MANCHESTER
1824

The University of Manchester

mcsb

The SBML Level 3 Annotation package

Allyson Lister, Neil Swainston, Dagmar Waltemath
et al.

Harmony 2011, New York, USA
20 April 2011

Background

- SBML does not contains semantics
- Annotations were introduced in Level 2
  - Also supported in Level 3 Core
- MIRIAM standard specify recommendations of how to use annotations
  - RDF
  - Triple: subject, predicate, object
  - “Tagging” SBML elements with MIRIAM URIs

Limitations of current annotations

- NO Statements about attributes
- NO Statements about statements
- NO Relations between statements
- NO Negative statements (still to solve...)
- SO... SBML Level 3 Annot package proposal
  - http://precedings.nature.com/documents/5610/version/1

Statements about attributes

```
<species id=”abc" metaid=”meta_abc” compartment=“cytoplasm”
   initialConcentration=“53.4”>
 <annotation>
  <rdf:RDF>
   <rdf:Description rdf:about=”#meta_abc”>
    <bqbiol:isDescribedBy>
     <rdf:Bag>
      <rdf:li rdf:resource="urn:miriam:pubmed:15387819"/>
     </rdf:Bag>
    </bqbiol:isDescribedBy>
   </rdf:Description>
  </rdf:RDF>
 </annotation>
</species>
```

Statements about attributes

- Use XPath to specify attribute that is the SUBJECT of the RDF triple:

```
<species metaid=”meta_abc” id=”abc" compartment=“cytoplasm”
   initialConcentration=“53.4”>
 <annotation>
  <rdf:RDF>
   <rdf:Description
   rdf:about=”xref://species[@metaid=‘meta_abc’]/@compartment”>
    ..
```

- May have to restrict the allowed XPath statements:

```
<rdf:Description rdf:about=”xref://species[7]/@compartment”>
```

Statements about statements

- Case study: who added an annotation?
  - Annotation of annotations

- Use RDF Reification
  - Existing standard of “chaining / linking” RDF statements
  - Apply rdf:ID to a triple statement
  - Use this rdf:ID as subject in subsequent statements

Statements about statements

- Other application: protein modifications
  - Cannot currently define “Protein X is modified by modification Y in position Z”
- With RDF Reification, we can link more complex statements by defining a blank node, A:
  - PROTEIN_X has MODIFICATION whose value is A
  - A has MODIFIER whose value is PHOSPHORYLATION
  - A has POSITION whose value is 237

Relations between statements

- Alternative annotations... OR

```
<species id=”glc" metaid=”meta_glc” name=“Glucose”>
 <annotation>
  <rdf:RDF>
   <rdf:Description rdf:about=”#meta_glc”>
    <bqbiol:is>
     <rdf:Bag>
      <rdf:li rdf:resource="urn:miriam:obo.chebi:CHEBI%3417234"/>
      <rdf:li rdf:resource="urn:miriam:kegg.compound:C00234"/>
     </rdf:Bag>
    </bqbiol:is>
   </rdf:Description>
  </rdf:RDF>
 </annotation>
</species>
```

Relations between statements

- Complexes... AND

```
<species id="Ca_calmodulin" metaid="cacam”>
 <annotation>
  <rdf:RDF>
   <rdf:Description rdf:about="#cacam”>
    <bqbiol:hasPart>
     <rdf:Bag>
      <rdf:li rdf:resource="urn:miriam:uniprot:P62158"/>
      <rdf:li rdf:resource="urn:miriam:kegg.compound:C00076"/>
     </rdf:Bag>
    </bqbiol:hasPart>
   </rdf:Description>
  </rdf:RDF>
 </annotation>
</species>
```

Relations between statements

- Complexes... AND

```
<species id="Ca_calmodulin" metaid="cacam”>
 <annotation>
  <rdf:RDF>
   <rdf:Description rdf:about="#cacam”>
    <bqbiol:hasPart>
     <rdf:Bag>
      <rdf:li rdf:resource="urn:miriam:uniprot:P62158"/>
      <rdf:li rdf:resource="urn:miriam:kegg.compound:C00076"/>
     </rdf:Bag>
    </bqbiol:hasPart>
   </rdf:Description>
  </rdf:RDF>
 </annotation>
</species>
```

Relation between statements

- RDF Containers and Collections describe groups

- RDF Containers describe an open set:

rdf:Bag	unordered set
rdf:Seq	ordered set
rdf:Alt	alternatives (“or”)

- RDF Collections describe a closed set:
  - rdf:List

Additional issues

- New predicates required?
- Previous example considered modifications
  - Currently no predicates regarding modifications supported by biomodels qualifiers
- Presumably, other requests...?
- Unify (combine?) qualifiers between SBML and BioPAX?

How should an libSBML API look?

- Discuss this afternoon?

How should an API look?

- Preferred option
  - (for me)

```
sbase.setRDF( String rdf )
String rdf = getRDF( Sbase sbase )
```

- Utterly unusable: we want to support non-RDF geeks

- (True) preferred approach: hide RDF

How should an API look?

- Simple (L2 and Core) case:

```
sbase.setStatement( String predicate,
  String object )
```

```
String[] predicates = getPredicates()
```

- May have to extend to deal with complex predicate “objects” such as vcards

How should an API look?

- Statements about attributes

```
sbase.setStatement( String attributeName,
String predicate, String object )
```

```
String[] attributeNames =
sbase.getExpectedAttributes()
```

- `object` as a struct of MIRIAM datatype and id?
  - resource not free text
  - e.g. `urn:miriam:kegg.compound` and `C00234`

How should an API look?

- Statements about statements

```
Statement statement = sbase.setStatement(
String predicate, String object )
```

- `Statement` struct of subject, predicate, object?

```
sbase.setStatement( Statement target,
String predicate, String object )
```

How should an API look?

- Relations between statements:

```
sbase.setStatements( String predicate,
ALTERNATIVES, String[] object )
```

rdf:Bag	UNORDERED_SET
rdf:Seq	ORDERED_SET
rdf:Alt	ALTERNATIVES
rdf:List	CLOSED_SET

How should an API look?

- Simple
- Hide RDF-ness but support all extensions in proposal
    - Use Raptor in the back-end but wrap it
- Future developments:
    - Link with MIRIAM to help with defining objects?
    - Define predicates in RDF schema?

Conclusion

- Extended Annotation package proposal in development
- Only intended to proposal extensions that will actually be used
- Please contribute if interested
  - Meet up this afternoon for riotous argument
  - neil.swainston@manchester.ac.uk

The SBML Level 3 Annotation package

Allyson Lister, Neil Swainston, Dagmar Waltemath
et al.

Harmony 2011, New York, USA
20 April 2011