libSBOLj 2.0: A Java Library to Support SBOL 2.0

Zhen Zhang[1], Tramy Nguyen[1], Meher Samineni[1], Nicholas Roehner[2], Goksel Misirli[3], Matthew Pocock[3], Ernst Oberortner[4], Jacob Beal[5], Kevin Clancy[6], Anil Wipat[3], Chris Myers[1]

[1]University of Utah
[2]Boston University
[3]Newcastle University
[4]DOE Joint Genome Institute
[5]Raytheon BBN Technologies
[6]ThermoFisher Scientific

SBOL Workshop
August 15, 2016

libSBOLj

- Crucial to the success of a standard is software infrastructure to support developers' integration of the standard within their tools.
- `libSBOLj` is a native Java implementation of the SBOL data structure, which provides an *application programmers interface* (API) for tool developers to interact with SBOL data objects.
- In addition, the library distribution includes detailed documentation for the class definitions and the methods provided by the API.
- Leveraging `libSBOLj` many software tools now support SBOL 1.1.
- `libSBOLj 2.0` will ease the adoption of SBOL 2.0 by tool developers.

SBOL 2.0 Data Model

- `libSBOLj 2.0` organizes all SBOL data within an *SBOL document*.
- Includes a list of each type of *top level* object: *collections*, *modules*, *components*, *sequences*, *models*, and *generic top level* objects.
- These lists are organized as hash maps to allow for easy search by their *unique reference identifiers* (URIs) and validation that they are distinct.
- Library includes methods for creating, updating, accessing, and removing these data objects, as well as, their child objects.

Creating an SBOLDocument

```
String prURI = "http://partsregistry.org";
SBOLDocument document = new SBOLDocument();
document.setDefaultURIprefix(prURI);
document.setComplete(true);
document.setCompliant(true);
document.setCreateDefaults(true);
document.setTypesInURIs(false);
```

- Default URI prefix - prefix to use when none provided to create method.
- Complete - ensure that all URI references point to valid SBOL objects.
- Compliant - ensure that all URIs in the document are compliant.
- Create defaults - implicitly create ComponentInstances as needed.
- Types in URIs - insert type in URIs between prefix and displayId.

Creating SBOL Data Objects

- SBOLDocument class includes create methods for each TopLevel object.
    - displayId only - URI prefix taken from default, no version
    - displayId, version - URI prefix taken from default
    - URIprefix, displayId, version
- All required fields are also parameters to these create methods.

$$\text{create}\langle\text{TopLevel}\rangle(\text{URIprefix, displayId, version, }\langle\text{required fields}\rangle)$$

```
ComponentDefinition TetR_promoter =
  document.createComponentDefinition("BBa_R0040", ComponentDefinition.DNA);
Sequence seq_187 = document.createSequence("seq_187",
  "tccctatcagtgatagagattgacatccctatcagtgatagagatactgagcac", Sequence.IUPAC_DNA);
```

Setting and Editing Optional Fields

- Methods to set/unset each optional field, as well as check if isSet.

```
TetR_promoter.setName("p_tetR");
TetR_promoter.setDescription("TetR␣repressible␣promoter");
if (TetR_promoter.isSetName()) {
  TetR_promoter.unsetName();
  TetR_promoter.setName("p(tetR)");
}
```

- Methods to add and remove URIs from lists, as well as to check if it contains a URI.

```
TetR_promoter.addRole(SequenceOntology.PROMOTER);
URI TetR_promoter_role2 = URI.create("http://identifiers.org/so/SO:0000613");
TetR_promoter.addRole(TetR_promoter_role2);
if (TetR_promoter.containsRole(TetR_promoter_role2)) {
  TetR_promoter.removeRole(TetR_promoter_role2);
}
```

- Methods to clear, get, and set lists of URIs.

```
TetR_promoter.clearRoles();
if (!TetR_promoter.getRoles().isEmpty()) {
  System.out.println("TetR_promoter␣set␣is␣not␣empty");
}
TetR_promoter.setRoles(new HashSet<URI>(Arrays.asList(SequenceOntology.PROMOTER)));
```

Creating and Editing References

- Methods to add referenced objects.

```
TetR_promoter.addSequence(seq_187);
TetR_promoter.addSequence(seq_187.getIdentity());
```

