SBML Level 3 Package: Flux Balance Constraints ('fbc')

Brett G. Olivier
b.g.olivier@vu.nl
Systems Bioinformatics
VU University Amsterdam
Amstedam, NH, The Netherlands

Frank T. Bergmann
fbergmann@caltech.edu
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA, US

November 15, 2012 (0.6)

Version 1 (Draft)

This is a draft specification for the package 'fbc' and not a normative document. Please send feedback to the Package Working Group mailing list at sbml-flux@lists.sourceforge.net

The latest release, past releases, and other materials related to this specification are available at
http://sbml.org/Documents/Specifications/SBML_Level_3/Packages/Flux_Balance_Constraints_(flux)

This release of the specification is available at
http://sbml.org/Documents/Specifications/Fbc_Level_1_Version_1

Contents

1 Introduction and motivation

Constraint based modelling is a widely used methodology used to analyze and study biological networks on both a small and whole organism (genome) scale. Typically these models are underdetermined and constraint based methods (e.g. linear, quadratic optimization) are used to optimize specific model properties. This is assumed to occur under a defined set of constraints (e.g. stoichiometric, metabolic) and bounds (e.g. thermodynamic, experimental and environmental) on the values that the solution fluxes can obtain.

Perhaps the most well known (and widely used) analysis method is Flux Balance Analysis (FBA; Orth et al., 2010) which is performed on Genome Scale Reconstructions (GSR's; Oberhardt et al., 2009). Using FBA a target flux is optimized (e.g. maximizing a flux to biomass or minimizing ATP production) while other fluxes can be bounded to simulate a selected growth environment or specific metabolic state.

As constraint based models are generally underdetermined, i.e. few or none of the kinetic rate equations and related parameters are known, it is crucial that a model definition includes the ability to define optimization parameters such as objective functions, flux bounds and constraints ... currently this is not possible in the Systems Biology Markup Language (SBML) Level 2 or Level 3 core specification (Hucka et al., 2011, 2003).

The question of how to encode constraint based (a.k.a. 'FBA') models in SBML is not new. However, advances in the methods used to construct GSR scale models and the wider adoption of constraint based modelling in biotechnological/medical applications have led to a rapid increase in both the number of models being constructed and the tools used to analyze them.

Faced with such growth, both in number and diversity, the need for a standardized data format for the definition, exchange and annotation of constraint based models has become critical. As the core model components (e.g. species, reactions, stoichiometry) can already be efficiently described in SBML (with its significant community, software and tool support) the Flux Balance Constraints Package aims to extend SBML core by adding the elements necessary to describe current and future constraint based models.

1.1 Proposal corresponding to this package specification

This specification for Flux Balance Constraints in SBML Level 3 Version 1 is based on the proposal by the same authors, located at the following URL:

http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Flux_Balance_Constraints_Proposal_(2012)

The tracking number in the SBML issue tracking system (SBML Team, 2010) for Flux Balance Constraints package activities is 3154219. The version of the proposal used as the starting point for this specification is the version of March 2012. Previous versions of the current proposal are:

Version 3 (March 2012)

http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Flux_Balance_Constraints_Proposal_(2012)

Version 2 (March 2011)

http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Flux_Constraints_Proposal

Version 1 (February 2010)

http://precedings.nature.com/documents/4236/version/1

Details of earlier independent proposals are provided in Section 2.

1.2 Tracking number

As initially listed in the SBML issue tracking system under: http://sourceforge.net/tracker/?func=detail&aid=3154219&group_id=71971&atid=894711.

1.3 Package dependecies

The Flux Balance Constraints package has no dependencies on other SBML Level 3 packages. It is also designed with the goal of being able to work seamlessly with other SBML Level 3 packages. For example any objects are entirely encapsulated and any extensions to existing SBML classes are defined as optional.

1.4 Document conventions

Following the precedent set by the SBML Level 3 Core specification document, we use UML 1.0 (Unified Modeling Language; Eriksson and Penker 1998; Oestereich 1999) class diagram notation to define the constructs provided by this package. We also use color in the diagrams to carry additional information for the benefit of those viewing the document on media that can display color. The following are the colors we use and what they represent:

- *Black*: Items colored black in the UML diagrams are components taken unchanged from their definition in the SBML Level 3 Core specification document.
- *Green*: Items colored green are components that exist in SBML Level 3 Core, but are extended by this package. Class boxes are also drawn with dashed lines to further distinguish them.
- *Blue*: Items colored blue are new components introduced in this package specification. They have no equivalent in the SBML Level 3 Core specification.

