A Database Backend for OWL

October 24, 2009

Jörg Henß
Joachim Kleb
Stephan Grimm
Jürgen Bock

FZI Research Center for Information Technology
at the University of Karlsruhe
Germany
Motivation

- Why do we need a Database Backend for OWL?
  - Scalable Storage
  - Fast Access
  - Transactions, Versioning, Permissions ...

- Why yet another Storage Layer?
  - Mostly triple-based Systems today
    - Cumbersome
  - Native support for OWL
    - Direct Manipulation

- Not our Focus: Reasoning
Native Approach

- By **native** we understand:
  - Mapping OWL language constructs one-to-one to storage layer

- Triple Structure
  - RDF-Store

- Axiomatic view
  - Restrictions, cardinalities
  - OWL Axioms act on **objects** not on **nodes**
  - An object model for OWL is required
Schema Representation

- **OWL as Objects and Axioms**
  - Classes, Individuals, etc.
  - Class Assertions, etc.

- **OWL-API as Model for OWL**
  - Java based API for OWL
  - Maintained by University of Manchester
  - OWL 2 ready
  - Protégé 4 is based upon

- **Use of Object-Relational mapping for persistence**
  - Stores object and axiom information in database
  - Several Mapping Strategies
Complex Classes

Restrictions

URI information

Conjunction/Disjunction

Complement

Enumerations
Evaluation

- Comparison with other database-based Systems:
  - IBM SOR
  - Jena SDB
  - Owlgres

- Ontologies with varying Complexity
  - FMA, wine9, LUBM

- Mixture of common tasks tested
  - Import and Load Time
  - Retrieval Queries
  - Assertion Queries
  - Axiom and Statistical Queries
Evaluation - Results

### Mixed Results
- Batch-Loading speeds up Import
- Strong Influence of Tbox Caching

### Memory Consumption much lower

---

### Table 1. Import and load time (ms).

<table>
<thead>
<tr>
<th></th>
<th>OWLDB</th>
<th>SDB</th>
<th>Owlgres</th>
<th>SOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import</td>
<td>314368</td>
<td>61484</td>
<td>123718</td>
<td>352323</td>
</tr>
<tr>
<td>Load</td>
<td>1579</td>
<td>953</td>
<td>47</td>
<td>51200</td>
</tr>
</tbody>
</table>

### Table 2. Retrieval Queries (ms).

<table>
<thead>
<tr>
<th></th>
<th>OWLDB</th>
<th>SDB</th>
<th>Owlgres</th>
<th>SOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annotation</td>
<td>14172</td>
<td>234</td>
<td>10</td>
<td>390</td>
</tr>
<tr>
<td>Class</td>
<td>3610</td>
<td>3828</td>
<td>125</td>
<td>360</td>
</tr>
</tbody>
</table>

### Table 3. Individual assertion retrieval time (ms).

<table>
<thead>
<tr>
<th></th>
<th>OWLDB</th>
<th>SDB</th>
<th>Owlgres</th>
<th>SOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individual</td>
<td>63</td>
<td>63</td>
<td>4468</td>
<td>125</td>
</tr>
<tr>
<td>Object Property</td>
<td>250</td>
<td>79</td>
<td>47</td>
<td>263</td>
</tr>
<tr>
<td>Data Property</td>
<td>31</td>
<td>15</td>
<td>3</td>
<td>47</td>
</tr>
</tbody>
</table>

### Table 4. Statistical query retrieval time (ms).

<table>
<thead>
<tr>
<th></th>
<th>OWLDB</th>
<th>SDB</th>
<th>Owlgres</th>
<th>SOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub Classes</td>
<td>52</td>
<td>1359</td>
<td>3</td>
<td>93</td>
</tr>
<tr>
<td>Inverse Properties</td>
<td>47</td>
<td>110</td>
<td>2</td>
<td>63</td>
</tr>
<tr>
<td>Transitive Properties</td>
<td>3</td>
<td>3</td>
<td>-</td>
<td>15</td>
</tr>
<tr>
<td>All Classes</td>
<td>31</td>
<td>329</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>All Individuals</td>
<td>1265</td>
<td>507524</td>
<td>203</td>
<td>188</td>
</tr>
</tbody>
</table>
Conclusion

- Novel Approach to Ontology Persistence
  - Object Relational Mapping

- Still Prototype
  - Further Optimisations required

- Next Steps:
  - Support OWL-evolution
  - Usage of full-text indices

- Download address
  - http://owldb.sourceforge.net/

- Part of the German Theseus Research Project