Current Trends in Testing XMLMSs

Irena Mlynkova

Department of Software Engineering, Charles University in Prague, Czech Republic
irena.mlynkova@mff.cuni.cz

Abstract Since XML technologies have become a standard for data representation, a huge amount of XMLMSs have emerged as well. Consequently, it is necessary to be able to experimentally test and compare their versatility, behaviour and efficiency. In this paper we provide an overview of existing approaches to testing XMLMSs and we discuss respective consequences and recommendations.

Keywords: XML benchmarking, XML test suites, XML data generators.

1. Introduction

Since XML [Bray et al. 2006] has become a de-facto standard for data representation and manipulation, there exists a huge amount of so-called XML processing tools, or more comprehensive *XML Management Systems* (XMLMSs) dealing with efficient management of XML data. They usually involve methods for storing, validating and querying of XML documents, nevertheless, there may also exist XMLMSs which support more complex operations with XML data, such as, e.g., transforming, updating, exchanging, compressing etc. Consequently, being users, we need to know which of the existing XMLMSs is the most sufficient for our particular application. On the other hand, being vendors who develop a new XMLMS, we need to test correctness and performance of our system and, especially, to compare its main advantages with competing SW. And being analysts, we are especially interested in comparison of various aspects of existing systems from different points of view.

A natural solution is to find results of an appropriate analysis. But, although there currently exists a number of such analytical papers, the speed of development of new XMLMSs is high and, hence, their results soon become obsolete. On the other hand, if we manage to find reasonably up-to-date analytical results, the testing scenarios usually do not fit well to all our use cases. Hence, in most situations we still need to prepare our own testing sets of data and operations that reasonably represent our situation.

The aim of this paper is to provide an overview of possibilities how to acquire or prepare XML testing scenarios and what are the limitations of the current approaches. We focus on existing conformance test suites, repositories of real-world XML data, XML benchmarking projects and data generators and we describe and discuss their main characteristics and especially issues related to their versatility. Finally, we provide an overview of the key findings and related recommendations. We will not compare particular XMLMSs, but systems and methods using which they can be tested and compared. We will show that the solutions for basic XML technologies are common, but in case of advanced or new ones the situation is much worse.

The text is structured as follows: The second section classifies and describes the existing approaches. Section 3 provides a summary of the findings and recommendations. And, finally, Section 4 provides conclusions.

2. Overview of Existing Approaches

Currently, there exists a number of approaches to experimental testing of XMLMSs and they can be classified variously. In general, a *benchmark* or a *test suite* is a set of *testing scenarios* or *test cases*, i.e. data and related operations which enable to compare versatility, efficiency or behavior of *systems under test* (SUT). In our case the set of data involves XML documents, possibly with their XML schemes, whereas the set of operations can involve any kind of XML-related data operations.

From the point of view of the type of data we can distinguish benchmarks which involve real-world data and benchmarks involving synthetic data. Though the former type seems to have more reasonable application, the problem is that according to analyses [Mlynkova et al. 2006] real-world XML documents are quite simple and do not cover most of the constructs allowed by W3C specifications. For instance, there are methods whose space complexity is closely related to depth of XML documents [Dvorakova et al. 2008] or methods that require training a neural network on particular data [Stanclova 2008]. And acquiring such specific real-world XML data can be difficult. A different type of classification distinguishes approaches involving a fixed set of testing data sets and approaches which enable to create them dynamically on the basis of user-specified parameters.

On the basis of the purpose of an XML benchmark, we can further distinguish benchmarks which deal with various types of data operations, such as, e.g., parsing, validating, querying, updating or transforming. And in particular areas we can establish also more finer classification on the basis of exploited languages and constructs, such as, e.g., DTD [Bray et al. 2006] vs. XML Schema [Thompson et al. 2004; Biron et al. 2004] benchmarks, XQuery [Boag et al. 2007] vs. XPath benchmarks, XPath 1.0 [Clark et al. 1999] vs. XPath 2.0 [Berglund et al. 2007] benchmarks etc.

