

# Technologies for a CERIF XML based CRIS

*Stefan Bärtsch*

GESIS-IZ, Bonn, Germany

## Abstract

The use of XML as a primary storage format as opposed to data exchange raises a number of questions regarding the technological feasibility for such an approach. Chief them are (1) storage of XML in the transactional environment of a CRIS and (2) the use of XML in a CRIS, especially the relationship between XML documents and the object model of a programming language. We present current approaches and technologies that can be applied to these fields and discuss their respective advantages and disadvantages. We conclude that while both native XML databases and Hybrid database support the use cases for a CRIS, native XML database are less complicated in use and deployment.

## 1 Introduction

The use of XML as a data exchange format is well established in practice, but utilization of large volumes of XML as the native data format is less common, especially in application domains that include transaction handling. To be employed in a transactional system such as a CRIS, an XML storage system must not only retrieve datasets from XML but must also be able to manipulate information while conforming to ACID (Codd, 1970) criteria. The challenges that XML faces stem both from the tree oriented structure of XML documents and from the current state of the technology.

### 1.1 Advantages of XML for a CRIS

Given the long history of the relational data model (Codd, 1970) and their nearly ubiquitous nature of RDBMS in data management today, the question has to be posed why a CRIS should use XML as its native data format. It is indeed the case that RDBMS are today proven technology. The systems and the relevant support technology, such as data modeling tools, are readily available both from commercial vendors and from the open source community. There are however application domains for which semi-structured data models such as XML are better suited than the relational model (Abiteboul et al. 2000). XML is better suited for evolving schemas according to the requirements of an application domain, XML is also well suited to document-centric (Schmidt et al., 2002) application domain. In addition, the close structural match between the XML data model and the object graphs that represent that data in a program avoids the object-relational gap and therefore the need for object relational mappers and the associated complexities. Given that XML has these theoretical advantages over the relational model in theory, we must investigate whether they can be carried forward into the practice and help to improve the development process and consequently the quality of a CRIS. We argue that this is the case with the advent of modern stor-

age system for XML such as native and hybrid XML databases and is practical given the performance increases both in computer hardware and XML tooling in recent years.

## 1.2 XML compared to the relational data model

XML has a number of different data models<sup>1,2</sup> that differ mostly in the types of nodes that are distinguished. In the practical application of the XML as a data format in applications like a CRIS however, the differences between the different data models will not be significant for most usage scenarios. The differences between the set-oriented relational data model and the tree structure that underlies XML can have significant consequences on the design and implementation of an application. In transactional contexts with a high degree of concurrency, the relational models in theory allow for more efficient locking mechanisms. The degree to which such theoretical differences manifest themselves in actual applications depends both on the underlying storage technology and the representation of the relevant entities.

## 1.3 XML as the native data format in a CRIS

Concerning the use of CERIF XML (Jörg, 2007) if the use of current XML storage technology (see section 2) is assumed, we argue that the use of CERIF XML in an transactional application such as a CRIS is feasible for the following reasons:

- CERIF XML is a data model without deep hierarchical structures, relationships between different entities are expressed by references, not by inclusion. This structure allows the fine-grained manipulation of data.
- Highly selective indexes are supported by CERIF XML for entities and attributes. Both entities and attributes from the relational data models are expressed as elements. Attributes that uniquely identify entities retain information about the type of the entity; cfPersID for example identifies the ID of a person entity. This structure allows retrieving and manipulating entities by indexes on the element name. Such indices are efficient and widely supported in XML storage systems. The positive aspects of the CERIF XML data models for transactional application are in part a result of the fact that the underlying CERIF data model is a relational model for transactional systems. This gives rise to the question how the extension mechanism in CERIF XML affects the feasibility of CERIF.

We would argue that be their very nature the extension mechanism will not negatively affect efficiency. Since extensions mostly add information not directly related to the core workflows of a CRIS, it is unlikely that content added by the extension mechanism would be of any relevant consequence to the system as a whole.

## 2 XML data storage technology

One prerequisite for use of XML as the native data model for a CRIS is the availability of a data management system that supports efficient retrieval as well as insertion in transactional contexts. A technology used should be standardized and be supported by multiple implementations that satisfy, for example, different requirements regarding costs and performance. In order to allow software developers working on a CRIS to focus on the

1 <http://www.w3.org/TR/xpath>

2 <http://www.w3.org/TR/xpath-datamodel>

applications and not on the details of the storage system, this technology should also be easy to use with XML.

