Multi-user Evaluation of XML Data Management Systems with XMach-1

Timo Böhme and Erhard Rahm

University of Leipzig, Germany
{boehme,rahm}@informatik.uni-leipzig.de
http://dbs.uni-leipzig.de

Abstract. XMach-1 was the first XML data management benchmark designed for general applicability [1]. It is still the only benchmark supporting a multi-user performance evaluation of XML database systems. After a brief review of XMach-1 we summarize three additionally proposed benchmarks (XMark, XOO7, Mbench) and provide a comparison between these benchmarks. We then present experiences and performance results from evaluating XML database systems with XMach-1.

1 Introduction

Current XML database systems exhibit considerable differences in their architectural foundations, concepts and functionality. When choosing a system for a specific application scenario these aspects and the resulting performance should be taken into consideration. The intention of XML database benchmarks is the comprehensive and realistic evaluation of XML database systems in order to allow for a performance comparison of them.

Several papers about storing XML in databases contain performance measurements based on self defined benchmarks [2, 3]. These studies commonly lack a detailed benchmark description, have only a few very specific operations tailored to the corresponding subject of the paper and are therefore not suitable for a general comparison.

The growing demand for well defined XML data management benchmarks led to the specification of various benchmarks in the last two years. The applicability of a benchmark depends on how close the benchmark simulates the application domain in question. To find the most appropriate benchmark it is necessary to carefully study the specifications of each of them. We therefore provide a comparison between four benchmarks highlighting their key features and distinctive features.

The rest of this paper is organized as follows. In the next section, we give an overview of the first general XML database benchmark, XMach-1. Section 3 briefly introduces three further XML database benchmarks: XMark, XOO7 and Mbench. We then compare the four benchmarks with respect to key features. In Section 5, we present experiences and results from evaluating XML database systems with XMach-1.

S. Bressan et al. (Eds.): EEXTT and DIWeb 2002, LNCS 2590, pp. 148-159, 2003.
© Springer-Verlag Berlin Heidelberg 2003

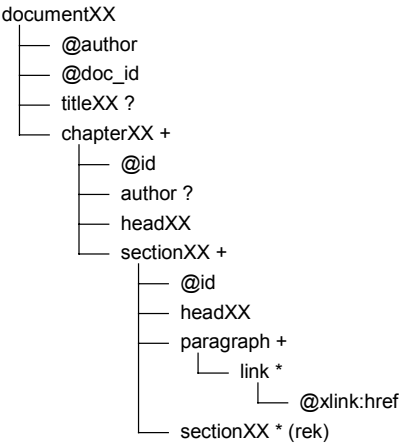


Fig. 1. Node hierarchy of XMach-1 text document

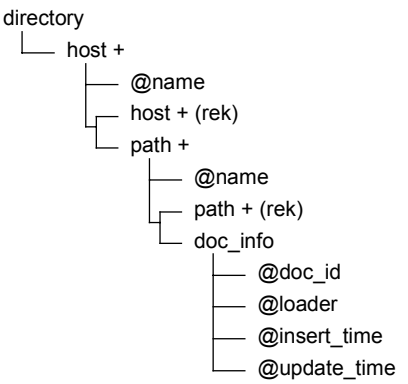


Fig. 2. Node hierarchy of XMach-1 metadata document

2 XMach-1 – An Overview

XMach-1 was developed by us at the University of Leipzig in 2000 and published at the beginning of 2001 [1]. It was thus the first published XML database benchmark with general applicability.

Three objectives are central for the design of the benchmark: scalability, multi-user processing and evaluation of the entire data management system. XMach-1 is based on a web application in order to model a typical use case of an XML data management system. The system architecture consists of four parts: the XML database, application servers, loaders and browser clients. The database contains a directory structure and XML documents that are assumed to be loaded from various data sources in the internet by loader programs (e.g. robots crawling the web or a registration tool where

web-site authors can add documents to the database). Every document has a unique URL which is maintained together with document metadata in the directory structure. The application servers run a web (HTTP) server and other middleware components to support processing of the XML documents and to interact with the backend database.