2.1 XML Data Sets

Currently, one of the most typical approaches to XML benchmarking is exploitation of fixed sets of real-world XML data that represent a particular field of XML processing. Apart from rather interesting than useful examples of XML documents, such as, e.g., the Bible in XML [Fields 1996], Shakespeare's plays [Bosak 1997], classic novels in XML [Wendover 2001] etc., the most common types of testing data sets are usually XML exports of various databases, such as, e.g., *IMDb* [IMDb 2008] database of movies and actors, *FreeDB* [FreeDB 2008] database of musical CDs, *DBLP* [DBLP 2008] database of scientific papers, *Medical Subject Headings* [MeSH 2008] database of medical terms etc. or repositories of real-world XML data coming from various resources, such as, e.g., project *INEX* [INEX 2007], *Open Directory Project* [Open Directory 2004] etc. There also exist examples of rather special XML data, such as, e.g., human genes [H-invDB 2007], protein sequences [UniProt 2008], astronomical NASA data [XDR 2002], linguistic trees in XML [Treebank 1999] etc., having very uncommon structure and, hence, requiring special processing. Some of these collections were not originally in XML format, but they were converted and stored in XML repositories.

As mentioned before, the real-world XML data have two disadvantages. They are usually very simple and most of them are provided without respective operations. Hence, though they represent a realistic resource of information, they enable to test only a limited set of aspects or situations.

2.2 XML Data Generators

A natural solution to the previous problem is to generate the testing data sets synthetically. Currently, we can find several implementations of XML data generators which generate XML data on the basis of user-provided setting of parameters.

The methods can be classified on the basis of the input parameters. The most general classification differentiates so-called *schema-unaware* and *template-based* generators. The schema-unaware generators, such as, e.g., *NiagDataGen* [Aboulnaga et al. 2001], support general structural parameters (e.g., number of levels of the required XML trees, numbers of subelements at each level etc.) and exploit various strategies, such as, e.g., Zip’s law, Markov chains, statistical distributions etc. to generate as realistic structure as possible randomly, but within the parameters. On the other hand, the template-based generators, such as, e.g., *ToXgene* [Barbosa et al. 2002], *VeXGene* [Jeong et al. 2006], *MemBeR* [Afanasiev et al. 2005], get on input a kind of annotated XML schema and generate XML documents valid against it. The structure of the resulting data is specified using the schema more precisely (although the full generality of DTD or XML Schema languages is not usually supported), whereas the annotations provide even more spe-

cific information, such as, e.g., probability distributions of occurrences of elements/attributes or lengths of string literals.

Apart from specification of structure of the required data, the generators also often deal with problems such as, e.g., where to get the textual data or element/attribute names to achieve as natural result as possible. For some applications, such as, e.g., XML full-text operations or XML compressing, may be the content of textual data important, but for techniques related to parsing, validating or querying the aspects are of marginal importance.

In general, the biggest advantage of the data generators is that they usually support a huge number of parameters a user can specify and, hence, provide quite a precise result. But, on the other hand, this is also a big disadvantage, because the user must know all these parameters of the required data. And this is of course realistic only in case of XML experts. Similarly to the case of real-world XML data, the synthetic XML data are not accompanied with respective operations as well. In fact, there seems to be no generator of, e.g., XPath queries over the given data having specified features.

2.3 Parsing and Validating XML Data

The basic operations with XML data are parsing and validating, i.e. checking their correctness. It involves conformance to W3C recommendations and, eventually, existing XML schemes. Naturally, the testing scenarios consist of correct and incorrect data and the aim is to test whether the SUT recognizes them correctly.