We also use the following typographical conventions to distinguish the names of objects and data types from other entities; these conventions are identical to the conventions used in the SBML Level 3 Core specification document:

AbstractClass: Abstract classes are classes that are never instantiated directly, but rather serve as parents of other object classes. Their names begin with a capital letter and they are printed in a slanted, bold, sans-serif typeface. In electronic document formats, the class names defined within this document are also hyperlinked to their definitions; clicking on these items will, given appropriate software, switch the view to the section in this document containing the definition of that class. (However, for classes that are unchanged from their definitions in SBML Level 3 Core, the class names are not hyperlinked because they are not defined within this document.)

Class: Names of ordinary (concrete) classes begin with a capital letter and are printed in an upright, bold, sans-serif typeface. In electronic document formats, the class names are also hyperlinked to their definitions in this specification document. (However, as in the previous case, class names are not hyperlinked if they are for classes that are unchanged from their definitions in the SBML Level 3 Core specification.)

`SomeThing`, `otherThing`: Attributes of classes, data type names, literal XML, and generally all tokens *other* than SBML UML class names, are printed in an upright typewriter typeface. Primitive types defined by SBML begin with a capital letter; SBML also makes use of primitive types defined by XML Schema 1.0 (Biron and Malhotra, 2000; Fallside, 2000; Thompson et al., 2000), but unfortunately, XML Schema does not follow any capitalization convention and primitive types drawn from the XML Schema language may or may not start with a capital letter.

For other matters involving the use of UML and XML, we follow the conventions used in the SBML Level 3 Core specification document.

2 Background

2.1 Problems with current SBML approaches

While there is currently no official way of encoding constraint based models in SBML L2 there have been pragmatic approaches used by a variety of groups and applications. Arguably the best and most widely used format is that used by the COBRA toolbox (Becker et al., 2007) where the metabolic network is well defined using SBML **Reaction** and **Species** classes. However, flux bounds and reactions that take part in the objective function are defined as **LocalParameter** objects and (implicitly) rely on all tools using the same naming convention. Similarly, reaction annotations are generally stored as key-value pairs in HTML **Notes** objects which has routinely led to different groups and software using in-house key definitions describing the same entity. While a step in the right direction this format is not suitable for implementation in SBML Level 3.

2.2 Past work on this problem or similar topics

The problem of describing and annotating 'FBA' models in SBML has been raised at various times in the past few years. In this regard there are two known putative proposals one by Karthik Raman and the other by the Church Laboratory. As far as we are aware these proposals never developed beyond their initial presentation at SBML forum/hackathons. In 2009 the discussion was reopened at the SBML Forum held in Stanford and has subsequently developed into the current active package proposal and this document (see Section 1).

Brett Olivier (2009) SBML Level 3 FBA package discussion

`http://sbml.org/images/4/4a/Olivier_sbml_forum_2009_09_04.pdf`

Karthik Raman (2005) Flux annotations in SBML

`http://sbml.org/images/d/d9/Raman-flux-annotations.pdf`

Church laboratory (pre 2005) Metabolic flux model annotations

`http://sbml.org/Community/Wiki/Old_known_SBML_annotations_list`

3 Proposed syntax and semantics

In this section, we define the syntax and semantics of the Flux Balance Constraints package for SBML Level 3 Version 1. We expound on the various data types and constructs defined in this package, then in Section 4 on page 12, we provide complete examples of using the constructs in example SBML models.

3.1 Namespace URI and other declarations necessary for using this package

Every SBML Level 3 package is identified uniquely by an XML namespace URI. For an SBML document to be able to use a given SBML Level 3 package, it must declare the use of that package by referencing its URI. The following is the namespace URI for this version of the Flux Balance Constraints package for SBML Level 3 Version 1:

"`http://www.sbml.org/sbml/level3/version1/fbc/version1`"

In addition, SBML documents using a given package must indicate whether understanding the package is required for complete mathematical interpretation of a model, or whether the package is optional. This is done using the attribute `required` on the `<sbml>` element in the SBML document. For the Flux Balance Constraints package, the value of this attribute must be set to "`true`".