The XML database contains of both types of documents: document-centric and data-centric. The largest part consists of document-centric XML data which mimic text documents such as books or essays in structure and content. These documents are synthetically produced by a parameterizable generator. In order to achieve realistic results when storing and querying text contents, text is generated from the 10,000 most frequent English words, using a distribution corresponding to natural language text. The documents vary in size (2-100 KB) as well as in structure (flat and deep element hierarchy). Fig. 1 shows the element and attribute hierarchy of these documents.

Table 1. Operations defined in XMach-1

ID	Description	Comment
Q1	Get document with URL X.	Reconstruction of complex structured document with ordering preserved.
Q2	Get doc_id from documents containing phrase X in a paragraph element.	Tests full-text retrieval capabilities.
Q3	Start with first chapter element and recursively follow first section element. Return last section elements.	Simulates navigating a document tree using sequential operators.
Q4	For a document with doc_id X return flat list of head elements which are children of section elements.	Restructuring operation simulating creation of a table of contents.
Q5	Get document name (last path element in directory structure) from all documents which are below a given URL fragment.	Operation on structured unordered data.
Q6	Get doc_id and id of parent element of author element with content X.	Selection using element content.
Q7	Get doc_id from documents which are referenced at least X times.	Tests group by and count functionality.
Q8	Get doc_id from the last X updated documents having an author attribute	Needs count, sort, join and existential operations and accesses metadata.
M1	Insert new document.	Tests insert performance for complex document with activated indices.
M2	Delete document with doc_id X.	Tests deletion performance for complex document with activated indices.
M3	Update name and update_time attributes for document with doc_id X.	Tests efficiency of update operations on attribute values.

The data-centric type is represented by a document containing metadata about the other documents such as URL, name, insert time and update time. All data in this document is stored in attributes (no mixed content) and the order of element siblings

is free. Compared to structured data in relational databases it shows some semi-structured properties like variable path length using recursive elements or optional attributes. The structure of this document is depicted in Fig. 2.

A distinctive feature of XMach-1 is support of a large number of document schemas with 2-100 documents per schema. This allows us to test a database system's ability to cope with a variable number of different element types and to test query execution across multiple schemas. Additionally, the benchmark supports schema-based as well as schema-less document storage.

The database contains at least 1000 text documents and can be scaled by increasing the number of documents by a factor of 10, 100, 1000, etc. The metadata document scales proportionally with the number of text documents. Ratios such as the number of documents per schema, number of authors per document etc. remain constant when scaling the database.

The XMach-1 workload mix consists of 8 query operations and 3 update operations which are described in Table 1. They cover a wide range of processing features like querying complete complex structured documents, full-text retrieval, navigational queries, queries using sorting and grouping operators etc. A formal specification of the operations in XQuery[1] syntax can be found in [4]. Update operations cover inserting and deleting of documents as well as changing attribute values. Despite the missing data manipulation language for update operations, we consider it as essential especially for multi-user performance to evaluate workloads with both queries and updates.

Since XMach-1 is a multi-user benchmark the primary performance metric is throughput measured in Xqps (XML queries per second). This value is calculated from the workload mix which defines firm ratios for each operation. The mix emphasizes the retrieval of complete documents whereas update operations have only a minor share of 2%. Nevertheless the latter one can have a significant impact on the concurrent execution of queries requiring access to the most recent data.

3 Further XML Database Benchmarks

After the specification of XMach-1 was published a number of additional XML database benchmarks were proposed. In this section we briefly introduce the benchmarks XMark, XOO7 and Mbench.

XMark. This benchmark was developed at the National Research Institute for Mathematics and Computer Science (CWI) of the Netherlands and made public in the middle of 2001 [5]. The benchmark focuses on the performance evaluation of the query processor which is reflected by the large number of specified operations. The guidelines of the benchmark design are discussed in [6].

