

Semtools - Bug #4906

Annotation query implementation (SPARQL)

03/29/2010 11:33 AM - ben leinfelder

Status:	Resolved	Start date:	03/29/2010
Priority:	Normal	Due date:	
Assignee:	ben leinfelder	% Done:	0%
Category:	SMS API	Estimated time:	0.00 hour
Target version:	Unspecified	Spent time:	0.00 hour
Bugzilla-Id:	4906		
Description			
Matt recommends materializing the Annotation and Ontology as instances in the OBOE model for running searches. I need to investigate how this can be done.			

History

#1 - 03/29/2010 01:24 PM - Shawn Bowers

I am not sure what it means here to materialize an annotation, but if it implies storing an annotation per se as an instance of an oboe model, this won't work. We can extract the measurement & observation "types" from an annotation, store these as subclasses of oboe classes, and then use this for indexing. To then use a language like SPARQL, we will need to store the ontology(ies) fully classified. This might be a good approach for answering "schema level" queries (i.e., as in the current morpho extensions). Other alternatives would be to use a relational db to store the index information and use this in combination with the ontologies.

#2 - 03/31/2010 07:52 AM - ben leinfelder

the current search implementation is simple and generous. some features that are desired:

-compound grouping conditions:

any[XYZ] vs. all[XYZ] vs. all[any[XY]any[Z]]

-negation

is[x] vs. isnot[X]

Shawn - can we talk about implementation ideas?

#3 - 03/31/2010 02:04 PM - ben leinfelder

I think we were saying that an every annotation would be an Annotation Instance (in some yet-undefined ontology that references the OBOE classes) and that you could use a query language (sparql) to find annotation instance matches based on the conditions. This would not deal with the data - only the "schema", as you called it.

So the short question: can we make an ontology and fill it with Annotation Instances?

#4 - 03/31/2010 02:14 PM - Shawn Bowers

(In reply to comment [#3](#))

I think we were saying that an every annotation would be an Annotation Instance (in some yet-undefined ontology that references the OBOE classes) and that you could use a query language (sparql) to find annotation instance matches based on the conditions. This would not deal with the data - only the "schema", as you called it.

So the short question: can we make an ontology and fill it with Annotation Instances?

What would the advantages be? What would the disadvantages be?

More importantly, why would we want an Annotation be be considered an "ontology"? I think conceptually it isn't meaningful, and so, this would only be so that we can take advantage of OWL/SPARQL technology. And it is unclear why we'd want to do this (would it be more efficient/increase the speed of queries?).

As I see it, we'd only really be using part of the annotation for this, which is specifically the "measurement and observation types" of the annotation. Extracting the "ontology" from this part of the annotation and using it as an index would make sense to me ... This wouldn't require creating a new ontology structure at all, we could simply use OBOE for this.

#5 - 03/31/2010 02:53 PM - ben leinfelder

Expressivity/simplicity?

Something that would allow us to retrieve a list of annotation [instances?/objects?] with a query like this:

```
SELECT annotation
WHERE entity = e1
AND (characteristic in (c1, c2) OR standard = s1)
AND protocol != p1
```

Right now we just loop through the entire list of annotations and inspect the Observations/Measurements one by one.

#6 - 03/31/2010 03:11 PM - Shawn Bowers

(In reply to comment [#5](#))

Expressivity/simplicity?

Something that would allow us to retrieve a list of annotation
[instances?/objects?] with a query like this:

```
SELECT annotation
WHERE entity = e1
AND (characteristic in (c1, c2) OR standard = s1)
AND protocol != p1
```

Right now we just loop through the entire list of annotations and inspect the
Observations/Measurements one by one.

Clearly, the way were are currently doing it is not scalable or efficient (since we're reading files into memory, and traversing data structures).
However, this looks more like an SQL query -- which if you simply want to store the structure of an annotation, would probably be much better (for
expressivity as well as efficiency).

#7 - 04/06/2010 04:19 PM - ben leinfelder

we are dropping this in favor of a relational-db approach:

http://bugzilla.ecoinformatics.org/show_bug.cgi?id=4922

#8 - 03/27/2013 02:28 PM - Redmine Admin

Original Bugzilla ID was 4906