The following fragment illustrates the beginning of a typical SBML model using SBML Level 3 Version 1 and this version of the Flux Balance Constraints package:

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core" level="3"
version="1"
      xmlns:fbc="http://www.sbml.org/sbml/level3/version1/fbc/version1"
      fbc:required="true">
```

3.2 Primitive data types

Section 3.1 of the SBML Level 3 Version 1 Core specification defines a number of primitive data types and also uses a number of XML Schema 1.0 data types (Biron and Malhotra, 2000). More specifically we make use of `integer`, `double`, `string`, `SIdRef` and `enum`. In addition we make use of two new primitives `FbcSId` and `FbcSIdRef`, see Figure 1 for the interrelation between these entities.

3.2.1 *Type* `FbcSId`

The type `FbcSId` is derived from `SId` (SBML Level 3 Version 1 Core specification Section 3.1.7) and has identical syntax. The `FbcSId` type is used as the data type for the identifiers of **FluxBound** (Section 3.5) and **Objective** (Section 3.6) classes. By using a separate identifier type we differentiate them from others defined in the **SBML** model and thus ensuring data encapsulation. In addition the **Objective** class `FbcSId` provides an identifier to the **Objective** which is set as active. The equality of `FbcSId` values is determined by an exact character sequence match and therefore comparisons of these identifiers must be performed in a case-sensitive manner.

3.2.2 *Type* `FbcSIdRef`

Type `FbcSIdRef` is used for all attributes that refer to identifiers of type `FbcSId`. Derived from `FbcSId` it has the restriction that the value of an attribute having type `FbcSIdRef` must match the value of a `FbcSId` attribute in the current model. In the FBC package the **ListOfObjectives** has an attribute of this type that is used to refer to an 'active' **Objective** .

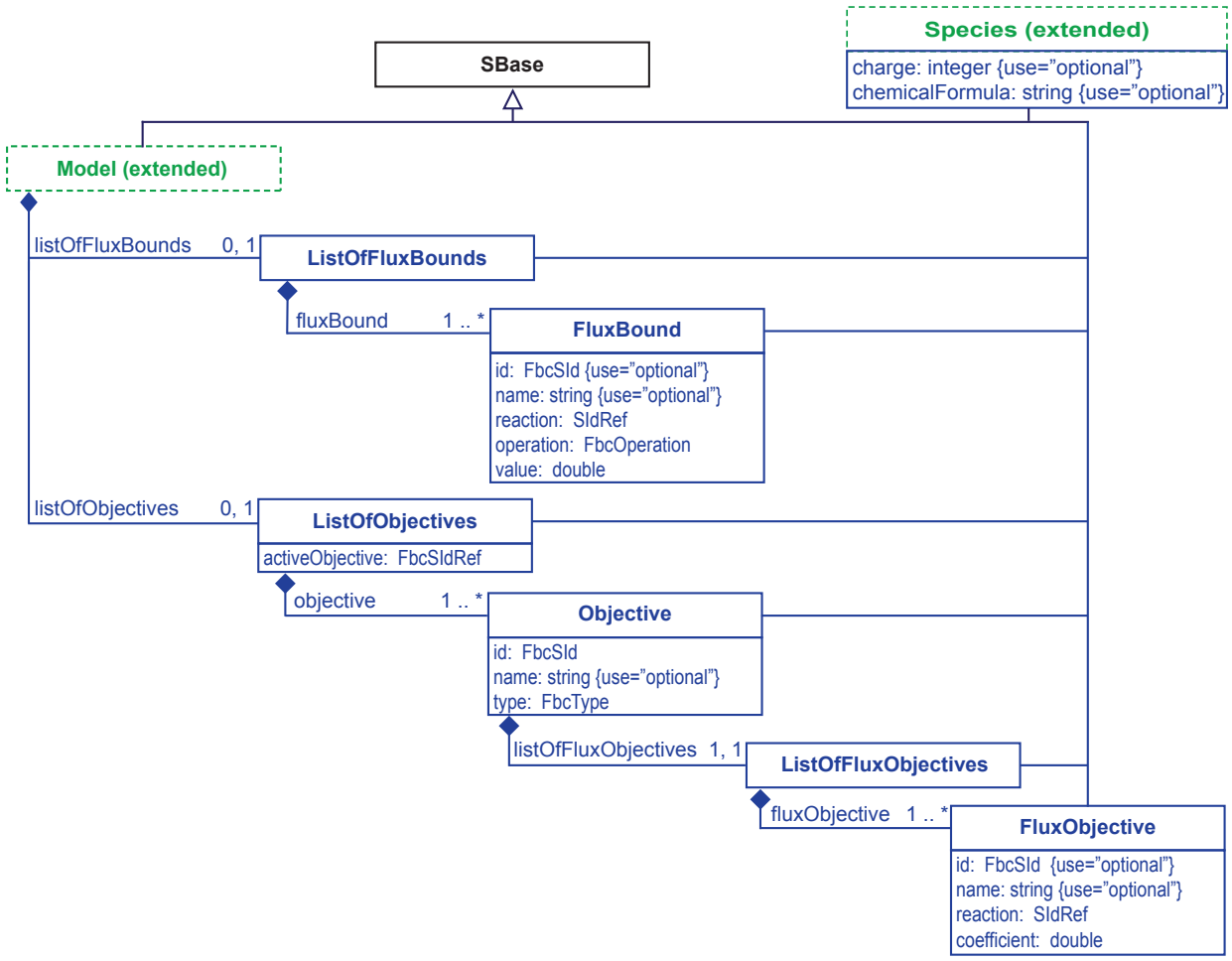

Figure 1: *A UML representation of the Flux Balance Constraints package classes. See Section 1.4 for conventions related to this figure. The individual classes are further discussed in the text.*

3.2.3 *Type* `FbcType`