XML Conformance Test Suites

The W3C consortium provides so-called *XML Conformance Test Suites* [Martinez et al. 2008], i.e. a set of metrics to determine how well a particular implementation conforms to *XML 1.0 (Second Edition)*, *XML 1.0 (Third Edition)*, *XML 1.1 (First Edition)* and *Namespaces in XML 1.1*. It consists of a set of 2000 XML documents which can be divided into two basic types – *binary* and *output tests*. Binary tests contain a set of valid XML documents, invalid XML documents, non-well-formed XML documents, well-formed errors tied to external entities and XML documents with optional errors. Depending on the category, the tested parser must either accept or reject the document correctly (therefore, the tests are called binary). On the other hand, the output tests enable to test whether the respective applications report information as required by the recommendations.

XML Parsers

On the other hand, the key users' interest in XML parsing and validating is efficiency and space overhead. Currently, we can distinguish so-called *event-driven* and *object-model* parsers. The former ones read the document and while reading they return the respective fragments of data, whereas the latter ones read the

document and build its complete model in memory. The former ones can be further divided into *push-parsers* and *pull-parsers*. In case of push-parsers the reading cannot be influenced, whereas pull-parsers read the next data only if they are "asked" to. And, later, there have also occurred their various combinations. Naturally, each of the approaches has its (dis)advantages and limitations that need to be experimentally tested.

Currently, there exist analyses [e.g., Farwick et al. 2007; Oren 2002] comparing either same or distinct types of XML parsers. But, they all involve only a selected subset of real-world or synthetic XML data. Although the authors usually make the testing sets as well as *test harnesses*[1] available, there seems to be no true benchmarking project which would enable to analyze all aspects and especially bottlenecks of XML parsing.

2.4 Querying XML Data

Since the key operation with XML data is undoubtedly querying, the biggest set of conformance test suites and XML benchmarks focuses on it. The W3C provides the *XML Query Test Suite (XQTS 1.0.2)* [Rorke et al. 2007] which contains over 15000 test cases, i.e. queries and expected results, which enable to test whether the W3C XML Query Language is fully supported. There also exists a set of W3C *XML Query Use Cases* [Chamberlin et al. 2007], i.e. examples illustrating important applications for an XML query language. Though they are not considered as test cases, they are often used as minimal requirements for an XML benchmark.

In general, there exists a large set of true XML query benchmarking projects. Their aim is to analyze versatility and performance of XML querying tools, i.e. the amount of query constructs they support and how efficiently they are processed. The seven best known representatives are *XMark* [Busse 2003], *XOO7* [Bressan et al. 2003], *XMach-1* [Bohme et al. 2001], *MBench* [Runapongsa et al. 2006], *XBench* [Yao et al. 2003], *XPathMark* [Franceschet 2005] and *TPoX* [Nicola et al. 2007]. The overview of their key features according to which they can be classified is depicted in Table 1.

The first characteristic of a benchmark is its type. We differentiate so-called *application-level* and *micro* benchmarks. Since an application-level benchmark is created to compare and contrast various applications, a micro-benchmark should be used to evaluate performance of a single system in various situations. Consequently, the operations respectively differ. In the former case the queries are highly different trying to cover the key situations, whereas in the latter case they can contain subsets of highly similar queries which differentiate, e.g., in selectivity. Note that the only representative of micro-benchmarks is MBench project.

[1] A software that tests a set of programs by running them under varying conditions and monitor their behavior and outputs.

