Comparison of Native XML Databases and Experimenting with INEX

Petr Kolář and Pavel Loupal

Dept. of Computer Science and Engineering
FEE, Czech Technical University
Karlovo náměstí 13, 121 35 Praha 2
Czech Republic

kolarp3@fel.cvut.cz, loupalp@fel.cvut.cz

DATESO 2006
Introduction – main goals

- summarize and compare approaches of design and architecture of native XML databases
- utilize the INEX dataset in several open source database systems (in this case only eXist and Apache Xindice)
- basic performance comparison outlined as a basis for discussion about suitability for particular database system
XML DB Products

- XML-Enabled Products
- Native XML Products
- Hybrid Products
### Some of NXD

- **Open-source**

<table>
<thead>
<tr>
<th>Product</th>
<th>Developer</th>
<th>DB Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berkeley DB XML, dbXML, eXist, ozone</td>
<td>Sleepycat Software, dbXML Group, Wolfgang Meier, ozone-db.org</td>
<td>Key-value, Proprietary, Relational, Object-oriented, Proprietary</td>
</tr>
<tr>
<td>Sedna XML DBMS, Timber, Xindice</td>
<td>ISP RAS MODIS, University of Michigan (non-commercial only), Apache Software Foundation</td>
<td>Proprietary, Shore, Berkeley DB, Proprietary (Model-based)</td>
</tr>
</tbody>
</table>

- **Commercial**

<table>
<thead>
<tr>
<th>Product</th>
<th>Developer</th>
<th>DB Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Birdstep RDM XML, eXtc, Ipedo, Natix, Neocore XMS, Tamino, X-Hive/DB, XStreamDB Native XML Database, Xyleme Zone Server</td>
<td>Birdstep, M/Gateway Developments Ltd., Ipedo, data ex machina, Xpriori, Software AG, X-Hive Corporation, Bluestream Corp., Xyleme SA</td>
<td>Object-oriented, Post-relational, Proprietary, File system(?), Proprietary, Proprietary (+ODBC), Proprietary. (+JDBC), Proprietary (Model-based), Proprietary</td>
</tr>
</tbody>
</table>

*NXD‡ Open-source‡ Commercial*
INEX dataset

- INitiative for the Evaluation of XML retrieval
- INEX data set (we use version 1.4) has 536MB of XML data. It is exactly 12,107 articles from 6 IEEE transactions and 12 journals from years 1995 to 2002
- In average each article contains 1,532 XML nodes
- The average depth of node is 6.9

http://inex.is.informatik.uni-duisburg.de/
eXist XML database version 1.0-dev-20060124

- Developed in Java, open source
- Supported Platforms: Platform independent
- Data Storage: B+-trees and paged files. Document nodes are stored in a persistent DOM—No support for binary files
- Transaction Support: No
- Authorization: Unix like, permissions at collection and document level
- XML Standards that are supported:
  - XPath/XQuery through XQuery engine
  - XUpdate
  - XInclude/XPointers
  - API: XML:DB
- Comes with great client GUI interface
- Types of indexes: Structural, Fulltext, Range

http://exist.sourceforge.net/
Xindice

- Xindice XML database version 1.0 (birthday)
  - Developed in Java, open source
  - Supported Platforms: Platform independent
  - Data Storage: Natively as indexed text files.
    - Collections as directories on file system
    - Documents in a collection as compressed text files (.tbl files); Hoffman codes.
  - No support for binary files
  - Transaction Support: No
  - Authorization: No support
  - Supported XML Standards:
    - XPath
    - XUpdate
    - AutoLinking
    - API: XML:DB, command line,
  - Unsupported XML standards: Xpointers, XQL, XQuery
  - No GUI available

http://xml.apache.org/xindice/
# Xindice vs. Exist

<table>
<thead>
<tr>
<th>Feature</th>
<th>eXist</th>
<th>Xindice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>Java</td>
<td>Java</td>
</tr>
<tr>
<td>Data storage</td>
<td>B+-trees and paged files. Persistent DOM</td>
<td>Natively as indexed text files, Hoffman codes</td>
</tr>
<tr>
<td>Binary files</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Transaction Support</td>
<td>Unix like, permissions at collection and document level</td>
<td>No</td>
</tr>
<tr>
<td>Authorization</td>
<td>No</td>
<td>No Support</td>
</tr>
<tr>
<td>Supported Standards</td>
<td>XPath/XQuery, XUpdate, Xinclude/XPointer</td>
<td>XPath, XUpdate, AutoLinking</td>
</tr>
<tr>
<td>APIs</td>
<td>XML:DB</td>
<td>XML:DB, command line</td>
</tr>
<tr>
<td>Client GUI</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Indices</td>
<td>Structural, Fulltext, Range</td>
<td>No</td>
</tr>
</tbody>
</table>
Experiment