The benchmark data is modeled after an internet auction database. It consists of a number of facts having a firm structure with data-centric aspects. However some document-centric features are introduced by the inclusion of textual descriptions. The

[1] http://www.w3.org/TR/xquery/

complete data is contained within a single document. For most of the element types the sibling order is free.

XMark's operations are made up of 20 queries. No update operations are specified. The queries are designed to capture different aspects of the query processing. Some queries are functional similar to test certain features of the query optimizer. In [7] seven systems were evaluated using XMark. It was shown that no system was able to outperform the others in all disciplines. Rather each physical XML mapping favors certain types of queries for which efficient execution plans could be generated.

XOO7. This benchmark was published [8] shortly after XMark and is a development from the National University of Singapore. It is derived from the object oriented database benchmark OO7 [9] with small changes in the data structure and additional operation types to better meet XML usage patterns.

In contrast to XMach-1 or XMark no specific application domain is modeled by the data. It is rather based on a generic description of complex objects using component-of relationships. This regular and fixed structure having all values stored in attributes exhibits a strong data-centric character. Similar to XMark some document-centric aspects are included using document tags with mixed content. Likewise the database is represented by a single document.

The benchmark only considers query operations grouped by the authors into relational queries, navigational queries and document queries. It defines a total of 23 operations including some newly added queries, especially in the document queries group. The evaluation focus is on the performance of the query processor in single user mode on a central server (1 computer).

Mbench. One of the latest additions to the family of XML database benchmarks is the Michigan Benchmark developed at the University of Michigan [10]. In contrast to its predecessors it is designed as a micro-benchmark aiming to evaluate the cost of individual pieces of core query functionality. Therefore it abstracts from specific application-level approaches defining only well-controlled data access patterns. This kind of benchmark restricts the operation execution to single user mode.

The benchmark data set is a synthetic structure created to simulate different XML data characteristics and to enable operations with predictable costs. Like XMark and XOO7 only one document covers the complete data. The main data structure consists of only one element which is nested with a carefully chosen fanout at every level. With an element hierarchy of 16 and a fixed fanout for each level most of the elements are placed at the deepest level. A second element is used to add intra-document references. The first element contains a number of attributes which can be used to select a defined number of elements within the database. With only two element types and the large number of attributes the data has clearly data-centric properties. Document-centric features are also represented since every element has mixed content and the element sibling order is relevant.

In order to meet the requirements of a micro-benchmark Mbench defines many (56) operations which are grouped into the categories selection queries, value-based join queries, pointer-based join queries, aggregate queries and updates. Within each group, often queries differ only with respect to a specific feature such as selectivity to measure its influence on query performance. Since the data set consists only of two element types typical navigational operations using element names are missing.

4 Benchmark Comparison

In this section we compare the introduced benchmarks with respect to key features such as application focus, evaluation scope, multi-user support, database and workload characteristics, etc. The comparison is intended to help choosing among the benchmarks for a particular evaluation purpose or application domain. Table 2 summarizes the main features which we will now discuss.

The high flexibility of XML leads to vastly different data structures, data types and operations in differents applications. The benchmarks try to accommodate typical characteristics from both document-centric and data-centric XML usage but with different focus. XMach-1 emphasizes the document-centric aspect the most while the other benchmarks focus on data-centric properties with a fixed database structure or a high number of attributes. Mbench is less data-centric (more document-centric) than XMark and XOO7 since each element has textual content.