- Methods to check if referenced object in the list, remove it, clear or set the entire list.

```
if (TetR_promoter.containsSequence(seq_187.getIdentity())) {
  TetR_promoter.removeSequence(seq_187.getIdentity());
}
TetR_promoter.clearSequences();
TetR_Promoter.setSequence(new HashSet<URI>(Arrays.asList(seq_187.getIdentity())));
```

- Methods to get list of referenced objects (complete documents only) or list of referenced URIs.

```
for (Sequence sequence : TetR_promoter.getSequences());
for (URI sequenceURI : TetR_promoter.getSequenceURIs());
```

Creating and Editing Child Objects

- Child object create methods only require displayId as URIprefix and Version inferred from the parent object.
- All required fields are additional parameters.

```
pIKELeftCassette.createSequenceConstraint(
  "pIKELeftCassette_sc",
  RestrictionType.PRECEDES,
  TetR_promoter.getDisplayId(),
  LacI_repressor.getDisplayId()
);
```

Copying/Renaming TopLevel Objects

- Four variants of createCopy and rename methods.

```
createCopy(topLevel);  // creates identical copy (must be into new document)
createCopy(topLevel, displayId);          // copy with new displayId
createCopy(topLevel, displayId, version); // copy with new displayId/version
createCopy(topLevel, URIprefix, displayId, version);
                                // copy with new URIprefix/displayId/version

ComponentDefinition TetR_promoter_copy =
  (ComponentDefinition)document.createCopy(TetR_promoter , "BBa_K137046");
ComponentDefinition TetR_promoter_rename =
  (ComponentDefinition)document.rename(TetR_promoter , "BBa_K137046");
```

- Copy TopLevel object and all objects that it references into a new SBOLDocument.

```
SBOLDocument newDoc = document.createRecursiveCopy(TetR_promoter);
```

Ontology Support

- Built-in support for several ontologies:
  - Sequence Ontology (SO) - for DNA type ComponentDefinition roles.
  - SystemsBiologyOntology (SBO) - for Model frameworks, Interaction types and Participation roles.
  - EDAM - for Model languages.

```
SequenceOntology.PROMOTER // Constants for common terms
SequenceOntology sequenceOntology = new SequenceOntology();
if (sequenceOntology.isDescendantOf(SequenceOntology.TERMINATOR,
  SequenceOntology.SEQUENCE_FEATURE)) {
  // Terminator is a sequence feature
}
for (URI termURI : sequenceOntology.getDescendantURIsOf(SequenceOntology.TERMINATOR)) {
  System.out.println("Id␣=␣" + getId(termURI) + "␣Name␣=␣" + getName(termURI));
}
```

- Several similar methods with different parameters or return values.

Annotations and GenericTopLevel Objects

- Software tools that need to store data that is not currently encoded within SBOL can do so using generic top level objects and custom annotations.
- When the library reader encounters a tag for a top level object that it does not recognize, this data is stored within a generic top level object.
- Within top level objects, when a tag is not recognized the data is stored within a custom annotation object.
- Tools using our library that do not recognize this data will round-trip it unmodified when writing an SBOL file.
- Tools that would like to make use of this data can interpret and manipulate the raw data, which is stored in a tree-like data structure.

Creating Annotations / GenericTopLevel Objects

- Methods to create Boolean, double, integer, String, URI, and nested annotations.

```
String prPrefix = "pr";
TetR_promoter.createAnnotation(new QName(prURI, "experience", prPrefix),
  URI.create("http://parts.igem.org/Part:BBa_R0040"));
```

- GenericTopLevel objects mostly composed of annotations.

```
String myersLabURI = "http://www.async.ece.utah.edu";
String myersLabPrefix = "myersLab";
GenericTopLevel datasheet=document.createGenericTopLevel("datasheet", "1.0",
  new QName(myersLabURI, "datasheet", myersLabPrefix));
datasheet.setName("Datasheet␣for␣Custom␣Parameters");
datasheet.createAnnotation(
  new QName(myersLabURI, "characterizationData", myersLabPrefix),
  URI.create(myersLabURI + "/measurement/Part:BBa_R0040"));
datasheet.createAnnotation(
  new QName(myersLabURI, "transcriptionRate", myersLabPrefix), 0.75);
TetR_promoter.createAnnotation(
  new QName(myersLabURI, "datasheet", myersLabPrefix),
  datasheet.getIdentity());
```