- We prepared set of XPath queries in following categories:
  - Selecting nodes (i.e. /article/fm/hdr/hdr1/crt/issn)
  - Predicates (i.e. /article/bdy/sec[last() - 1])
  - Selecting Unknown Nodes (i.e. /*/*[@*])
  - Selecting Several Paths (i.e. //article/fm/hdr | //article/bdy/sec)

- We measured time needed to perform the each prepared query on Xindice and Exist on the same hardware
# Results 1

<table>
<thead>
<tr>
<th>No.</th>
<th>Query</th>
<th>Records retrieved</th>
<th>Query duration time [s]</th>
<th>XeXist</th>
<th>Xindice</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>/article</td>
<td>12104</td>
<td>1,3</td>
<td></td>
<td>230</td>
</tr>
<tr>
<td>2</td>
<td>/article/fm/hdr/hdr1/crt/issn</td>
<td>11666</td>
<td>2,2</td>
<td>98</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>//issn</td>
<td>11666</td>
<td>1,3</td>
<td>447</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>/article/bdy/sec[1]</td>
<td>11955</td>
<td>1,9</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>/article/bdy/sec[last()]</td>
<td>11955</td>
<td>5,6</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>/article/bdy/sec[last() - 1]</td>
<td>11019</td>
<td>5,8</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>/article/bdy/sec[position() &lt; 3]</td>
<td>22974</td>
<td>8,1</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>//sec[@type]</td>
<td>868</td>
<td>1,0 more than 10 min</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>//sec/p/ ref[@type = 'bib']</td>
<td>108496</td>
<td>81,3</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>/article/fm/hdr/hdr2/pdt[yr = '1995']</td>
<td>1623</td>
<td>2,6</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>/article/fm/hdr/hdr2/pdt[yr = '1995' and mo = 'Spring']</td>
<td>72</td>
<td>4,0</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>/article/*</td>
<td>58472</td>
<td>164,3</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>/ * / * [@*]</td>
<td>49</td>
<td>352,0</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>//fig[@*]</td>
<td>52857</td>
<td>70,6</td>
<td>NA</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>//article/fm/hdr</td>
<td>//article/bdy/sec</td>
<td>77487</td>
<td>8,6</td>
<td>NA</td>
</tr>
<tr>
<td>16</td>
<td>//article/fm/hdr/hdr1</td>
<td>//article/fm/hdr/hdr2</td>
<td>24208</td>
<td>3,8</td>
<td>NA</td>
</tr>
</tbody>
</table>
Results 2

- The time needed to load INEX data set into database:
  - 25 minutes for Xindice
  - 97 minutes for eXist

- The data on filesystem took:
  - 600 MB for Xindice
  - 1300 MB for eXist

- Our hardware configuration was based on a personal computer with Intel Celeron 1.7 Ghz processor, 512MB RAM and Windows XP(SP2) operating system

- INEX XML data set in version 2003 (1.4)
Summary

- Xindice has totally failed in our experiments probably due to index malfunction (but Xindice looks like that Indexes are working)
- Most of XPath queries running over Xindice returned an empty result set.
- On the contrary, eXist showed much better behavior.
- Automatically generated structural index in eXist that is very efficient
- eXist has also an user friendly GUI for both database management and ad-hoc query processing
Conclusion 1

- The aim of our experiment was in principle not successful
- We were not able to import the INEX dataset into all proposed native XML databases
- Our results show that for further experiments we should consider only the eXist database
- Xindice can be used just as an example of a basic native XML database, for large data set is not usable
  - At this moment is available Xindice Version 1.1b4
Conclusion 2

- It is needed to perform further comparisons among other native XML databases
- Also, we plan to add some of non-native (or hybrid) XML databases.

The end of the poster presentation