Table 1. Main characteristics of XML query benchmarks

	XMark	**XOO7**	**XMach-1**	**MBench**	**XBench**	**XPathMark**	**TPoX**
Type of benchmark	Application-level	Application-level	Application-level	Micro	Application-level	Application-level	Application-level
Number of users	Single	Single	Multiple	Single	Single	Single	Multiple
Number of applications	1	1	1	1	4	1	1 but complex
Documents in data set	Single	Single	Multiple	Single	Single/ multiple	Single	Multiple
Data generator	✓	✓	✓	✓	✓	✓	✓
Key parameters	Size	Depth, fan-out, size of textual data	Number of documents / elements / words in a sentence, probability of phrases / links	Size	Size	Size	Size + number of users
Default data set	Single 100MB document	3 documents (small, medium, large) with predefined parameters	4 data sets of 10000 / 100000 / 1000000 / 10000000 documents	Single document with 728000 nodes	Small (10MB) / normal (100MB) / large (1GB) / huge (10GB) document	1 XMark document and 1 sample document from a book	XS (3.6 millions of documents, 10 users), S, M, L, XL, XXL (360 billions of documents, 1 million users)
Schema of documents	DTD of an Internet auction database	DTD derived from OO7 relational schema	DTD of an document having chapters, paragraphs and sections	DTD / XSD of the recursive element	DTD / XSD	DTD	XSD
Number of schemes	1	1	Multiple	9	1	2	1 consisting of multiple
Number of queries	20	23	8	49	19, 17, 14, 16	47 + 12	7
Query language	XQuery	XQuery	XQuery	SQL, XPath	XQuery	XPath	XQuery
Number of updates	0	0	3	7	0	0	10

Another set of benchmark characteristics describe the general purpose of the benchmark, i.e. the number of users it is intended for, the number of applications it simulates and the number of documents within its data set. As we can see, most of the benchmarks are single-user, single-application and involve only a single document. This observation meets the general criteria that a benchmark should be sim-

ple. Nevertheless, while the single-document data set is not of a great problem, the single-application feature can be highly restrictive. The only exception, XBench, involves four classes of XML applications with different requirements – text-centric/single document (TC/SD), text-centric/multiple documents (TC/MD), data-centric/single document (DC/SD) and data-centric/multiple documents (DC/MD). Nevertheless, there are XML use cases that cannot be simulated only using various data sets. XMach-1 and TPoX projects are multi-user and enable to test other XML management aspects, such as, e.g., indexing, schema validation, concurrency control, transaction processing, network characteristics, communication costs etc. They both consist of four parts – an XML database, application server(s), loaders and browser clients. The SUT is represented via the application server which interacts with the XML database. The loaders load and delete various XML data into/from the database via the application servers. And browser clients are assumed to query and retrieve the stored XML data. In addition, since most of the features of the systems can be controlled via parameters, they can be even set to query-only, single-user, single-document.

Another important aspect of XML benchmarking projects are characteristics of the data sets. As we can see, all the representatives involve a data generator that enables to influence parameters of the synthetic data and various default data sets. But, on the other hand, in most cases the only parameter that can be specified is the size of the data. Most of the benchmarks involve own simple data generator, some of them (i.e. XBench and TPoX) exploit a more complex data generator (in the two cases ToXgene), but pre-set a subset of its parameters. Also note that all the projects involve also one or more DTDs or XSDs[2] of the data.

The last important set of characteristics describes the operation set of the projects. All the projects involve a set of queries, some of them (i.e. XMach-1, MBench and TPoX) also a set of update operations. As we have mentioned before, the two multi-user benchmarks also support additional, less XML-like operations with the data. Nevertheless, the most popular operations are XQuery queries or in some cases (i.e. MBench and XBench) the queries are specified abstractly and, hence, can be expressed in any language, though their XQuery expression is usually provided as well. The queries try to cover various aspects of the language, such as, e.g., ordering, casting, wildcard expressions, aggregations, references, constructors, joins, user-defined functions etc.

Analysis of Benchmarking Projects

Paper [Afanasiev et al. 2006] analyzes the first five benchmarking projects and deals with their purpose, versatility, current usability etc. The authors have found out that only 1/3 of papers on XQuery processing use a kind of benchmark which is probably caused by the fact that 38% of benchmark queries are incorrect or outdated. In addition, 29% of the queries are XPath 1.0 queries, 61% are XPath 2.0 queries and only 10% cannot be expressed in XPath. An important finding is that

[2] XML Schema definitions

the most popular benchmarking project seems to be the simple XMark. It indicates that users do not want to bother with complicated setting of parameters.

