University of Bologna

Department of Computer Science

Advancement on a Query Language for Metadata

Authors:

Ferruccio Guidi

Irene Schena

fguidi@cs.unibo.it, schena@cs.unibo.it

WHAT DO WE WANT?

QUERIED INFORMATION

- A Semantic Web of metadata on mathematical objects

DESIGN GOALS

- Syntax and Semantics abstracting from the...
  - representation *format* of the queried (meta)data (i.e. XML)
  - specification of *RDF* but supporting its characteristics that are independence, interchange and scalability.
- Features designed according to the following approach:
  - objects are resources,
  - statements about objects are relations between resources.

fguidi@cs.unibo.it, schena@cs.unibo.it

W3C REQUIREMENTS

An RDF-QL should...

- semantically query models and schemas, since inferring from schemas adds expressive power to the query language;
- abstract from the RDF syntax specification
- provide facilities for:
  - path traversal of property values: reach compound values and travelling through the RDF graph model;
  - subsumption between classes/properties;
  - classification of resources based on their properties;
  - inverse inferences
  - generalization/specialization relations between properties

AN RDF-QL VS OTHER QUERY LANGUAGES

Why other QL's fail to capture the semantics of RDF?

The RDF modeling primitives are substantially different from those defined in object or relational database models.

- Classes do not define objects or relation types: an instance of a class is just a resource with a URI, without any value or state.
- Resources may belong to different classes not necessarily pairwise related by specialization. The instances of a class may have quite different properties associated to them, while there may be no class on which the union of these properties is defined.
- Properties may also be refined by respecting a minimal set of constraints, i.e. domain and range compatibilities.

PROPOSAL

General language architecture.

- A resource is denoted by a URI reference (URI with opt. ID).
- Each resource comes with named attributes holding multiple string values and specifying its relations with their values.
- These attributes are grouped into *attribute sets* containing the ones that are meaningful in the same context.
- A relation between resources may specify an attribute set for its targets with the aim of providing additional information.
- A query gives a list of resources with their attribute sets.
- Queries have both a textual syntax and an XML syntax.

PROPOSAL

Main types of queries (textual syntax).

- **reference** `<list-of-quoted-constant-strings>`
  a list of resources given explicitly, without attributes.
- **pattern** `<quoted-constant-string>`
  All available resources, each without attributes, whose URI reference match a given POSIX 1003.2 regular expression.
- **relation** `<constant>` `<query>` **attributes** `<list-of-names>`
  Every resource in the specified relation with some resource returned by the given query. Each of these with the attribute set defined by the relation (if any) whose elements can be renamed.
- `<query>` **union** `<query>`
  The attribute sets of common resources are put together.

PROPOSAL

Main types of queries (continued).

- `<query>` **intersect** `<query>`
  In this case the attributed sets are "multiplied".
- **select** `<name>` **in** `<query>` **where** `<condition>`
  The resources returned by `<query>` that meet `<condition>` where `<name>` is instantiated with their URI references.

Main conditional operators.

Conditions contain names for resources and attributes, and refer to attribute sets using specific operators like the one below.

- **ex** `<condition>` is true if there is a pool of attribute sets, one for each resource in `<condition>`, that satisfies the condition itself.

fguidi@cs.unibo.it, schena@cs.unibo.it

IMPLEMENTATION AND TESTING

Implementation.

We are working on the implementation of an interpreter based on the PostgreSQL DBMS which is the best we have tested up to now

Currently we have metadata information (stored in RDF format) on:

- document dependences: described as direct or inverse, with a "position" attribute holding mathematically meaningful values;
- general Dublin Core meta-information (manually inserted).

The database contains 416700 statements on 15600 documents.

The automated process that generates the meta-information on document dependences and builds the database, takes about 7 hours.

IMPLEMENTATION AND TESTING

Testing.

We have a generator of queries about theorems directly referencing a given set of resources with constraints on the "position" attribute.

Here is a performance test involving 177 generated queries:

results	**time/results (mean)**	**time/results (variance)**
$0 \rightarrow 9$	$0.92s$	$0.75s^2$
$10 \rightarrow 74$	$0.45s$	$0.13s^2$
$75 \rightarrow 104$	$0.55s$	$0.01s^2$
$0 \rightarrow 104$	$0.81s$	$0.61s^2$

The major time waste occurs when the interpreter queries the DBMS.

fguidi@cs.unibo.it, schena@cs.unibo.it

IMPLEMENTATION AND TESTING

What a generated query looks like?

```
let %universe be
  reference "cic:/Coq/Init/Datatypes/nat.ind#1/1/1",
            "cic:/Coq/Init/Datatypes/nat.ind#1/1/2",
            "cic:/Coq/Init/Peano/mult.con",
            "cic:/Coq/Init/Peano/le.ind#1/1"
in
select uri0 in
      select ref0 in relation "use" reference "cic:/Coq/Init/Peano/le.ind#1/1"
         attributes $str0 where ex ref0.$str0 contains "mainconclusion"
  intersect
      select ref1 in relation "use" reference "cic:/Coq/Init/Peano/mult.con"
         attributes $str1 where ex ref1.$str1 contains "inconclusion"
  intersect
      select ref2 in relation "use" reference "cic:/Coq/Init/Datatypes/nat.ind#1/1/2"
         attributes $str2 where ex ref2.$str2 contains "inconclusion"
  intersect
      select ref3 in relation "use" reference "cic:/Coq/Init/Datatypes/nat.ind#1/1/1"
         attributes $str3 where ex ref3.$str3 contains "inconclusion"
where
      select uri1 in relation "usedby" uri0 attributes $pos
         where ex (uri1.$pos contains "mainconclusion or uri1.$pos contains "mainconclusion")
   subset
      %universe
```

IMPLEMENTATION AND TESTING

What a generated query looks like?

The previous query was generated from the statement $2 * n \leq 2 * m$ where n and m belong to nat and 2 is the double successor of 0.

We ask for the theorems or definitions that are candidates to prove it because their types depend on the same resources.

The HELM resources used by the statement are:

$m \leq n$	cic:/Coq/Init/Peano/le.ind#1/1
$m * n$	cic:/Coq/Init/Peano/mult.con
$m + 1$	cic:/Coq/Init/Datatypes/nat.ind#1/1/2
0	cic:/Coq/Init/Datatypes/nat.ind#1/1/1

STATE OF THE ART

RDF Query products (an incomplete list)

- RDFDB: a scalable, database using a query language like SQL.
- Redland: a library providing a high-level interface for RDF.
- RDFS Explorer: an interpreter and query engine able to query semantically the interpreted data with a Prolog engine.
- Squish: a query engine which can take SQL-like query strings.
- VRP and RQL: a platform satisfy all requirements of a RDF database. VRP validates Statements against a Schema. RQL is a query language for RDF proposed by ICS-FORTH.
- GINF: contains a module for interpreting and serializing RDF streams transferring data between heterogeneous applications.