The Flux Balance Constraints package defines a new enumerated type `FbcType` which represents the optimization sense of the objective function. It can have one of the following two values "`maximize`" or "`minimize`".

3.2.4 *Type* `FbcOperation`

The Flux Balance Constraints package defines a new enumerated type `FbcOperation` which represents a boolean operator. It can take only one of the following values: "`lessEqual`", "`greaterEqual`", "`less`", "`greater`" or "`equal`".

3.3 The extended Model class

The **SBML Model** class is extended with the addition of two children, i.e. a `listOfFluxBounds` and a `listOfObjectives`. The **Model** may contain at most one of these lists.

3.3.1 *The FBC listOfFluxBounds*

As shown in Figure 1 the **ListOfFluxBounds** is derived from ***SBase*** and inherits the attributes `metaid` and `sboTerm`, as well as the subcomponents for **Annotation** and **Notes**. **ListOfFluxBounds** must contain at least one **FluxBound** (defined in Section 3.5).

3.3.2 The FBC listOfObjectives

As shown in Figure 1 the **ListOfObjectives** is derived from ***SBase*** and inherits the attributes `metaid` and `sboTerm`, as well as the subcomponents for **Annotation** and **Notes**. Unlike most other **SBML ListOf__** classes, **ListOfObjectives** introduces an additional required attribute `activeObjective`. The **ListOfObjectives** must contain at least one **Objective** (defined in Section 3.6).

The `activeObjective` *attribute*

This attribute contains a "`value`" of type `FbcSIdRef` that can only refer to an existing **Objective** (FbcSId). This required attribute exists so that when multiple **Objective** 's are included in a single model, the model will always be well described i.e. there is a single, primary objective function which closes the solution space.

3.4 The extended Species class

The Flux Balance Constraints package extends the SBML Level 3 Version 1 Core **Species** class with the addition of two attributes:

The `charge` *attribute*

The optional attribute `charge` which contains a signed `integer` referring to the **Species** object's charge (as defined in the **SBML** Level 2 Version 1 specification)

The `chemicalFormula` *attribute*

The optional attribute `chemicalFormula` containing a `string` that represents the **Species** objects elemental composition.