Benchmark Repository
From the overview of the benchmarks and their various features it is obvious, that a single fixed set of queries cannot allow testing of various aspects of applications. Hence, the main aim of the *MemBeR repository* [Afanasiev et al. 2005] of micro-benchmarks is to allow users to add new data sets and/or queries for specific performance assessment tasks. The repository has a predefined structure involving XML documents and their parameters, XML queries and their parameters, experiments and their parameters (i.e. related documents and/or queries), micro-benchmarks (i.e. sets of experiments) and micro-benchmark result sets. A new micro-benchmark or a new result set must be specified as an XML document conforming to a pre-defined DTD which describes all the related characteristics.

Currently the repository contains three categories of benchmarks – XPath, query stability and XQuery. The benchmarks can be further classified into performance, consumption, correctness and completeness benchmarks on the basis of the resulting metric, type of scalability (data/query), usage of schema, query processing scenarios (e.g., persistent database, streaming etc.), query language and tested language feature.

2.5 Transforming, Updating and Other Operations with XML Data

Since the key aspects of XML processing are undoubtedly parsing, validating and querying, most of the existing benchmarking projects focus mainly on them. But, there are also other popular and useful XML technologies, such as, e.g., XSL transformations [Clark 1999], XML compression methods etc., and, hence, there occur also benchmarks determined for other purposes. Surprisingly, the number of such special-purpose projects is low or the only existing representatives are quite old and, hence, obsolete. An example of the situation is benchmarking of XSL transformations. The only known benchmarking project is *XSLTMark* [Kuznetsov et al. 2000] which is not maintained anymore and supports only constructs of version 1.0 from 1999. Similarly, there exist several analytical papers which compare a subset of XSLT processors [e.g., Kuznetsov et al. 2001; Caucho 2005], nevertheless, most of them are based on the obsolete XSLTMark data set or distinct sets of real-world data.

From one point of view the situation may be caused by the fact that most of other technologies, such as, e.g., XSLT, XPointer [DeRose et al. 2002], XLink [DeRose et al. 2001] etc., are based on one of the basic ones, mostly XPath queries. Thus, an argument against special benchmarking projects may be that projects for benchmarking XML queries are sufficient enough. But, on the other hand, the exploitation of, e.g., XPath in XSL can be quite different from typical

exploitation in data retrieval. And, in addition, there are other important aspects of XSL transformations than the path queries which influence their correctness and efficiency. Furthermore, if we consider even more special operations on XML data, such as, e.g., XML compressing, the respective benchmark may deal with features which are for other types of XML processing marginal. However, the amount of these special purpose benchmarks is still low – we can hardly find at least a single representative for each of the areas.

On the other hand, there are XML technologies that have become popular only recently and, consequently, their benchmarking projects are relatively rare. A representative of this situation is XML updating. As we can see in Table 1, some of the existing query benchmarks involve few update operations, but a true XML update benchmarking project has been proposed only recently [Phan et al. 2008]. And a similar situation can be found in case of technologies related to semantic web [Dokulil et al. 2008].

3. Summary

We can sum up the state of the art of existing XML benchmarking projects into the following findings and recommendations:

- The most typical source of testing XML data are repositories with fixed, real-world XML data. But though the data are realistic, they are usually too simple to cover all possible XML constructs and, mostly, they are not accompanied with respective operations.
- A solution to this problem can bring various generators of synthetic XML data. They enable to specify the precise structure of the target data and exploit various approaches to simulate real-world situations. Nevertheless, the problem is that such systems require a user well skilled in XML technologies and, especially, data characteristics. And, naturally, these data are not accompanied with respective operations as well.
- Since parsing and validating are two most important basic operations with XML data, the W3C has defined appropriate conformance test suites which enable to test their correct behaviour. Hence, this area of testing sets is well covered.
- On the other hand, the key users' interest of XML parsing is efficiency and space overhead. Although there exist several papers and projects dealing with this topic that provide results of respective analyses, there seems to be no true test suite that would cover the key influencing aspects and bottlenecks.
- The second key operation on XML data is undoubtedly querying. The W3C provides the XML Query Test Suite and XML Query Use Cases which enable to test the full support of the language and provide a set of typical application of XML querying. Furthermore, there exist several

well-known and verified benchmarking projects with different purposes, features, advantages and disadvantages.