Serialization Methods

- SBOLReader class:

```
SBOLReader.setURIPrefix("http://www.myparts.org");
SBOLReader.setVersion("1.0");
SBOLReader.setTypesInURI(false);
SBOLReader.setCompliant(true);
SBOLReader.setDropObjectsWithDuplicateURIs(false);
SBOLReader.setKeepGoing(true);

SBOLReader.getSBOLVersion(String OR File OR InputStream);
SBOLDocument document = SBOLReader.read(String OR File OR InputStream);
document.read(String OR File OR InputStream);

SBOLReader.clearErrors();
SBOLReader.getNumErrors();
SBOLReader.getErrors();
```

- SBOLWriter class:

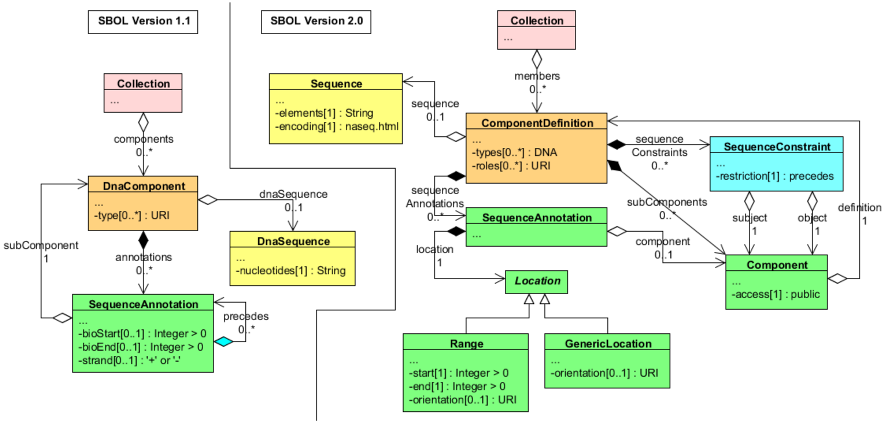
```
SBOLWriter.setKeepGoing(true);

SBOLWriter.write(document, String OR File OR OutputStream);
SBOLWriter.write(document, String OR File OR OutputStream, fileType);
fileType = SBOLDocument.RDF, RDFV1, GENBANK, FASTAformat
document.write(String OR File OR OutputStream, fileType);

SBOLWriter.clearErrors();
SBOLWriter.getNumErrors();
SBOLWriter.getErrors();
```

Conversion

- Supports conversion to/from GenBank and FASTA formats.
- Reads/Writes SBOL 1.1 data files.



Validation

- Many validation rules checked when documents are read.

```
SBOLReader.setKeepGoing(true);
SBOLDocument document = SBOLReader.read(file);
if (SBOLReader.getNumErrors() > 0) {
    for(String error : SBOLReader.getErrors()) {
        System.out.println(error);
    }
}
```

- Remaining validation rules checked by validateSBOL method.

```
boolean complete = true;     // Check that all referenced objects are included
boolean compliant = true;    // Check that all URIs are compliant
boolean bestPractice = true; // Check best practice validation rules
SBOLValidate.validateSBOL(document,complete,compliant,bestPractice);
if (SBOLValidate.getNumErrors() > 0) {
    for(String error : SBOLValidate.getErrors()) {
        System.out.println(error);
    }
}
```

- Method to compare to SBOLDocuments.

```
SBOLValidate.compareDocuments(file,document,file2,document2);
```

More Information

- `libSBOLj` is open source under the Apache 2.0 License.
- More information: `http://sbolstandard.org/software/libSBOL/java/`.
    - Current snapshot on GitHub.
    - Latest release (Version 2.1.0) and GitHub and Maven.
    - Issue tracker for reporting bugs and feature requests.
    - JavaDocs for all public methods.
    - A brief getting started tutorial (basis for this talk).
    - A detailed code example and sample project for CRISPR circuit.
    - Several example code files.

Acknowledgments

Zhen Zhang (Utah)

Tramy Nguyen (Utah)

Meher Samineni (Utah)

Nicholas Roehner (Boston)

Goksel Misirli (Newcastle)

Matthew Pocock (Newcastle)

Ernst Oberortner (DOE JGI)

Jacob Beal (Raytheon)

Kevin Clancy (ThermoFisher)

Anil Wipat (Newcastle)

Supported by National Science Foundation Grants DBI-1356041 and DBI-1355909, and the Engineering and Physical Sciences Research Council under Grant Number EP/J02175X/1.