```
<species metaid="meta_M_atp_c␣" id="M_atp_c" name="ATP"
    compartment="Cytosol" boundaryCondition="false"
    fbc:charge="-4" fbc:chemicalFormula="C10H12N5O13P3"
    initialConcentration="0" hasOnlySubstanceUnits="false"
/>
```

While there are many ways of referring to an elemental composition the purpose of the `chemicalFormula` attribute is to allow reaction balancing and validation which is particularly important in constraint based models. To this end it is recommended that the format of `chemicalFormula` should follow the Hill system (or notation). Here the number of carbon atoms in a molecule is indicated first, followed by the number of hydrogen atoms and then the number of all other chemical elements in alphabetical order. When the formula contains no carbon; all elements, including hydrogen, are listed alphabetically Hill (1900, 2012).

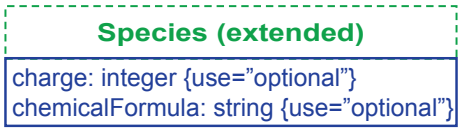

Figure 2: *A UML representation of the extended* ***SBML Species*** *class used in the Flux Balance Constraints package. See Section 1.4 for conventions related to this figure.*

3.5 The FBC FluxBound class

FluxBound is a new FBC class derived from **SBML** ***SBase*** that inherits `metaid` and `sboTerm`, as well as the subcomponents for **Annotation** and **Notes**. The purpose of this class is to hold a single (in)equality that provides the maximum or minimum value that a reaction flux can obtain at steady state. It is is relatively straight forward and

implements four attributes.

The `id` *and token* `name` *attributes*

A **FluxBound** has two optional attributes: `id` an attribute that holds an `FbcSId` and `name` that contains a `string`.

The `reaction` *attribute*

The required `reaction` attribute that takes only an **SBML Reaction** `SId` as a value.

The `operation` *attribute*

The `operation` attribute contains a value of type `FbcOperation` that can take a limited set of boolean operators as defined in Section 3.2.4. The `operation` attribute represents a mathematical (in)equality of the form <`reaction`> <`operator`> <`value`> e.g. $R_5 >= 0$, $R_5 < \infty$ and $R_7 = 1.0$. The mapping between traditional mathematical symbols and `FbcOperation` values is as follows:

$$\begin{aligned} <= &\mapsto \text{“}\texttt{lessEqual}\text{”} \\ >= &\mapsto \text{“}\texttt{greaterEqual}\text{”} \\ < &\mapsto \text{“}\texttt{less}\text{”} \\ > &\mapsto \text{“}\texttt{greater}\text{”} \\ = &\mapsto \text{“}\texttt{equal}\text{”} \end{aligned}$$

The `value` *attribute*

The `value` attribute holds a `double` value representing the numerical value of the flux bound. This may include an explicitly defined $\pm\infty$ encoded as a value, e.g. “`INF`”.

Encoding the FluxBound

As described in Section 3.5 the flux bound represents a mathematical (in)equality of the form <`reaction`> <`operator`> <`value`>. In SBML Level 3 Version 1 with FBC this is encoded as:

```
<fbc:listOfFluxBounds>
 <fbc:fluxBound fbc:id="R1b" fbc:reaction="R1" fbc:operation="greaterEqual" fbc:value="1.2"/>
 <fbc:fluxBound fbc:id="R2b" fbc:reaction="R2" fbc:operation="lessEqual" fbc:value="-1.2"/>
 <fbc:fluxBound fbc:id="R3b" fbc:reaction="R3" fbc:operation="greaterEqual" fbc:value="-INF"/>
 <fbc:fluxBound fbc:id="R4b" fbc:reaction="R4" fbc:operation="lessEqual" fbc:value="INF"/>
 <fbc:fluxBound fbc:id="R5b" fbc:reaction="R5" fbc:operation="equal" fbc:value="1"/>
 <fbc:fluxBound fbc:id="R6b" fbc:reaction="R6" fbc:operation="greater" fbc:value="-1"/>
 <fbc:fluxBound fbc:id="R7b" fbc:reaction="R7" fbc:operation="less" fbc:value="1"/>
</fbc:listOfFluxBounds>
```

This example illustrates two things: the encoding of ∞ and that care should be used when selecting inequalities such as “`less`” or “`greater`”. While mathematically there is a difference, this difference is only practically relevant when working with rational arithmetic (solvers).

3.6 The FBC Objective class

The FBC **Objective** class is derived from **SBML** ***SBase*** and inherits `metaid` and `sboTerm`, as well as the subcomponents for **Annotation** and **Notes**. An integral component in a complete description of a steady-state model is the so-called ‘objective function’ which generally consist of a linear combination of model variables (fluxes) and a sense (direction). In the FBC package this concept is succinctly captured in the **Objective** class.

The `id` *and token* `name` *attributes*

An **Objective** has a required attribute `id` that holds an `FbcSId` and and optional attribute `name` that contains a `string`.

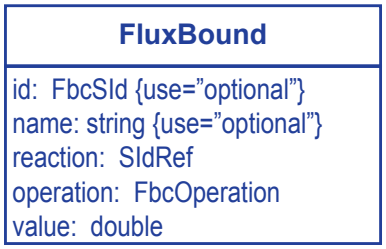

Figure 3: *A UML representation of the Flux Balance Constraints package* ***FluxBound*** *class. See Section 1.4 for conventions related to this figure.*

The `type` *attribute*

The required `type` attribute contains an `FbcType` type which represents the sense of the optimality constraint and can take one of two values:

$$maximize \mapsto \text{“}\texttt{maximize}''$$
$$minimize \mapsto \text{“}\texttt{minimize}''$$

The `listOfFluxObjectives` *element*

The attribute `listOfFluxObjectives` which contains a **ListOfFluxes** is derived from and functions like a typical **SBML ListOf__** class with the restriction that it cannot be empty and therefore must contain one or more elements of type **FluxObjective** (see Section 3.7).

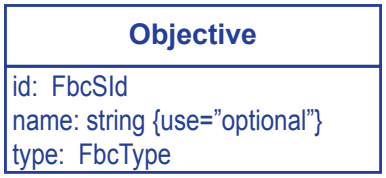

Figure 4: *A UML representation of the Flux Balance Constraints package* ***Objective*** *class. See Section 1.4 for conventions related to this figure.*

Encoding the Objective

The Flux Balance Constraints package allows for the definition of multiple model objectives with one being designated as active (see Section 3.6) as illustrated in this example:

```
<fbc:listOfObjectives fbc:activeObjective="obj1">
 <fbc:objective fbc:id="obj1" fbc:type="maximize">
  <fbc:listOfFluxes>
   <fbc:fluxObjective fbc:reaction="R101" fbc:coefficient="1"/>
  </fbc:listOfFluxes>
 </fbc:objective>
 <fbc:objective fbc:id="obj2" fbc:type="minimize">
  <fbc:listOfFluxes>
   <fbc:fluxObjective fbc:reaction="R102" fbc:coefficient="-2.5"/>
   <fbc:fluxObjective fbc:reaction="R103" fbc:coefficient="1"/>
  </fbc:listOfFluxes>
 </fbc:objective>
</fbc:listOfObjectives>
```

Note how both **Objective** instances differ in `type` and each contains different set of **FluxObjectives** (see Section 3.7).

3.7 The FBC FluxObjective class

The FBC **FluxObjective** class is derived from **SBML *SBase*** and inherits `metaid` and `sboTerm`, as well as the sub-components for **Annotation** and **Notes**.

The **FluxObjective** class is a relatively simple container for a model variable weighted by a signed linear coefficient.

The `id` *and* `name` *attributes*

A **FluxObjective** has two optional attributes: `id` an attribute that holds an `FbcSId` and `name` that contains a `string`.

The `reaction` *and* `coefficient` *attributes*

The required `reaction` attribute contains an `SId` that is restricted to refer only to an **SBML Reaction** while the `coefficient` attribute holds a `double` referring to this variables objective coefficient.

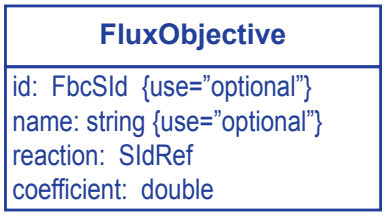

Figure 5: *A UML representation of the Flux Balance Constraints package* ***FluxObjective*** *class. See Section 1.4 for conventions related to this figure.*

4 Examples

This section contains a variety of examples of SBML Level 3 Version 1 documents employing the Flux Balance Constraints package.

4.1 FBC syntax examples

These examples are provided to highlight the Flux Balance Constraints package syntax.

4.1.1 Example One

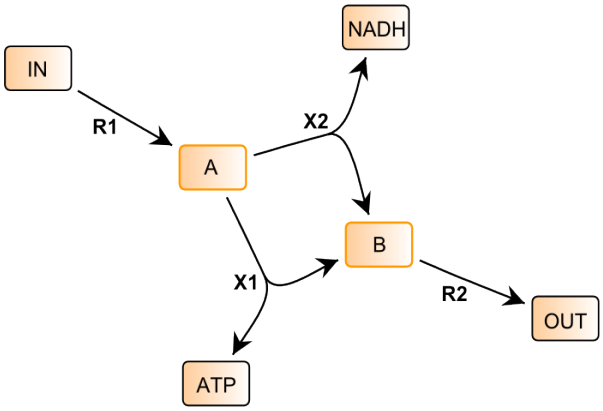

Figure 6: *FBC syntax example: a simple four reaction pathway. The reactions are R1, R2, X1, X2 with fixed species IN, OUT, ATP, NADH and variable species A, B.*

As shown in Figure 6 the first example is a simple four reaction pathway that transforms metabolite *IN* to *OUT*. To begin with it is possible to compactly describe this network in terms of its reaction stoichiometry as shown in Table 1. While the stoichiometry contains the structural properties of the reaction network the full description of a

	R1	R2	X1	X2
A	1	0	-1	-1
B	0	-1	1	1

Table 1: *Example one: stoichiometric matrix,* **N**

biological model can be described as a set of ordinary differential equations (ODE's). While other formalisms may exist here we will concentrate exclusively on kinetic models where the change in concentration of each variable component in the system ($\frac{ds}{dt}$) is a non-linear function of the rates of the reactions which either create or consume it (the product of the stoichiometric matrix, **N** and the vector of reaction rates, **v**).

$$\frac{ds}{dt} = \mathbf{Nv} \tag{1}$$

The formulation of the kinetic model, as shown in Equation 1 is typical of the kind that can already be described using 'traditional' **SBML** where the vector **v** would contain rate equations as a function of parameters and concentrations. In the XML representation of this model provided below the reaction stoichiometry is captured in the **Species** and **Reaction** definitions. In a constraint based model these rate laws are considered unknowns and the

system is assumed to be in steady state (see Equation 2):

$$\mathbf{NJ} = 0 \tag{2}$$

Note how the rate vector **v** is now represented as the flux vector **J**. To perform a typical steady-state analysis such as flux balance analysis (FBA) we need to include to more pieces of information into our model description:

1. A capacity constraint: the maximum limit (upper bound) of the flux through reaction *R1* is 1 (with an arbitrary unit of flux).
2. An objective target which can be maximized or minimized: in this example the flux through reaction *R2* will be *maximized*.

However, SBML Level 3 Version 1 Core does not have an unambiguous way of encoding either a capacity constraint or an objective target and for this we now need to use the additional constructs provided by the Flux Balance Constraints package. In the example XML shown below, this additional information is encoded from Line 83–92. Encoded in this way the constraint based model can now be formulated as a Linear Program and analyzed using FBA (here shown in LP format as used by IBM CPLEX):

