Annotations

Goals

- Instance URIs can be both an *belv:AnnotationConceptScheme* and *belv:NamespaceConceptScheme* (i.e. <http://www.openbel.org/bel/namespace/mesh-diseases>)
- Annotation (and Namespace) domains will need URIs. For example the "Disease" domain might be <http://www.openbel.org/belv/DiseaseDomain> which is a subClass of <http://www.openbel.org/belv/ConceptDomain>
- BEL evidence annotations (e.g. within the Biological Context) will need to map annotation names to a URI. This URI maybe either be a type of *belv:AnnotationDefinition* or *belv:Domain*. BEL Evidence should provide a way to map these names to URIs (i.e. the *DEFINE ANNOTATION* records or format equivalent (XML, JSON)).
- Apply a type class to instances of *belv:AnnotationConcept*. For example the MESH Disease term ***meshd:D008579*** could be described as:

  ```
  meshd:D008579 rdf:type belv:DiseaseAnnotationConcept
  belv:DiseaseAnnotationConcept rdfs:subClassOf belv:AnnotationConcept
  ```
  - The concept types do not overlap between *belv:AnnotationConceptScheme* and *belv:NamespaceConceptScheme*.
  - Concept types will enable use of vocabularies like EFO containing different annotation types. Users may need to define their own sub classes of annotation types outside of belv, but these will be subClasses of AnnotationConcept.
- Custom annotations
  - Creating a new instance of *belv:AnnotationConceptScheme* will require an unused URI on import. A tabular file would be useful for creating a SKOS-based annotation definition.
  - We will need to create the class *belv:AnnotationDefinition* to group the possible annotation definitions *belv:AnnotationConceptScheme*, *belv:AnnotationList*, and *belv:AnnotationPattern*.
    - Will instances of *belv:AnnotationList* and *belv:AnnotationPattern* be blank nodes? Or should we require a URI to be attributed to all annotation definitions.
    - Can *belv:AnnotationConceptDomain* be a subclass of *AnnotationDefinition*?
  - The *belv:AnnotationList* could be defined using rdf:List.
  - We can go beyond a list and pattern by leveraging XSD types as the annotation definition's domain. For example, this could allow us to define a **p-value** annotation definition using xs:decimal.
- Annotation Definition schema:

  ```
  (schema)
  belv:AnnotationConceptSchema rdfs:subClassOf belv:AnnotationDefinition
  belv:AnnotationList rdfs:subClassOf belv:AnnotationDefinition
  belv:AnnotationPattern rdfs:subClassOf belv:AnnotationDefinition
  (instance data example)
  bel:namespace/meshd rdf:type belv:AnnotationConceptScheme
  ```
- Allow the creation of annotation list and pattern through the openbel-server API.
- Creation/Update for BEL Evidence should fail if a referenced annotation definition has not been previously defined. This is to have the user control new annotation definitions as a discrete step and avoid bad data in practice.
- If a domain uri is used to define an annotation, rooting order may need to be used to resolve ambiguous values (e.g., if "Cancer" can be mapped to both MESHD and DO, use DO)

Summary

Support defining/setting Annotations as a:

- AnnotationConceptScheme e.g., disease-ontology (by uri)
- list of values (rdf:List?)
- pattern (leverage xsd type)
- domain e.g., Disease

Data

(2015-02-04) - Created by Natalie Catlett.