Table 2. Comparison of XML database benchmarks

	XMach-1	XMark	XOO7	Mbench
main data focus	document-centric	data-centric	data-centric	data-centric
evaluation scope	DBMS	query processor	query processor	core query operators
# user	multi-user	single-user	single-user	single-user
# server	>=1	1	1	1
# documents	10^n (n>=3)	1	1	1
# schemas	#documents/20	1	1	1
# element types	4 * #schemas + 7	74	9	2
DB size	16 KB * #documents	10 MB – 10 GB	ca. 4 MB – 1 GB	50 MB * 10^n (n=1,2,3,4)
#nodes/KB	10	18	67	12
# queries	8	20	23	49
# update op.	3	0	0	7

A fundamental difference between the benchmarks lies in their evaluation scope. With its concept of evaluating the entire database system in multi-user mode XMach-1 covers the user view on the system as a whole. Therefore all components of the database system like query processing, caching, locking, logging etc. are included in the evaluation. The other benchmarks restrict themselves to the evaluation of the query processor in single-user mode to determine the performance for specific queries. XMark and XOO7 evaluate fairly complex queries stressing various features of the query language, while Mbench uses a higher number of smaller operations to systematically evaluate core functions (operators) of the query processor.

All XML data of a database can either be in a single document or spread across several documents. XMach-1 uses many smaller documents with a mean size of 16 KB. This allows easy scalability of the database and gives flexibility to the database system for data allocation, locking, caching etc. The other benchmarks require

the whole database be a single document. This is a significant restriction for some current XML database systems performing document-level locking etc. and would make it difficult to use these benchmarks for multi-user processing.

Since one of the strengths of XML lies in the flexible schema handling it should be natural for an XML database to easily handle multiple schemas. However this feature is only tested in XMach-1. The other benchmarks use a single fixed schema. As a result the number of element types remains unchanged for different database sizes. With its very small number of element types Mbench leads to artificial storage patterns in systems with a element-determined database organization such as some XML-to-relational mapping approaches.

Each of the benchmarks supports a parameterized scaling of the database from a few megabytes to some gigabytes. However the performance for loading the database, querying etc. is not only determined by the size but also by the structural complexity of the data. To give a rough indicator for this we have determined the number of XML nodes[2] per kilobyte data. As indicated in Table 2 for each benchmark this ratio is invariant w.r.t. the database size. XOO7 has by far the highest ratio which stresses its data-centric focus. The large share of textual content in Mbench is also reflected in its comparatively low value.

The differences in evaluation scope can also be seen in the number of query operations. XMach-1 evaluating the whole database system in multi-user mode specifies only a smaller number of complex queries since throughput is the primary metric. XMark and XOO7 having their focus on the query processor use twice as many query operations to capture most query capabilities. Mbench has even more operations to evaluate distinct parts of the core functions of the query processor. Only two benchmarks, XMach-1 and Mbench, consider update operations although they can impact performance substantially.

The comparison shows that both XMach-1 and Mbench have a clear focus. XMach-1 is targeted to evaluating entire database systems in multi-user mode using document-centric data whereas Mbench focuses on evaluating the core operators of the query processor in single user mode. XMark and XOO7 are similar in many respects. Their key differences come from the different schema characteritics. Here XMark has some advantages by supporting a rich structure as well as a more realistic text usage than XOO7.

5 XMach-1 – Experiences and Results

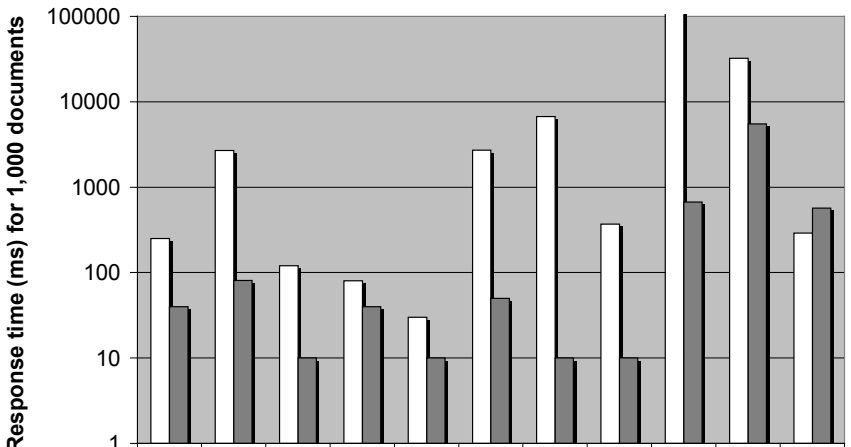
Since the first implementation of XMach-1 in early 2001 we used it to evaluate several XML database systems and subsequent versions of them. We discuss some of our experiences and present some performance results to indicate the performance achieved by current systems.