## 2.1 Relational Databases

Relational database management systems offer different levels of support for XML data. On the most basic level, XML documents are simply treated as text. Some systems that follow this approach can check whether documents are well-formed and some offer SQL extension functions to query documents using XPath expressions; examples for such systems would be Sun MySQL 5.1 and PostgreSQL 8.3. When such support for XML is part of a RDBMS, an important differentiator lies in the storage architecture that provides the support. If the text of an XML document is parsed each time the document is accessed, performance characteristic for the system will be comparable to retrieving the whole document from the database and parse it in the client. Another approach to store XML in a relational database without the need to parse documents on each request is Shredding (Funderburk et al. 2002). When Shredding is used, a document is parsed and then mapped to the tables and attributes of an relational schema. Several methods of shredding exist, most of which rely on the existence of a mapping between XML documents and their representation in the relation schema. Since the use of such a mapping limits the RDBMS to XML documents for which a mapping exists, and since shredding has a negative impact on insertion performance, shredding is not an optimal solution for XML storage.

## 2.2 Hybrid Databases

Hybrid databases (Beyer et al. 2006; Nicola & van der Linden, 2005) support both the relational and the XML data model. Both XQuery and SQL are offered as query languages. Compared to the handling of XML data in relational databases as described above, using a native storage mechanism for XML data in relational results in better performance since reparsing or shredding XML is not necessary.

From a technical perspective, Hybrid databases are best suited for usage scenarios where relational data and XML are used in combination, for example when XML is added to an existing system based on relational data. If a system does not need to support legacy data from relational databases, the advantage that Hybrid databases have over native XML databases in handling different data models is not relevant.

## 2.3 Native XML Databases

For the context of this work, we regard a data management system as a native XML database when the interfaces exposed by the system are primarily XML-oriented, that is when XML is used as the logical data model and when the internal storage model of the database management system is optimized for XML (Fiebig et al., 2002; Meier 2002). One advantage of such systems when compared to Hybrid databases is often their relative simplicity. Since native XML database management systems are focused only on the XML data-model, both management and use of such systems requires fewer resources and skills than the use of hybrid systems. Another advantage of native XML databases is the availability of systems with good standard compliance and performance characteristics both as proprietary software and in the form of open source project. One problematic aspect in the field of native

XML Databases is the current lack of implementation for a standardized update mechanism. Currently, support for the update part of the XQuery Standard<sup>3</sup> can not be assumed for all systems, even those that fully support the retrieval aspects. We argue that this situation, while clearly not optimal, does not limit the uses of native XML databases for a CRIS using CERIF XML. The fine granularity of entities in CERIF XML allows the implementation of updates by deleting and reinserting documents without prohibitive impacts on performance. Since document manipulation using this approach can be mostly decoupled from the interfaces provided by the XML databases, only update and delete operations for documents would have to be changed when adapting a CRIS for different native XML databases. While this is clearly not an optimal solution, it does not hinder the development of a hypothetical CERIF XML CRIS.

### 3 XML Handling

As the technology to implement a CERIF XML based CRIS exists both in the form of native XML database management systems and in Hybrid databases, we now discuss the feasibility of related technology such as APIs, data retrieval, validation and transformation.

#### XML Object Models

Depending on the programming language used, multiple APIs are available for the manipulation of XML. Since SAX and pull parsers are best suited for large data volume and streaming scenarios, these will not be discussed. Manipulating XML using an API based on the Document Object Model or a related standard can be a suitable approach to create and manipulating and navigate fragments of XML. However, for use with CERIF XML Entities, data bindings such as JAXB<sup>4</sup> or JiBX<sup>5</sup> offer better separation of concerns for marshalling and demarshalling. The API offered for programmatic XML manipulation is different for the XML database management systems currently on the market. Some systems implement a DOM<sup>6</sup> API over all elements in the database, thus making it possible to handle data in the database in the same way as a XML document in memory. For CERIF XML, we do not consider such functionality necessary since individual CERIF XML entities are rather compact.

