Implementation Road Map

Group:
Hanh & Tinh

19/04/2004
Content

- Implementation Steps
- Used Tools
- RDQL
- Referential Projects
Implementation Steps

- Email messages feeding:
  - Writing a Java module to get email messages from POP3/IMAP servers.

- Parsing email messages:
  - Write a Java module to parse email messages’ header information to RDF data

- Data storing: Store data into
  - RDBMS
  - RDF data

- Data Querying:
  - Write a module using RDQL to query information from RDF Data
Used Tools

- IDE platforms:
  - JBuilder: a pure-Java environment for application development
  - Eclipse Platform: used to create apps as diverse as websites, embedded Java programs, C++ programs and Java Beans.

- RDF parser:
  - Jena: allows us to parse, create and search RDF models
    - Jena is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS and OWL, including a rule-based inference engine.
  - The Jena Framework includes:
    - A RDF API
    - Reading and writing RDF in RDF/XML...
    - An OWL API
    - In-memory and persistent storage
    - RDQL – a query language for RDF
Used Tools

- **RDF parser:**
  - **Joseki:**
    - Joseki is a server for publishing RDF models on the web. Models have URLs and they can be accessed by query using HTTP GET. Joseki is part of the [Jena](http://jena.sourceforge.net) RDF toolkit.
    - Joseki provides a coarse-grained [RDF WebAPI](http://jena.sourceforge.net) that is based on extracting a subgraph from the published RDF. The extracted RDF can then be processed locally with the fine-grained API provided by Jena.

- **RDBMS:**
  - MySQL
  - PostgreSQL

- **Ontology editor:**
  - Protégé

- **Reference:**
  - Projects in [www.sourceforge.net](http://www.sourceforge.net)
RDQL
A Query Language for RDF

- **Syntax**
  ```
  SELECT variables listing
  FROM rdf documents
  WHERE patterns
  AND filter expressions
  USING prefix declaration
  ```

- **SELECT clause**
  defines a list of variables to be return. Each variable is introduced by a question mark (?)
  Example: `SELECT ?name, ?email, ?tel_number`

- **FROM clause**
  specifies the path or URL of RDF document to be queried. Otherwise, RDF document can be specified on command line
WHERE clause:
indicates a list of triple patterns which have to be matched by each valid query result set.
All patterns representing a RDF statement have the form (subject, predicate, object) where subject, predicate, and object can either be a \(<\text{URI}>\) or a \(?\text{var}. The object can moreover be a “literal”.

Example:
WHERE (?resource, <http://www.w3.org/2001/vcard-rdf/3.0/EMAIL>, “tinh@gmx.at”)

AND clause:
Is boolean expressions over values of URIs and literals.
Example:
AND ?name EQ “Nguyen Huu Tinh”, ?age <=30
RDQL (3)

- **USING clause:**
  RDQL provides a way to shorten the length of URIs (used in `WHERE` and `AND` clauses) by defining a string prefix.
  Every prefix is defined in USING clause as following example:
  
  ```
  WHERE (?resource, <vCard:EMAIL>, "tinh@gmx.at")
  USING vCard FOR <http://www.w3.org/2001/vcard-rdf/3.0/>
  ```

- **An Example:**
  ```
  SELECT ?resource, ?givenName
  WHERE (?resource, <vCard:N>, ?z),
  (?z, <vCard:Given> ?givenName)
  USING vCard FOR <http://www.w3.org/2001/vcard-rdf/3.0#>
  ```

(see RDF document graph... next slide)
RDQL (4)

Executing this query, with command line application:
```
java jena.rdfquery --data test.rdf --query test
```

Result:
```
<table>
<thead>
<tr>
<th>resource</th>
<th>givenName</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://somewhere/SarahJones/">http://somewhere/SarahJones/</a></td>
<td>&quot;Sarah&quot;</td>
</tr>
<tr>
<td><a href="http://somewhere/MattJones/">http://somewhere/MattJones/</a></td>
<td>&quot;Matthew&quot;</td>
</tr>
</tbody>
</table>
```
Using RDQL from Java

- **Key Classes**

  All the important classes are in package `com.hp.hpl.jena.rdf.rdql`; the package `com.hp.hpl.jena.rdf.rdql.parser` contains the parser for the concrete syntax.

  - **Query** – the query itself, including access to parser
  - **QueryExecution** – the execution algorithm interface
  - **QueryEngine** – the local execution algorithm
  - **QueryResults** – the results iterator
  - **ResultBinding** – one collection of variable bindings
Using RDQL from Java (2)

- **Java Code**
  A query is created by passing a string to the Query class. If the source is not specified within the query, then the source must then be given, usually by passing a model to the query object, or specifying the URL of a model.

```java
String queryString = "SELECT ....." ;
Query query = new Query(queryString);

// Need to set the source if the query does not.
query.setSource(model);
QueryExecution qe = new QueryEngine(query);

QueryResults results = qe.exec();
for ( Iterator iter = results ; iter.hasNext() ; )
{
    ResultBinding res = (ResultBinding)iter.next() ;
    ... process result here ...
}
results.close() ;
```
import com.hp.hpl.jena.rdf.model.*;
import com.hp.hpl.jena.riot.*;
import java.util.*;
import java.io.*;

public class rdql_test {
    static public void main(String[] argv) {
        try {
            Model model = ModelFactory.createDefaultModel();
            model.read(new FileInputStream("test.rdf"), "http://nowhere/", "RDF/XML") ;

            String queryString = "SELECT ?resource, ?givenName " +
                                 "WHERE (?resource, <vCard:N>, ?z), (?z, <vCard:Given>, ?givenName)" +
                                 "USING vCard FOR <http://www.w3.org/2001/vcard-rdf/3.0#>";

            Query query = new Query(queryString);
            query.setSource(model);
            QueryExecution qe = new QueryEngine(query);

            QueryResults results = qe.exec();
            for ( Iterator iter = results ; iter.hasNext() ; ) { 
                ResultBinding res = (ResultBinding)iter.next();
                Object rs = res.get("resource") ;
                Object gname = res.get("givenName") ;
                System.out.println("rs = "+rs+"   gname = "+gname) ;
            }
            results.close();
        } catch (Exception ex) {
            System.err.println("Exception: "+ex);
            ex.printStackTrace(System.err);
        }
    }
}
Referential Projects

- Sesame
  - http://sesame.administrator.nl
  - http://www.openrdf.org
- MyLifeBits
- ePerson
- Eclipse
  - http://www.eclipse.org/
- Jena and Joseki
  - http://jena.sourceforge.net/
  - http://www.joseki.org/
Thank you!!!