We started with the evaluation of native XML database systems. Their increased XML functionality over XML-enabled relational DBMSs made it easier to implement XMach-1. Still these products were rather new on the market and exhibited significant limitations, especially w.r.t. full-text indexing and multi-user processing. Prob-

[2] An XML node is either an element or an attributes.

lems were unacceptably long full-text index generation times, lacking support for phrase searches and for indexing across multiple schemas. In multi-user mode we observed substantial locking bottlenecks due to document-level locking leading to very high query response times during parallel writes. Locking at the document level would obviously be completely unacceptable for a database with a single document only. Other locking problems were caused by the index updates for inserting or deleting documents. Some systems were unable to support more than 20 concurrent clients and crashed. In [11] we discuss further problem areas for the first versions of XML database systems.

Most of the issues were resolved in subsequent versions of the systems leading to improved performance. This is exemplified by Fig. 3 showing the 90% percentile response times for the 11 operations (cf. Table 1) specified in XMach-1. Operations Q1-Q8 are queries ranging from document retrieval (Q1) to complex queries involving join, sort and aggregation (Q7, Q8). M1-M3 are data manipulation operations including document insert (M1), document deletion (M2) and updates (M3). A detailed description of the operations can be found in [1]. The measurements for this and the following experiments were carried out on an Intel Pentium III computer running at 800 MHz having 512 MB of main memory and a 40 GB IDE hard disk. The database size was 1000 documents.



	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	M1	M2	M3
old version	250	2684	120	80	30	2713	6709	370	1E+05	32306	291
new version	40	81	10	40	10	50	10	10	671	5518	570

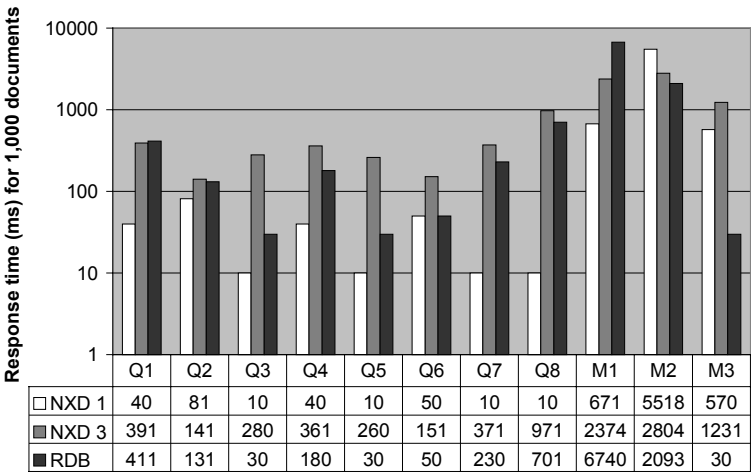
Fig. 3. Comparison of 90th percentile response times for old and new version of a native XML database (NXD 1) in single-user mode

Table 3. Changes in database size for a native XML database system (NXD 2)

	data (MB)	index (MB)
NXD 2 version 1	64,4	72,4
NXD 2 version 2	44,5	35,6
NXD 2 version 3	25,9	31,9

Fig. 3 indicates substantial performance improvements of up to several orders of magnitude between the two versions. All eight query types could be executed with a single-user response time of under 100 ms, favored by the small database size. Some of these improvements were achieved by an optimized benchmark implementation utilizing features of the new version of the system. The high response times for M1 and M2 in both versions stem from an inefficient implementation of the full-text index which has to be updated.