- Although all the query benchmarking projects involve a kind of data generator, the most popular ones seem to be those which are of simple usage (e.g., XMark), i.e. having only few parameters to specify. On the other hand, these benchmarks usually provide only very simple data, of one special type and complexity.
- In general, the area of query benchmarks is relatively wide and the projects usually try to cover the key query operations. But if we consider other XML technologies which involve path queries, such as, e.g., XSLT, XPointer, XLink etc., the typical usage can strongly differ. Hence, these technologies require special treatment and special benchmark projects. Surprisingly, in these areas the amount of respective benchmarks is surprisingly low. Mostly there exists no appropriate benchmark project.

The general observation of our analysis is that the basic XML data operations, i.e. parsing, validating and querying, ale well covered with respective test suites and benchmarking projects. The situation in case of other technologies is much worse. Nevertheless, in these cases we can always exploit either real-world XML data or, if they do not cover our test cases, synthetically generated data sets.

On the other hand, this situation opens a wide research area of both proposing special-purpose benchmarking projects and test suites, as well as performing respective analyses of existing implementations.

4. Conclusion

The main goal of this paper was to describe and discuss the current state of the art and open issues of ways how to test XMLMSs. We have dealt especially with the problem of gathering or preparing the testing data sets and operations. We have focussed mainly on conformance test suites, repositories of real-world XML data, XML benchmarking projects and data generators. We have provided an overview and classification of the existing approaches and their features and summed up the key findings and recommendations. In general, this paper should serve as a good starting point for users, developers and analysts who are interested in testing a selected subset of XMLMS implementations.

Acknowledgments This work was supported by the National Programme of Research (Information Society Project 1ET100300419).

References

Aboulnaga, A., Naughton, J.F. & Zhang, C. (2001): Generating Synthetic Complex-structured XML Data. In WebDB'01: Proc. of the 4th Int. Workshop on the Web and Databases. Santa Barbara, California. Informal proceedings.

Afanasiev, L., Manolescu, I. & Michiels, P. (2005): MemBeR: A Micro-Benchmark Repository for XQuery. In XSym'05: Proc. of 3rd Int. XML Database Symp., LNCS. Springer-Verlag.

Afanasiev, L. & Marx, M. (2006): An Analysis of the Current XQuery Benchmarks. In ExpDB'06: Proc. of the 1st Int. Workshop on Performance and Evaluation of Data Management Systems, pp. 9 – 20, Chicago, Illinois, USA. ACM.

Barbosa, D., Mendelzon, A. O., Keenleyside, J. & Lyons, K. A. (2002): ToXgene: A Template-Based Data Generator for XML. In SIGMOD'02: Proc. of the 2002 ACM SIGMOD Int. Conf. on Management of Data, page 616, Madison, Wisconsin, USA. ACM.

Berglund, A., Boag, S., Chamberlin, D., Fernandez, M. F., Kay, M., Robie, J. & Simeon, J. (2007): XML Path Language (XPath) 2.0. W3C. http://www.w3.org/TR/xpath20/.

Biron, P. V. & Malhotra, A. (2004): XML Schema Part 2: Datatypes (Second Edition). W3C. www.w3.org/TR/xmlschema-2/.

Boag, S., Chamberlin, D., Fernandez, M. F., Florescu, D., Robie, J. & Simeon, J. (2007): XQuery 1.0: An XML Query Language. W3C. http://www.w3.org/TR/xquery/.

