UNIVERSITÄT LEIPZIG

University of Leipzig

# Benchmarking XML Database Systems – First Experiences


Timo Böhme
boehme@informatik.uni-leipzig.de

Erhard Rahm
rahm@informatik.uni-leipzig.de


http://dbs.uni-leipzig.de

# Overview

- Introduction
- Benchmark objectives
- XMach-1 specification
  - Architecture
  - DB-structure / -population, operations, metrics
- Implementation
- Results

# Introduction

- Increasing usage of XML
  - standard interchange format (e-business)
  - improved web format (server side/client side)
  - native data format for applications
- Demand for XML capable databases
- Problem: impedance mismatch
  relational data model ⇔XML (semi-structured)
- Different architectures for XML databases
- Pros & Cons? ⇨Benchmark

# Benchmark objectives

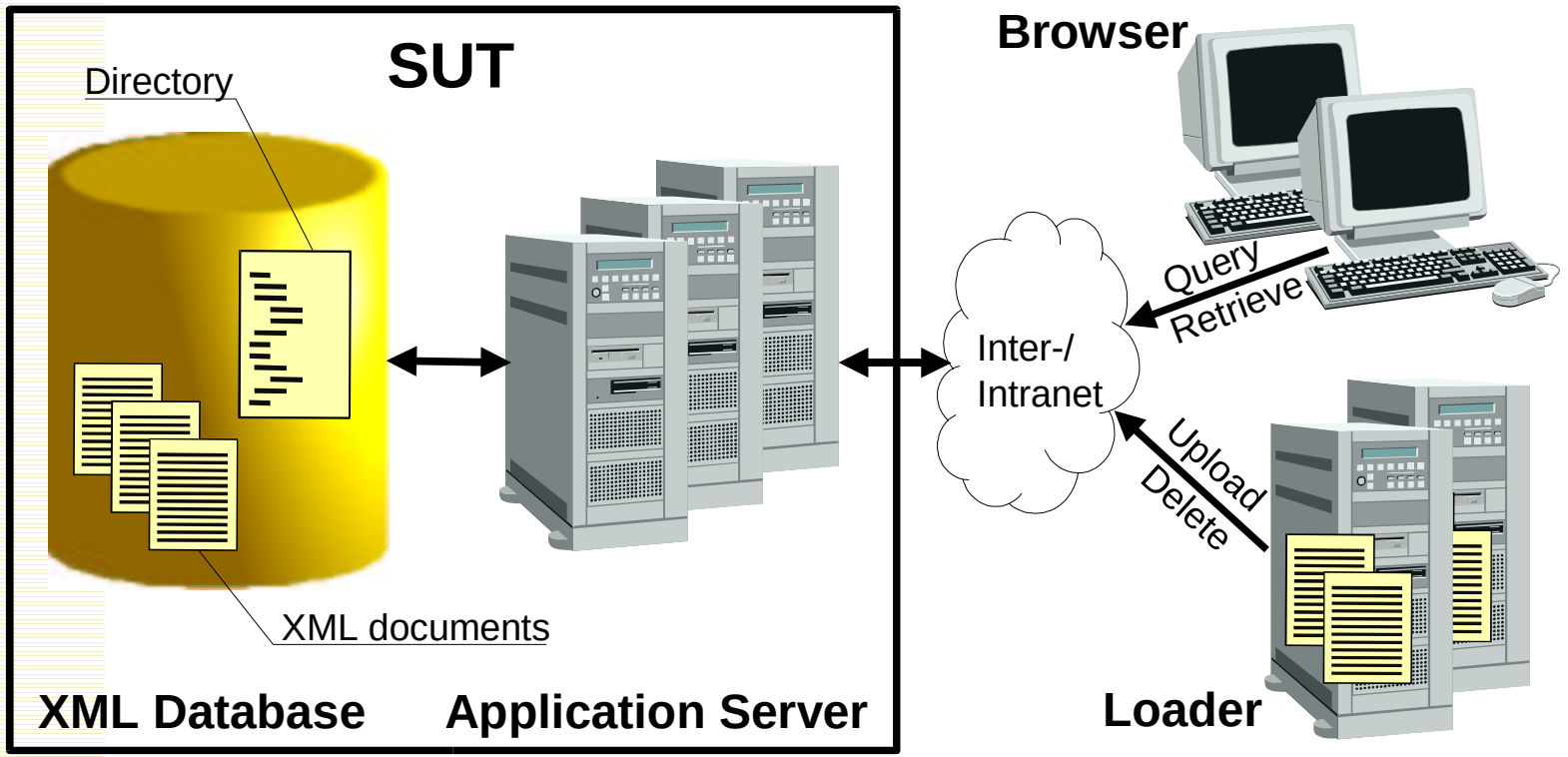
- Domain-specific
  - web-based application domain
- Store different kinds of XML documents
  - document-centric and data-centric
  - schema-less and schema-based
- Operations
  - XML-specific
  - database-specific
- Multi-user
- Throughput and response times

