JSBML: a flexible Java library for working with SBML

We have carefully read all reviewer comments and address each individual concern subsequently in the following point-to-point list. The comments of the reviewers are highlighted in red typewriter font, whereas our answers appear in blue roman font.

Reviewer: 1

This applications note describes a new Java library for SBML. This package is extremely useful to those trying to develop a systems biology tools that are portable to multiple operating systems. We have experimented with this library, and we have found it quite useful. A few minor issues:
1) Include JNI abbreviation after "Java Native Interface", and perhaps add a citation.

We have corrected the text but did not add the citation. The Java Native Interface is a generic feature of the programming languages used, and does not need to be cited in the current paper (as all the generic technologies such as Java, XML etc.)

2) Start of section 2 has changed the font, please correct.

Corrected: The fonts were different because we had used the "methods" environment to describe JSBML in details. To unify the font usage through the document, we removed the Implementation section since its content can be found in the abstract.

3) Page 2, "in form of a tree" -> "in the form of a tree"

Corrected.

Reviewer: 2

The authors describe the implementation of a version of libSBML written but written in Java. One of the authors (Hucka) is the lead developer of libSBML and therefore it is very likely that libSBML and JSBML will maintain parity and will not diverge. In summary, this is a very worthy project that will benefit a great many in the standards community. I was glad to see that JSBML will be able to read mathML as well as infix syntax. I could not see any issues with the manuscript and therefore recommend publication.

Thank you!

Reviewer: 3

To the best of my knowledge, a preliminary version of this library has been presented at the 11th International Conference on Systems Biology (ICSB 2010) by a poster, and at the COmputational Modeling in Biology NEtwork meeting (COMBINE 2010), which brought to the publication of a brief ppt

presentation on Nature Precedings. Some parts of the text in the paper (mainly introduction and conclusions) are copied from the ICSB 2010 poster, but the paper is on the whole original.

The documents quoted by the reviewers were posters and presentations (by the same authors), and are not prior art. The current manuscript is the first article describing JSBML.

The subject of the paper is relevant to the journal and the library presented seems to work properly, however, I have noticed that the jar file jsbml-0.8-b1.jar I have downloaded one week ago has been updated on April 18 by a new file jsbml-0.8-b2.jar. The old version (i.e., 0.8-b1) lacked of some important features, such as:

- method setConsistencyChecks in class SBMLDocument, which enables to add or subtract consistency checks on SBMLDocument objects,

- capability of loading SBML files of Level 3 Version 1 (which is stated in the abstract).

The b1 and b2 in the file name stand for beta 1 and beta 2, as stated. There is no stable release at the present moment, but we schedule the first stable release for June first this year.

I have not found important problems in the new version (i.e., 0.8-b2) of JSBML but I suggest to the authors to carefully test every feature of their code before publishing this paper.

JSBML is already used in several software for production purposes. For instance, Clemens Wrzodek and Andreas Dräger just released the program KEGGtranslator (see http://www.cogsys.cs.uni-tuebingen.de/software/KEGGtranslator/) that converts the content of the KEGG PATHWAY database to SBML. A publication on this program is currently also under review in Bioinformatics (second review state, i.e., revised manuscript). Furthermore, Andreas Dräger and Alexander Dörr are developing a simulator using JSBML and are testing its SBML capabilities using the SBML Test Suite (see http://sbml.org/Facilities/Online_SBML_Test_Suite). The authors are also using an automatic tool to systematically compare the result of what is read and written by JSBML compared to libSBML (http://jsbml.svn.sourceforge.net/viewvc/jsbml/modules/compare/). Finally, JSBML is used by BioModels.net Database, by the Cytoscape team (see http://www.cytoscape.org/), and the Vanted team (see http://vanted.ipk-gatersleben.de/). All these usages provide an intensive and complete test of JSBML, in complement to the unit tests included in the library.

As of today, not many SBML Level 3 models are available, which probably explains why the support for SBML level 3 core had some bugs in the first beta version of JSBML. The SBML Test suite will soon be released with the addition of SBML Level 3 models, which will allow us to test more in depth the support for SBML Level 3 core. Please note that both, the SBML Test suite and the SBML extension packages, are developed separately from JSBML.