Bohme, T. & Rahm, E. (2001): XMach-1: A Benchmark for XML Data Management. In BTW'01: Datenbanksysteme in Buro, Technik und Wissenschaft, 9. GI-Fachtagung, pp. 264 – 273, London, UK. Springer-Verlag.

Bosak, J. (2007): Jon Bosak's XML examples. http://www.ibiblio.org/bosak/.

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E. & Yergeau, F. (2006): Extensible Markup Language (XML) 1.0 (Fourth Edition). W3C. http://www.w3.org/TR/REC-xml/.

Bressan, S., Lee, M.-L., Li, Y. G., Lacroix, Z. & Nambiar, U. (2003): The XOO7 Benchmark. In Proc. of the VLDB 2002 Workshop EEXTT and CAiSE 2002 Workshop DTWeb on Efficiency and Effectiveness of XML Tools and Techniques and Data Integration over the Web-Revised Papers, pp. 146 – 147, London, UK. Springer-Verlag.

Busse, R., Carey, M., Florescu, D., Kersten, M., Manolescu, I., Schmidt, A. & Waas, F. (2003): XMark – An XML Benchmark Project. Centrum voor Wiskunde en Informatica (CWI), Amsterdam. http://www.xml-benchmark.org/.

Carey, M. J., DeWitt, D. J. & Naughton, J. F. (1993): The OO7 Benchmark. SIGMOD Record (ACM Special Interest Group on Management of Data), 22(2), pp. 12 – 21.

Caucho (2005): XSLT Benchmark. Caucho Technology, Inc. http://unsemaul.sportsseoul.com/resin-doc/features/xslt-benchmark.xtp.

Chamberlin, D., Fankhauser, P., Florescu, D., Marchiori, M. & Robie, J. (2007): XML Query Use Cases. W3C. http://www.w3.org/TR/xquery-use-cases/.

Clark, J. (1999): XSL Transformations (XSLT) Version 1.0. W3C. http://www.w3.org/TR/xslt.

Clark, J. & DeRose, S. (1999): XML Path Language (XPath) Version 1.0. W3C. http://www.w3.org/TR/xpath/.

DBLP (2008): Digital Bibliography & Library Project. http://dblp.uni-trier.de/.

DeRose, S., Daniel, R., Grosso, P., Maler, E., Marsh, J. & Walsh, N. (2002): XML Pointer Language (XPointer). W3C. http://www.w3.org/TR/xptr/.

DeRose, S., Maler, E. & Orchard, D. (2001): XML Linking Language (XLink) Version 1.0. W3C. http://www.w3.org/TR/xlink/.

Dokulil, J., Yaghob, J. & Katreniakova, J. (2008): Everything You Ever Wanted to Learn from the Semantic Web, but Were Unable to Ask. In ADVCOMP'08: Proc. of the 2nd Int. Conf. on Advanced Engineering Computing and Applications in Sciences, Valencia, Spain. IEEE.

Dvorakova, J. & Zavoral, F. (2008): Xord: An Implementation Framework for Efficient XSLT Processing. In IDC'08: Proc. of the 2nd Int. Symposium on Intelligent Distributed Computing, Catania, Italy. Springer-Verlag.

Farwick, M. & Hafner, M. (2007): XML Parser Benchmarks: Part 1 & 2. XML.com. http://www.xml.com/pub/a/2007/05/09/xml-parser-benchmarks-part-1.html. http://www.xml.com/pub/a/2007/05/16/xml-parser-benchmarks-part-2.html.

Fields, M. (1996): Mark Fields's Ebooks. http://www.assortedthoughts.com/downloads.php.

Franceschet, M. (2005): XPathMark – An XPath Benchmark for XMark Generated Data. In XSym'05: Proc. of 3rd Int. XML Database Symposium, LNCS. Springer-Verlag.

FreeDB (2008): http://www.freedb.org/.

H-InvDB (2007): Annotated Human Genes Database. http://www.jbirc.aist.go.jp/hinv/.