# Benchmark objectives (2)

- Portable
- Scaleable
  - database size
  - load volume
- Simple
  - only basic XML features
  - expect only basic functionality

# XMach-1 Specification

## Architecture



# XMach-1 Specification
## DB-structure / -population

- Database structure
  - documents (type: text)
    - synthetically generated (arbitrary number and sized)
    - different DTD‘s
    - non-uniform document structure and word frequency (Zipf)
  - document directory (type: data)
    - metadata (URL, update time, ...)
    - free element order
- Database population
  - initialize with 1.000-10.000.000 documents

# XMach-1 Specification

## DB-structure / -population (2)

```
<directory>
  <host name="com">
    <host name="test-company">
      <host name="www">
        <path name="products">
          <path name="overview.xml”>
            <doc_infodoc_id="2"
              loader="robot1"
              insert_time="20000110152530"
              update_time="20010217134500"/>
          </path>
        </path>
      </host>
      <host name="support">
        <path name="help.xml">
          <doc_infodoc_id="3" ... />
        </path>
      </host>
    </host>
  </host>
</directory>
```

```
<documentXX  author="Flo Tilla"  doc_id="d1">
  <titleXX>Language prepare</titleXX>
  <chapterXX  id="c1">
    <author>Jackquelin Jacoba</author>
    <headXX>Scheme units</headXX>
    <sectionXX  id="s1">
      <headXX> One intended spiritual</headXX>
      <paragraph>
        Said in seriously.<link  xlink:href="d2" />
        Federal at check. Mother image tanks of.
      </paragraph>
      <sectionXX  id="s2">
        <headXX>Satisfy west</headXX>
        <paragraph>...</paragraph>
      </sectionXX>
    </sectionXX>
  </chapterXX>
  <chapterXX  id="c2">
    ...
  </chapterXX>
</documentXX>
```

# XMach-1 Specification
## Operations

- Queries

1. Get document with given URL.
2. Get doc_id from documents containing a given phrase in paragraph elements.
3. Start with first chapter element and recursively follow first section element. Return last section element.
4. Return flat list of head elements which are children of section elements.
5. Get all document names below a given URL fragment.
6. Get doc_id and id of parent element of author element with given content.
7. Get doc_id from documents which are referenced by at least X other documents.
8. Get doc_id from the last 100 updated documents having an author attribute.

# XMach-1 Specification
## Operations

Q1 *Description:* Get document with given URL.
*Parameter:* URL = /ahost1.bhost2.chost3/001_loader1.xml

```
LET $a := /directory/host[@name="chost3"]/host[@name="bhost2"]/host[@name="ahost1"]/
          path[@name="001_loader1.xml"]/doc_info/@doc_id,
    $b := /*[@doc_id = $a]
RETURN $b
```

Q7 *Description:* Get doc_id from documents which are referenced by at least X other documents.

```
FOR  $refId IN distinct(/*//link/@href)
LET   $refDocs := distinct(/*[.//link/@href=$refID]/@doc_id)
WHERE count($refDocs) >= X
RETURN
   <docid>{string($refID)}</docid>
```

# XMach-1 Specification
## Operations (2)

- Data manipulation
  1. Insert new document.
  2. Delete a document.
  3. Update name and update_time attribute for a document.
- Operation mix with fixed operation shares and response time restrictions per operation
  - main operation: query 1 (30%)

