for years mainly in collaborative environments. These traditional techniques [2,3] are based on diff lined-based algorithms to locate the differences between two versions of a text. For XML documents, where the organization in lines can be neglected, line-based approaches are inappropriate since the structure of the document is lost. The necessity to manage XML versions not only is important in XML databases but also in XML document management because nowadays more and more applications use it to store their configurations, data, etc, such as OpenOffice and Microsoft Office.

XML solutions have been centered mainly in some of the following ideas. *Delta XML management* is based on traditional change operation procedures adapted to XML [4,5]. It consists of obtaining and storing the XML differences between two versions (*delta XML*). An exhaustive study of XML diff approaches is made in [10] where the authors use an C++ implementation of [4] to manage XML OpenOffice document versions. However delta XML solutions have the same problems than traditional techniques, it means, neither XML validation nor XML query cannot be carried out in these solutions.

Multiversion XML [6,7] define an indexing technique for branched versioning which they called BT-Tree and BT-ElementList respectively, however they cannot be used in XML Standard Query languages (XQuery or XPath[2]).

Temporal XML Representation based on temporal database topics [21] representing and managing historical information in XML. In [11] a technique for