Another concern I have about JSBML regards the unavailability of "complex functionalities of libSBML, such as model consistency checking, SBML validation, and conversion between different SBML Levels and Versions". In the introduction of the paper authors say that "separate community efforts are underway to provide such facilities via web services". From my point of view the lack of these facilities could penalize the library, thus I

suggest to develop at least some basic facilities as long as these web services are not available. The solution proposed in section 2.4.6 of the User guide, to use libSBML for parsing SBML by linking JSBML to libSBML, seems not to be very significant since it brings back to the platform dependence problem of libSBML.

As of the beta2 release (0.8-b2), model consistency checking and SBML validation are available directly in JSBML. They require a network connection, using remotely the official SBML Validator using libSBML (http://sbml.org/Facilities/Validator/Validator_Web_API). We do not obligatory consider it as a final solution, as this service presents some limitations. However it is working correctly. The web services that will provide all of the missing complex functionalities are currently actively developed by the SBML team and should be made publicly available before the end of the year. They will be based on a web application running on a Tomcat server that people can install easily locally to avoid network latency if needed. As soon as these services are made available, we will release a new version of JSBML.

From the very beginning of the JSBML project in 2009, it was decided for maintenance and consistency reasons not to duplicate all of the work done with libSBML, but to try to reuse it instead. Another option, though in a longer time frame is the implementation of the SBML validations rules using relax-NG and/or schematron. These rules, encoded in XML, will be used directly in Java without having to pass through a web service.

Finally, I suggest the authors to revise two errors in the User guide: - page 9, line -9: Line 19 -> Line 16

Corrected

- page 9, line -7: (line 10) -> (line 9)

Corrected

and to implement soon the "open tasks" listed in section 3 of the User guide:
- a complete validator for SBML,

This is written in the open tasks but not really planned for all the reasons mentioned earlier. We have updated the User Guide to mention that.

- the support for SBML Level 3,

JSBML already supports SBML Level 3 core. At the moment none of the Level 3 non-core packages are stable. As their specifications will stabilize, JSBML support will be added.

- the toSBML() methods in SBase,
- constructors and methods with namespaces,
- the libSBML compatibility module.

These functionalities have been planned for the next stable release of JSBML, the 1.0 release that will be developed once the 0.8 release cycle is finished.

BIOINFORMATICS **APPLICATIONS NOTE**

JSBML: a flexible Java library for working with SBML

Andreas Dräger [1,*], Nicolas Rodriguez [2,*], Marine Dumousseau [2],
Alexander Dörr [1], Clemens Wrzodek [1], Nicolas Le Novère [2], Andreas Zell [1],
Michael Hucka [3,*]

[1]Center for Bioinformatics Tuebingen (ZBIT), University of Tuebingen, Tübingen, Germany
[2]European Bioinformatics Institute, Wellcome Trust Genome Campus, Hinxton, Cambridge, UK
[3]Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA, USA

Associate Editor: XXXXXXX

ABSTRACT

Summary: The specifications of the Systems Biology Markup Language (SBML) define standards for storing and exchanging computer models of biological processes in text files. In order to perform model simulations, graphical visualizations, and other software manipulations, an in-memory representation of SBML is required. We developed JSBML for this purpose. In contrast to prior implementations of SBML APIs, JSBML has been designed from the ground up for the Java™ programming language, and can therefore be used on all platforms supported by a Java Runtime Environment. This offers important benefits for Java users, including the ability to distribute software as Java Web Start applications. JSBML supports all SBML Levels and Versions through Level 3 Version 1, and we have strived to maintain the highest possible degree of compatibility with the popular library libSBML. JSBML also supports modules that can facilitate the development of plugins for end-user applications, as well as ease migration from a libSBML-based backend.

Availability: Source code, binaries, and documentation for JSBML can be freely obtained under the terms of the LGPL 2.1 from the website http://sbml.org/Software/JSBML.

Contact: jsbml-team@sbml.org

Supplementary information: Supplementary data are available at Bioinformatics online.