# XMach-1 Specification
## Performance metrics

- Metrics: XML-Queries per second (Xqps)
- Two versions:
  - Xqps (with schema)
  - $Xqps_{sl}$ (schema-less)
- Think time restriction for clients
- Result report:
  - database size, population time
  - throughput + number of clients, response times
  - hardware, software

# Implementation

- Reference implementation

  http://dbs.uni-leipzig.de/en/projekte/XML/XmlBenchmarking.html

- Current evaluation
  - Native XML databases
- Planned evaluation
  - RDBMS with XML extension

# Results

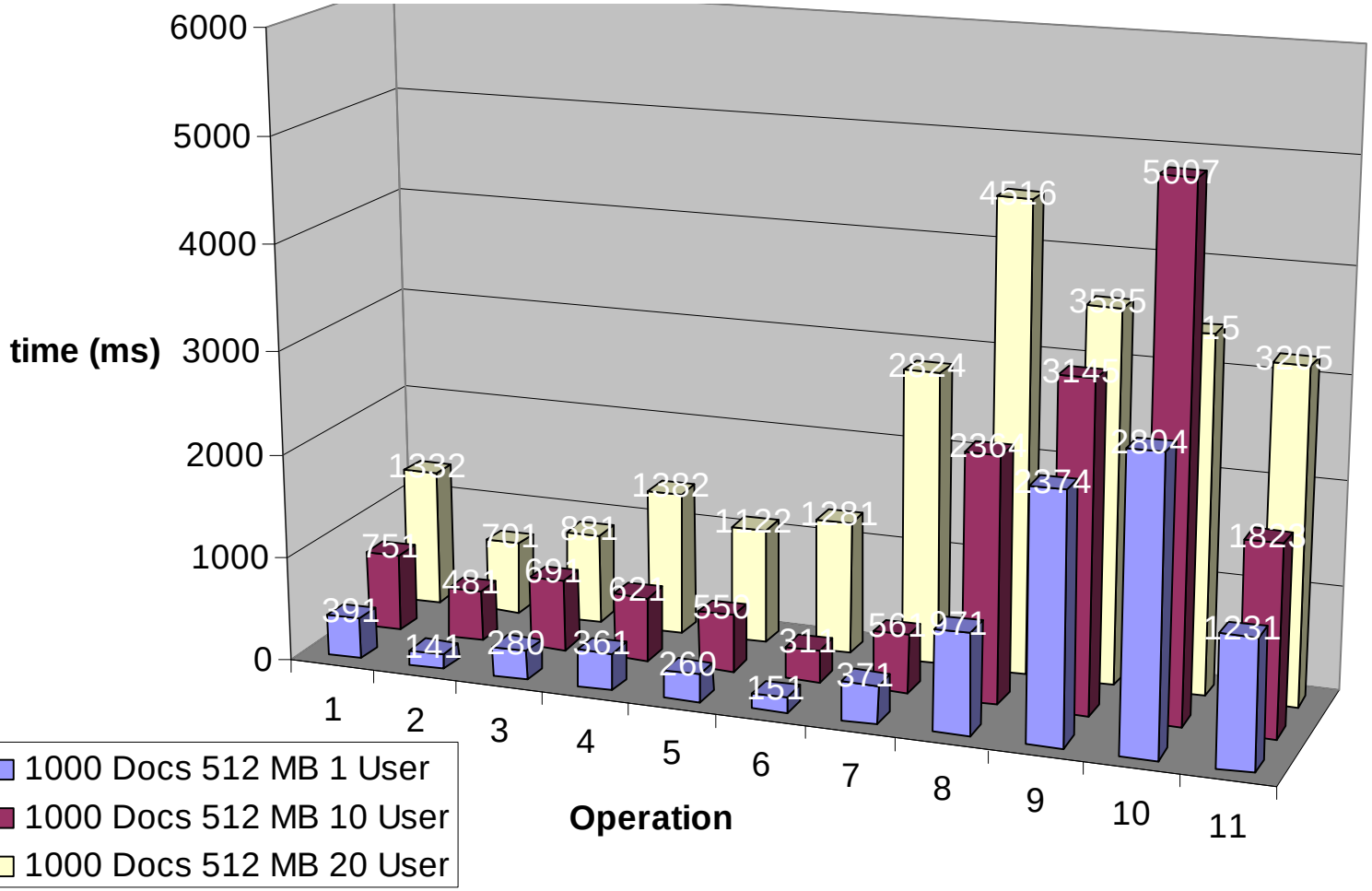
- Hardware/Operating System
  - Server: Intel PIII 800 MHz, 256/512 MB, 1 HDD, Windows 2000
  - Client: Sun Ultra 10, 440 MHz, 256 MB, 1 HDD, Solaris
  - Network: 10 Mbit Ethernet
- Database size:
  - 1,000 documents (Ø size 15,7 kB, Ø 128 elements); 41 DTDs
  - 10,000 documents (Ø size 15,3 kB, Ø 124 elements); 695 DTDs