We also observed significant improvements w.r.t. the space requirements to store and index a certain amount of raw XML data. As an example, Table 3 compares the database sizes for data and indexes of consecutive versions of another native XML database system NXD2. The databases were populated using the 1000 documents configuration of the benchmark which has a raw data size of about 16 MB. The changes in the depicted database system resulted in a reduction of the database size of about a factor of 2.5. In general, a ratio of 1.5 to 3 between the database size and the size of raw data is typical for current native XML database systems. Database population takes between 90 and 600 seconds with the 1000 documents setting. Including index generation, 200 to 800 seconds are needed primarily due to full-text indexing. This corresponds to a poor loading throughput of about 20 – 80 KB raw data per second which would result into unacceptable loading times for larger databases.



	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	M1	M2	M3
NXD 1	40	81	10	40	10	50	10	10	671	5518	570
NXD 3	391	141	280	361	260	151	371	971	2374	2804	1231
RDB	411	131	30	180	30	50	230	701	6740	2093	30

Fig. 4. Comparison of 90th percentile response times in single-user mode

When we recently started our evaluation of XML-enabled relational database systems (XRDB) we still found functional shortcomings compared to native XML database systems. One major drawback is insufficient support of an XML query language.

Whereas native XML databases have at least a complete XPath implementation and are starting to support XQuery as well, XRDB's have in most cases only a limited support of XPath with restricted utilization of indices. Another problem is that current XRDBs cannot efficiently run queries across multiple schemas because of their schema driven XML architecture. On the other hand, XRDBs benefit from mature relational functionality and comparably efficient full-text support.

In Fig. 4 and Fig. 5 we compare results for current XML databases running XMach-1. Systems NXD 1 and NXD 3 are commercial native XML databases whereas system RDB is a standard relational database system. The relational implementation of XMach-1 uses a newly developed middleware for a generic mapping of XML data to relations. The RDB mapping is independent of an XML schema and uses only three tables for elements, attributes and large element contents. The mapping supports a sophisticated numbering scheme for XML nodes incurring low renumbering effort for updates as well as a fast navigation and extraction of document fragments.

The 90th percentile response time of all XMach-1 operations for the three database systems are shown in Fig. 4. As can be seen NXD 1 outperforms both other systems in most query operations by an order of magnitude. However update operations are quite slow which stems partly from the full-text index deficiency. The fast execution of Q7 and Q8 was achieved by using extra data structures automatically maintained by database triggers. These optimizations for queries come at the expense of increased overhead for loading and inserting/deleting the XML documents. NXD 3 and RDB exhibit comparable performance figures. The mentioned mapping approach with its optimizations was key to the remarkably good query performance of RDB.

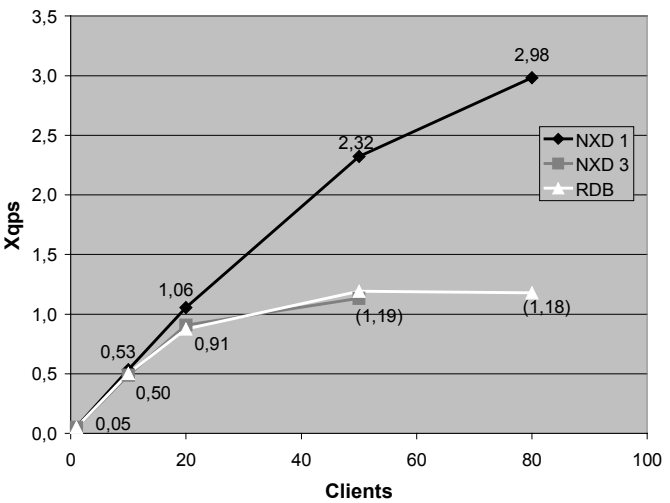


Fig. 5. Throughput comparison

Fig. 5 illustrates the multi-user performance of the three systems. The throughput value Xqps (XML queries per second) measures the number of Q1 operations per second within the query mix. Each client has to wait between two consecutive re-

quests for 1-10 seconds. The small database size largely excludes congestion accessing the external storage devices. Therefore throughput is bound by CPU and locking bottlenecks raised by update operations. NXD 1 again reaches the best value and scales nearly linearly until 50-80 clients. With approximately 3 Xqps it achieves the top value of all evaluated XML database systems so far. Both other systems reach their maximum with 0.9 Xqps and 20 clients. For more clients somewhat higher throughput values (shown in parentheses) were achieved but without meeting the 3 second response time limit of XMach-1.

