Querying an RDF Registry

Norman Gray
VO-TECH / Uni. Leicester / Uni. Glasgow, UK

IVOA Interop, Beijing, 2007 May 15

overview

If you convert the Registry to RDF, what do you get?

- Yet another vaguely SQL-like query language.
- All the implications of a given registry entry.
- Shareable overlaid structure (bookmarks, recommendations, saved queries).
- Flexible semantic store.

acronym intro

- All the world is *triples*, consisting of *resources* named by URIs (`http:...` or `ivo:...` or `urn:example#Norman`)
- . . . which have *properties* whose *values* are resources or literals.
- RDF/RDFS/OWL describe these using `rdf:type`, `rdfs:subClassOf`, `owl:symmetricProperty`, and so on.

There is an analogy with XML Schemas, *but it is a loose one* -- they're not addressing the same problem. Same for O-O.

rdf/owl/semweb/sql/xml — respective strengths

RDF/OWL/reasoning now largely stable (though The Semantic Web will forever be Vision). Now engineering rather than CS.

Using the architectural principles which let HTML take over the internet. Very open and flexible; has existing powerful query language. Did I mention standards?

RDB to XML to RDF -- spectrum of strengths. XML is more natural than RDF where the information density is high, and the information regular or highly constrained; RDF/SW is natural for incomplete or ragged data.

rdf schemas give you reasoning

1. X is SecondaryEducationContentLevel
2. SecondaryEducationContentLevel is a subclass of SchoolContentLevel
3. thus X is SchoolContentLevel

1. Y (CurationDescription) `publisher` X
2. `publisher` hasInverse `publishes`
3. thus X `publishes` Y

Add transitive, functional & symmetric properties, subclass/subproperty relations, and you magnify what you say.

It's not the query language that's the win, here, but the fact that the reasoner can expand the set of assertions in your knowledgebase, by drawing all possible conclusions.

Plus you can add derived types

. . . and annotations

. . . and easy extension and versioning.

what does this look like for the registry?

- RDF Schema versions of XSchemas ConeSearch 1.0, SIA 1.0, TabularDB 0.3, VODataService 1.0, VORegistry 1.0, STC 1.30, and VOResource 1.0. VOResource written by hand (more idiomatic), the others generated automatically from the XSchema.
- XSLT transformations from v1.0 instances to RDF, generated from the XSchema.
- Works for all of Ray's 1.0 RM test instances.

db volume

Triplestores are for bulk instances, and trade off volume/speed vs. expressiveness: fast RDFS reasoning vs. slow OWL reasoning.

- Expressiveness: RDFS, OWL DLP, OWLIM, OWL Lite, OWL DL, SWRL, OWL Full.
- Jena: 10^4 + OWL-DL — boom! 3store: 10^4 + RDFS — whizz!
- OWLIM/TRREE: 10^8 triples for €1000/cpu in 2006.
- Hybrid solution: offline OWL reasoning 'compiled' to bulk RDFS assertions.

SPARQL:

```
prefix vor: <http://www.ivoa.net/xml/VOResource/v1.0#>
prefix sia: <http://www.ivoa.net/xml/SIA/v1.0#>

select ?r ?t
where {
 ?r vor:capability ?cap.
 ?cap [ sia:imageServiceType [ a sia:ImageServiceTypeAtlas ]].
 ?r vor:content [ vor:contentLevel [ a vor:ResearchContentLevel ] ].
 ?r vor:identifier [ vor:authorityID ?authid].
 FILTER REGEX(?authid, "\.ca$") .
 ?r vor:title ?t.
}
```

queries, ii

SPARQL, with user-defined classes:

```
prefix vor: <http://www.ivoa.net/xml/VOResource/v1.0#>
prefix sia: <http://www.ivoa.net/xml/SIA/v1.0#>
prefix me: <http://example.org/norman#>

select ?r
where {
  ?r a me:ResearchAtlas.
}
```

options

- Community resource or desktop resource?
- Lots of reasoning, or very little?
- Clever SPARQL or clever triplestore?
- Annotations?
- Community-specified types?
- Provide the service and the applications will come!