IMDb (2008): The Internet Movie Database. http://www.imdb.com/.

INEX (2007): INitiative for the Evaluation of XML Retrieval. http://inex.is.informatik.uni-duisburg.de/.

Jeong, H. J. & Lee, S.H. (2006): A Versatile XML Data Generator. Int. J. of Software Effectiveness and Efficiency, 1(1), pp. 21 – 24.

Kuznetsov, E. & Dolph, C. (2000): XSLT Processor Benchmarks. XML.com. http://www.xml.com/pub/a/2001/03/28/xsltmark/index.html.

Kuznetsov, E. & Dolph, C. (2001): XSLT Benchmark Results. XML.com. http://www.xml.com/pub/a/2001/03/28/xsltmark/results.html.

Martinez, S. I., Grosso, P. & Walsh, N. (2008): Extensible Markup Language (XML) Conformance Test Suites. W3C. http://www.w3.org/XML/Test/.

MeSH (2008): Medical Subject Headings. http://www.nlm.nih.gov/mesh/meshhome.html.

Mlynkova, I., Toman, K. & Pokorny, J. (2006): Statistical Analysis of Real XML Data Collections. In COMAD'06: Proc. of the 13th Int. Conf. on Management of Data, pp. 20 – 31, New Delhi, India. Tata McGraw-Hill Publishing Company Ltd.

Nicola, M., Kogan, I. & Schiefer, B. (2007): An XML Transaction Processing Benchmark. In SIGMOD'07: Proc. of the 2007 ACM SIGMOD Int. Conf. on Management of Data, pp. 937 – 948, New York, NY, USA. ACM.

Open Directory Project (2004): http://rdf.dmoz.org/.

Oren, Y. (2002): SAX Parser Benchmarks. SourceForge.net. http://piccolo.sourceforge.net/bench.html.

Phan, B. V. & Pardede, E. (2008): Towards the Development of XML Benchmark for XML Updates. In ITNG'08: Proc. of the 5th Int. Conf. on Information Technology: New Generations, pp. 500 – 505, Las Vegas, Nevada, USA. IEEE.

Rorke, M., Muthiah, K., Chennoju, R., Lu, Y., Behm, A., Montanez, C., Sharma, G. & Englich, F. (2007): XML Query Test Suite. W3C. http://www.w3.org/XML/Query/test-suite/.

Runapongsa, K., Patel, J. M., Jagadish, H. V., Chen, Y. & Al-Khalifa, S. The Michigan benchmark: towards XML Query Performance Diagnostics (Extended Version) http://www.eecs.umich.edu/db/mbench/mbench.pdf.

Stanclova, J. (2006): The Associative Recall of Spatial Correlated Patterns. In CIARP'06: Proc. of 11th Iberoamerican Congress on Pattern Recognition, pp. 539 – 548, Cancun, Mexico. Springer-Verlag.

Thompson, H. S., Beech, D., Maloney, M. & Mendelsohn, N. (2004): XML Schema Part 1: Structures (Second Edition). W3C. www.w3.org/TR/xmlschema-1/.

Treebank (1999): The Penn Treebank Project. http://www.cis.upenn.edu/~treebank/.

UniProt (2008): Universal Protein Resource. http://www.ebi.uniprot.org/index.shtml.

Wendover, A. (2001): Arthur's Classic Novels. http://arthursclassicnovels.com/.

XDR (2002): XML Data Repository. www.cs.washington.edu/research/xmldatasets/www/repository.html.

Yao, B. B., Ozsu, M. T. & Keenleyside, J. (2003): XBench – A Family of Benchmarks for XML DBMSs. In Proc. of the VLDB 2002 Workshop EEXTT and CAiSE 2002 Workshop DTWeb on Efficiency and Effectiveness of XML Tools and Techniques and Data Integration over the Web-Revised Papers, pp. 162 – 164, London, UK. Springer-Verlag.