We started to examine multi-user performance for larger databases requiring a higher degree of IO activity. Some systems had significant scalability problems preventing the execution of some query types. Response times for some queries were order of magnitudes higher than with a 1,000 documents collection. Our XML-to-relational mapping also faced scalability problems for larger data sizes since the relational query optimizer cannot use information provided by the numbering scheme.

6 Conclusion

XML database benchmarks allow to compare the performance of XML database systems for a well-defined environment and operation mix. We reviewed and compared four proposed benchmarks to reveal their primary focus and applicability. From these benchmarks, XMach-1 is the only one supporting the performance evaluation for both single-user and multi-user processing. Moreover it considers not only the query processor but measures the performance of an entire XML database system. Furthermore, it has a focus on document-centric XML databases and considers schema-less and schema-based document collections.

Our experiences with XMach-1 have shown that functionality and performance of XML database systems have considerably improved during the last two years. Native systems have generally performed better than XML-enabled relational database systems. Still, our prototype implementation of a generic XML to relational mapping indicates that generic XML data management on relational databases can reach comparable performance to native XML databases at least for smaller databases. Most XML database systems still face significant scalability problems with larger data volumes and multi-user mode. Hence, there is a big need for further performance improvements and enhanced implementation concepts for XML data management.

References

[1] Böhme, T.; Rahm, E.: XMach-1: A Benchmark for XML Data Management. In Proceedings of German database conference BTW2001, pp. 264-273, Springer, Berlin, March 2001.

[2] Florescu, D.; Kossmann, D.: Storing and Querying XML Data using an RDMBS. In: IEEE Data Engineering Bulletin, Volume 22, Number 3, pp. 27-34, September 1999.

[3] Florescu, D.; Kossmann, D.; Manolescu, I.: Integrating Keyword Search into XML Query Processing. In: Proc. of the 9th WWW Conference, Amsterdam, June 2000.
[4] Böhme, T.; Rahm, E.: Benchmarking XML Data Management Systems. http://dbs.uni-leipzig.de/en/projekte/XML/XmlBenchmarking.html, June 2002.
[5] Schmidt, A.; Waas, F.; Kersten, M. L.; Florescu, D.; Manolescu, I.; Carey, M. J.; Busse, R.: The XML Benchmark Project. Technical Report INS-R0103, CWI, Amsterdam, Niederlande, April 2001.
[6] Schmidt, A.; Waas, F.; Kersten, M. L.; Florescu, D.; Carey, M. J.; Manolescu, I.; Busse, R.: Why And How To Benchmark XML Databases. SIGMOD Record, Volume 30, Number 3, pp. 27-32, September 2001.
[7] Schmidt, A.; Waas, F.; Kersten, M. L.; Carey, M. J.; Manolescu, I., Busse, R.: XMark: A Benchmark for XML Data Management. In Proceedings of the 28th VLDB Conference, Hong Kong, 2002.
[8] The XOO7 Benchmark. http://www.comp.nus.edu.sg/~ebh/XOO7.html, 2002
[9] Carey, M. J.; DeWitt, D. J.; Naughton, J. F.: The OO7 Benchmark. In Proceedings of the ACM SIGMOD International Conference on Management of Data, pp.12-21, June 1993.
[10] Runapongsa, K.; Patel, J. M.; Jagadish, H. V.; Al-Khalifa, S.: The Michigan Benchmark. http://www.eecs.umich.edu/db/mbench/description.html, 2002.
[11] Böhme, T.; Rahm, E.: Benchmarking XML Database Systems – First Experiences. Ninth International Workshop on High Performance Transaction Systems (HPTS), Pacific Grove, California, October 2001.