1 INTRODUCTION

The XML-based Systems Biology Markup Language (SBML, Hucka *et al.* 2003) is the *de facto* standard file format for the storage and exchange of quantitative computational models in systems biology, supported by more than 210 software packages to date (Mar. 2011). Much of this success is due to its clearly defined specifications and the availability of libSBML (Bornstein *et al.*, 2008), a portable, robust, full-featured, and easy-to-use library.

LibSBML provides many methods for the manipulation and validation of SBML files through its Application Programming Interface (API). Primarily written in C and C++, libSBML also provides automatically-generated language bindings for Java™, among other programming languages. However, the full platform independence brought by the use of Java is limited in libSBML

because the binding is only a wrapper around the C/C++ core, implemented using the Java Native Interface (JNI). As a consequence, some software developers experience difficulties deploying portable libSBML-based Java applications. Several groups in the SBML community thus began to develop their own Java libraries for SBML. To avoid needless duplication, some of these groups recently pooled their efforts and created JSBML, an open-source project to develop a pure Java library for SBML.

The primary aim of the project is to provide an API that maps all SBML elements to a flexible and extended Java type hierarchy. Where possible, JSBML strives for 100 % compatibility with libSBML's Java API, to ease the transition from one library to the other. There are currently no plans to re-implement the more complex functionalities of libSBML, such as model consistency checking, SBML validation, and conversion between different SBML Levels and Versions; separate community efforts are underway to provide such libSBML facilities via web services.

The software produced by the project is freely available from http://sbml.org/Software/JSBML.

2 IMPLEMENTATION

A key achievement of the JSBML project is the development of an extended type hierarchy, designed from scratch based on the SBML specifications, but still following the naming conventions of methods and classes in libSBML. For each element defined in at least one SBML Level/Version combination, JSBML provides a corresponding class reflecting all of its properties. SBML elements or attributes not part of higher SBML Levels (removed or made obsolete) are marked as deprecated. JSBML defines superclasses or interfaces for elements that share common properties. For instance, the interface `NamedSBase` does not directly correspond to a data type in one of the SBML specifications, but serves as the superclass of all `SBase`-derived classes that can be addressed by an identifier and a name. Similarly, all classes that may contain a mathematical expression implement the interface `MathContainer`. A full overview of this type hierarchy can be found in the supplementary data associated with this article. JSBML also supports SBML *notes* in XHTML format, as well as SBML *annotations*, including MIRIAM identifiers (Le Novère *et al.*, 2005) and SBO terms (Le Novère *et al.*, 2006). When building JSBML, the latest SBO
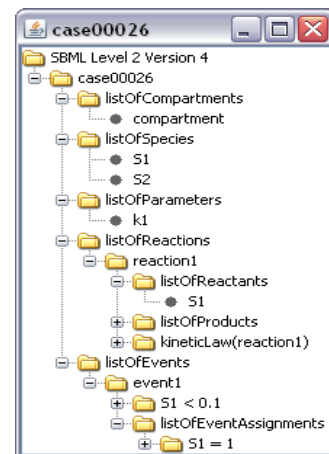
---

*to whom correspondence should be addressed

Dräger and Rodriguez et al.

```java
import javax.swing.*;
import org.sbml.jsbml.*;

/** Displays the content of an SBML file in a {@link JTree} */
public class JSBMLvisualizer extends JFrame {

  /** @param document The sbml root node of an SBML file */
  public JSBMLvisualizer(SBMLDocument document) {
    super(document.getModel().getId());
    getContentPane().add(new JScrollPane(new JTree(document)));
    pack();
    setVisible(true);
  }
  /** @param args Expects a valid path to an SBML file. */
  public static void main(String[] args) throws Exception {
    new JSBMLvisualizer((new SBMLReader()).readSBML(args[0]));
  }
}
```

(a) The SBML parser in JSBML understands the hierarchical data structure of SBML.

(b) Example for SBML test case 26.

Fig. 1: Using JSBML for reading and visualizing an SBML file. The type `SBase` extends the Java interfaces `Serializable` for saving JSBML objects in binary form or sending them over a network connection, `Cloneable` for creating deep object copies, and `TreeNode`. The last interface allows callers to apply any recursive operation, such as using `JTree` for display (see 1b for an example).