#### 3.1 XML Query Languages

When querying XML, XQuery and XPath are the most relevant languages. Even though XPath is a sublanguage of XQuery, both languages can be considered in separation since XPath alone supports many use cases for data retrieval from XML. XPath is well suited to retrieve single elements from a number of XML documents. The language is well supported by tools such as XML Editors, libraries with at least XPath 1.0 support are available for the majority of programming languages and programmers with experience in XML are likely to know XPath 1.0. XQuery is a more complex language than XPath. Experience with the language in software developers can not be assumed. Full support for the 1.0 standard in tools or database management systems is also not always available. However, XQuery has the ability not only to express more complex queries to data but is also capable of data transfor-

3 <http://www.w3.org/TR/xquery-update-10/>

4 <https://jaxb.dev.java.net/>

5 <http://jibx.sourceforge.net/>

6 <http://www.w3.org/DOM/Activity>

mations. For a CRIS based on CERIF XML, XQuery and therefore XPATH offer a standardized and powerful method of data retrieval. One potential problem in using XQuery on large volumes of data is the relative infancy of XML query optimization as compared to relational database management systems. This problem is not likely to manifest with CERIF XML, due to the aforementioned element structure that supports highly selective indices.

### 3.3 XML Data Transformation

The primary means to transform XML, either into another XML format or into another data model are XSLT and XQuery. Since XSLT 2.0 and XQuery are based on the same XML data model and have the same power of expression for transforming XML documents, both languages are suitable as data transformation languages in a CRIS. In the experience of the authors, XSLT is better suited to transform XML into formats for display, for example (X)HTML pages, while XQuery works better in roles like data aggregation or report creation.

### 3.4 XML Data Validation

Data quality is one of the most important requirements for a CRIS. For a CRIS using XML as its internal data format, schema validation is among the most common methods to ensure data quality. Schema languages such as XML Schema, RelaxNG and to some degree DTDs allow the validation of structural constraints of a document. In addition to such languages, Schematron, a XSLT based schema language, makes it possible to write assertions that evaluate nonstructural properties of documents, such as the condition that the sum of a number of integer attributes must always equal 100. All schema validation approaches discussed above have weaknesses in the validation of relationships between different documents. Such validation can be done in either XSL Transformations or XQuery by using the document function. This method of validation however would have to be developed specifically and it is unclear what performance impact such a validation would have for a production system of non-trivial size.

## 4 Conclusion

XML and CERIF XML in particular is suited for use as the native data format for a CRIS. The current technology for XML data storage and the APIs and languages available to query, transform and validate XML data support this application domain. As with all new technologies in software development, one limiting factor for applying this technology in practice is the experience of the participants. We would therefore argue for the adoption of the more usable native XML databases over Hybrid databases and for using in incremental software development methodology in order to identify risks early in the software development life cycle.

## 5 References

- Abiteboul, S.; Buneman, P.; and Suciu, D. (2000). *Data on the Web: From Relations to Semi-structured Data and XML*. Morgan Kaufmann.

- Beyer, K.; Cochrane, R.; Hvizdos, M.; Josifovski, V.; Kleewein, J.; Lapis, G.; Lohman, G; Lyle, R; Nicola, M.; Özcan F., et al (2006). DB2 goes hybrid: Integrating native XML and XQuery with relational data and SQL. IBM Systems Journal, 45 (2): 271–298.
- EF Codd, (1970). A relational model of data for large shared data banks. Communications of the ACM, 13 (6): 377–387.
- Fiebig, T.; Helmer, S.; Kanne, C.; Moerkotte, G.; Neumann, J.; Schiele, R.; Westmann, T. (2002). Anatomy of a native XML base management system. The VLDB Journal The International Journal on Very Large Data Bases, 11 (4): 292–314.
- Funderburk, J.; Kiernan, G.; Shanmugasundaram, J.; Shekita, E.; and Wei, C. (2002) XTABLES: Bridging relational technology and XML. IBM Systems Journal, 41 (4): 616–641.
- Gray, J and Reuter, A. (1993). Transaction Processing: Concepts and Techniques. Morgan Kaufmann.
- Jörg, B.; Krast, O.; Jeffery, K.; van Grootel, G. (2007). CERIF2006 XML-1.1 o Data Exchange Format Specification. Technical report, Eurocris.
- Meier, W. (2003). eXist: An Open Source Native XML Database. Web, WebServices, and Database Systems: Node 2002 Web- And Database-Related Workshops, Erfurt, Germany, October 7-10,
- Nicola, M.; van der Linden, B. (2005). Native XML support in DB2 universal database. Proceedings of the 31st international conference on Very large data bases, pages 1164–1174.
- Schmidt, A; Waas, F; Kersten, M; MCarey, J; Manolescu, I; Busse, R;.(2002) XMark: a benchmark for XML data management. Proceedings of the 28th international conference on Very Large Data Bases-Volume 28, pages 974–985.

## Contact Information

*Stefan Baerisch*  
Lennéstr. 30  
53113 Bonn  
Germany  
stefan.baerisch@gesis.org  
<http://www.gesis.org/staff/sbaerisch>