# Results
## Database population

| #docs | | X-Hive 1.1.2 + Inktomi 4.1 | | | | NX1 | NX3 | | | | NX4 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | main memory (MB) | 256 | | 512 | | 256 | 256 | | 512 | | 256 |
| 1,000 | size d+i (MB) | 206.3 | | | | 80.1 | 96 | | | | 58.1 |
| | size data \| index (MB) | 145.6 | 60.7 | 145.6 | 60.7 | | | | | | |
| | time d+i (sec) | 616 | | 503 | | 330 | 692 | | 619 | | 340 |
| | time data \| index (sec) | 88 | 528 | 87 | 416 | | 324 | 368 | 322 | 297 | |
| 10,000 | size d+i (MB) | 927.6 | | | | 883.2 | | | | | 491 |
| | size data \| index (MB) | 560.5 | 367.1 | 560.5 | 367.1 | | | | | | |
| | time d+i (sec) | 10,803 | | 7,096 | | 4,969 | | | | | 13,786 |
| | time data \| index (sec) | 971 | 9832 | 924 | 6,172 | | | | | | |

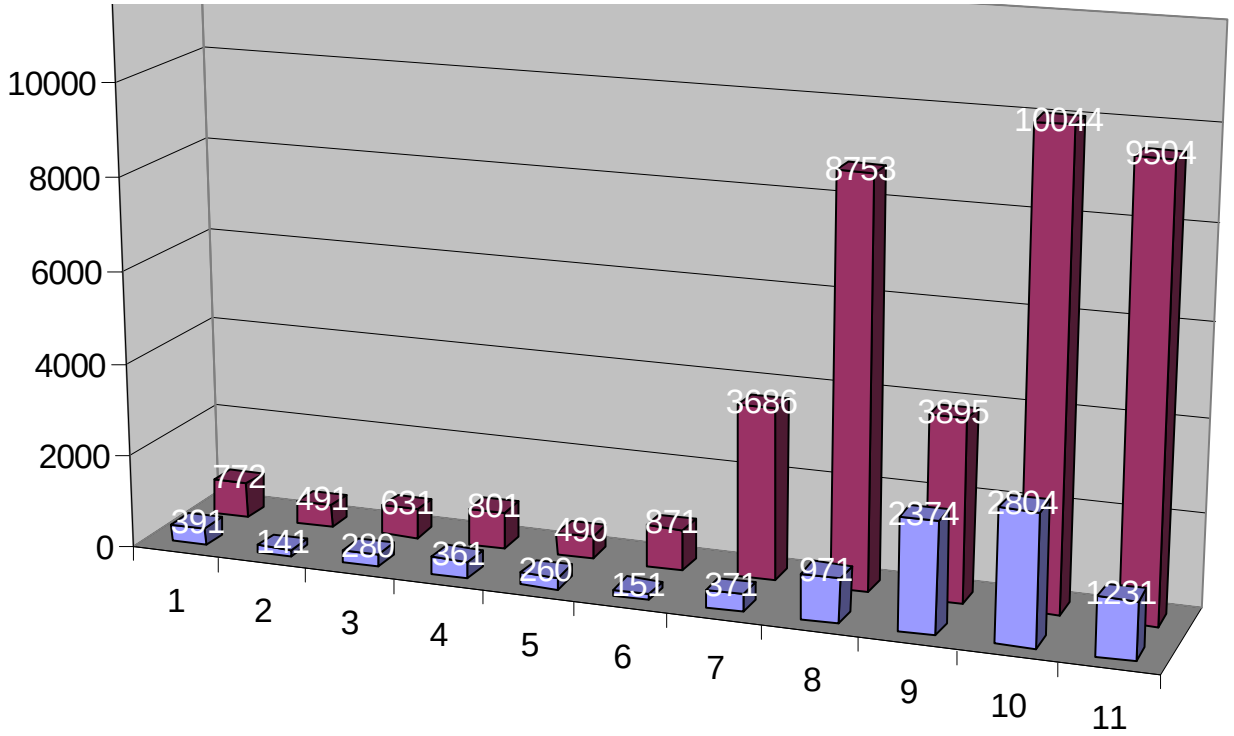
# Results

## Response times X-Hive/DB



# Results

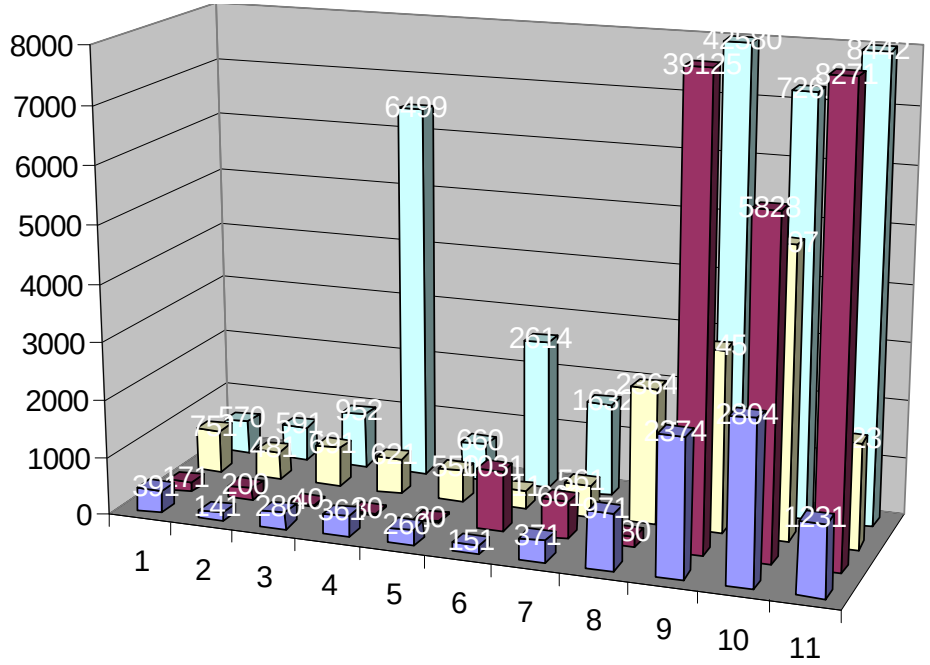
## Response times X-Hive/DB




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1000 Docs 512 MB 1 User | 391 | 141 | 280 | 361 | 260 | 151 | 371 | 971 | 2374 | 2804 | 1231 |