OBO export can directly be downloaded and parsed (Holland *et al.*, 2008). The `Model` class provides several methods, all beginning with the name `find*`, for querying SBML elements. Filters enable users to search lists for elements that possess specific properties. All `ListOf*` elements in JSBML implement Java's `List` interface, making iteration and the use of generic Java types possible. Fig. 1 demonstrates how the hierarchically structured content of an SBML file can be easily visualized in the form of a tree.

JSBML includes parsers that read mathematical formulas in both MathML format and an infix formula syntax. Internally, it converts formulas into an abstract syntax tree representation; it can write out the trees in MathML, infix, and LATEX formula notations. In addition, although JSBML does not implement full-featured consistency checking of SBML models, it does throw Java exceptions in some situations to prevent users from creating invalid content. It implements a check for overdetermined models using the algorithm of Hopcroft and Karp (1973); this is also used to identify variables in algebraic rules. Further, JSBML can automatically derive the units of a mathematical expression. Whenever a property of some `SBase` is altered, an `SBaseChangeEvent` is fired that notifies dedicated listeners. As one possible application, graphical user interfaces could automatically react when the model is changed. Using modules, JSBML capabilities can be further extended; it can therefore be used as a communication layer between an application and libSBML or CellDesigner (Funahashi *et al.*, 2003)—this also facilitates turning an existing application into a plugin for CellDesigner.

3 CONCLUSION

JSBML is an ongoing project that provides comprehensive and entirely Java-based data structures to read, write, and manipulate SBML files. Its layered architecture allows for the creation of Java Web Start applications and CellDesigner plugins based on stand-alone programs with very little effort. New versions of SBMLsqueezer (Dräger *et al.*, 2008) and Biomodels Database (Li *et al.*, 2010) have already been released using JSBML. Many other projects are planned.
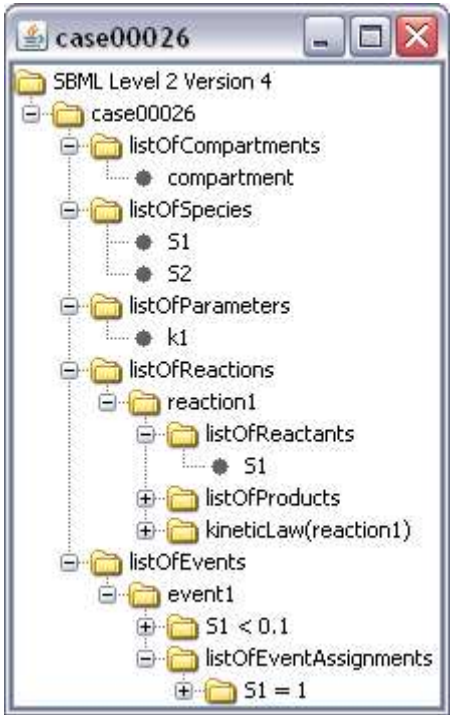
REFERENCES

Bornstein *et al.* (2008). LibSBML: an API Library for SBML. *Bioinformatics*, **24**(6), 880–881.

Dräger *et al.* (2008). SBMLsqueezer: a CellDesigner plug-in to generate kinetic rate equations for biochemical networks. *BMC Syst. Biol.*, **2**(1), 39.

Funahashi *et al.* (2003). CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. *BioSilico*, **1**(5), 159–162.

Holland *et al.* (2008). BioJava: an Open-Source Framework for Bioinformatics. *Bioinformatics*, **24**(18), 2096–2097.

Hopcroft and Karp (1973). An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.*, **2**, 225.

Hucka *et al.* (2003). The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, **19**(4), 524–531.

Le Novère *et al.* (2005). Minimum information requested in the annotation of biochemical models (MIRIAM). *Nat. Biotechnol.*, **23**(12), 1509–1515.

Le Novère *et al.* (2006). Adding semantics in kinetics models of biochemical pathways. In Kettner and Hicks, eds., *2nd International ESCEC Workshop. Beilstein Institut, Rüdesheim, Germany*, pages 137–153, Rüdessheim/Rhein, Germany. ESEC.

Li *et al.* (2010). BioModels Database: An enhanced, curated and annotated resource for published quantitative kinetic models. *BMC Syst Biol*, 2010, 4, 92

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Example for SBML test case 26.