| 10000 Docs 512 MB 1 User | 772 | 491 | 631 | 801 | 490 | 871 | 3686 | 8753 | 3895 | 10044 | 9504 |

# Results

## Response times X-Hive/DB vs. NX3




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| X-Hive/DB 1000/1 | 391 | 141 | 280 | 361 | 260 | 151 | 371 | 971 | 2374 | 2804 | 1231 |
| NX3 1000/1 | 171 | 200 | 40 | 30 | 20 | 1031 | 661 | 180 | 39125 | 5828 | 8271 |
| X-Hive/DB 1000/10 | 751 | 481 | 691 | 621 | 550 | 311 | 561 | 2364 | 3145 | 5007 | 1823 |
| NX3 1000/10 | 570 | 591 | 952 | 6499 | 660 | 2614 | 1632 | 1162 | 42580 | 7261 | 8442 |

# Summary

- first multi-user benchmark for XML data management
- considers data- and document-centric aspects
- applicable for different architectures
- reference implementation available
- evaluation of different systems in progress
- little automatic optimization of XPath queries
- good performance only with direct index usage and direct accessing DOM structure
- weak point: text indexing

# Further Information

- T. Böhme, E. Rahm: XMach-1: A Benchmark for XML Data Management. Proc. 9. BTW-Conference, Springer, Oldenburg, March 2001
- http://dbs.uni-leipzig.de/en/projekte/XML/XmlBenchmarking.html