```
\\ Example one LP format

Maximize
objective1_objf:  + 1.0 R2

Subject To
 A: + R1 - X1 - X2 = 0.0
 B: - R2 + X1 + X2 = 0.0

Bounds
R1 <= 1.0
0.0 <= R2 <= +inf
0.0 <= X1 <= +inf
0.0 <= X2 <= +inf

END
```

Solving this problem we find that maximization of flux through *R2* gives an optimal solution of *one* shown with one possible solution for **J** (see Equation 3).

$$\begin{pmatrix} 1 & 0 & -1 & -1 \\ 0 & -1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1.0 \\ 1.0 \\ 0.0 \\ 1.0 \end{pmatrix} = \mathbf{0} \tag{3}$$

The following code block is the XML that represents the model described as Example One in the preceding text.

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1/core"
      xmlns:fbc="http://www.sbml.org/sbml/level3/version1/fbc/version1"
      level="3" version="1" fbc:required="false">
  <model id="fbcSpecExample1" timeUnits="time">
    <listOfUnitDefinitions>
      <unitDefinition id="volume">
        <listOfUnits>
          <unit kind="litre" exponent="1" scale="0" multiplier="1"/>
        </listOfUnits>
      </unitDefinition>
      <unitDefinition id="substance">
        <listOfUnits>
```

```
          <unit kind="mole" exponent="1" scale="0" multiplier="1"/>
        </listOfUnits>
      </unitDefinition>
      <unitDefinition id="time">
        <listOfUnits>
          <unit kind="second" exponent="1" scale="0" multiplier="1"/>
        </listOfUnits>
      </unitDefinition>
    </listOfUnitDefinitions>
    <listOfCompartments>
      <compartment id="compartment" spatialDimensions="3" size="1"
       units="volume" constant="true"/>
    </listOfCompartments>
    <listOfSpecies>
      <species id="IN" compartment="compartment" initialConcentration="0"
       substanceUnits="substance" hasOnlySubstanceUnits="false"
       boundaryCondition="true" constant="false"/>
      <species id="OUT" compartment="compartment" initialConcentration="0"
       substanceUnits="substance" hasOnlySubstanceUnits="false"
       boundaryCondition="true" constant="false"/>
      <species id="A" compartment="compartment" initialConcentration="0"
       substanceUnits="substance" hasOnlySubstanceUnits="false"
       boundaryCondition="false" constant="false"/>
      <species id="B" compartment="compartment" initialConcentration="0"
       substanceUnits="substance" hasOnlySubstanceUnits="false"
       boundaryCondition="false" constant="false"/>
      <species id="ATP" compartment="compartment" initialConcentration="0"
       substanceUnits="substance" hasOnlySubstanceUnits="false"
       boundaryCondition="true" constant="false"/>
      <species id="NADH" compartment="compartment" initialConcentration="0"
       substanceUnits="substance" hasOnlySubstanceUnits="false"
       boundaryCondition="true" constant="false"/>
    </listOfSpecies>
    <listOfReactions>
      <reaction id="R1" reversible="false" fast="false">
        <listOfReactants>
          <speciesReference species="IN" stoichiometry="1" constant="true"/>
        </listOfReactants>
        <listOfProducts>
          <speciesReference species="A" stoichiometry="1" constant="true"/>
        </listOfProducts>
      </reaction>
      <reaction id="R2" reversible="false" fast="false">
        <listOfReactants>
          <speciesReference species="B" stoichiometry="1" constant="true"/>
        </listOfReactants>
        <listOfProducts>
          <speciesReference species="OUT" stoichiometry="1" constant="true"/>
        </listOfProducts>
      </reaction>
      <reaction id="X1" reversible="false" fast="false">
        <listOfReactants>
          <speciesReference species="A" stoichiometry="1" constant="true"/>
        </listOfReactants>
        <listOfProducts>
          <speciesReference species="ATP" stoichiometry="1" constant="true"/>
          <speciesReference species="B" stoichiometry="1" constant="true"/>
        </listOfProducts>
      </reaction>
      <reaction id="X2" reversible="false" fast="false">
        <listOfReactants>
          <speciesReference species="A" stoichiometry="1" constant="true"/>
        </listOfReactants>
        <listOfProducts>
          <speciesReference species="B" stoichiometry="1" constant="true"/>
          <speciesReference species="NADH" stoichiometry="1" constant="true"/>
```

```
        </listOfProducts>
      </reaction>
    </listOfReactions>
    <fbc:listOfFluxBounds>
      <fbc:fluxBound fbc:reaction="R1" fbc:operation="lessEqual" fbc:value="1"/>
    </fbc:listOfFluxBounds>
    <fbc:listOfObjectives fbc:activeObjective="objective1">
      <fbc:objective fbc:id="objective1" fbc:type="maximize">
        <fbc:listOfFluxes>
          <fbc:fluxObjective fbc:reaction="R2" fbc:coefficient="1"/>
        </fbc:listOfFluxes>
      </fbc:objective>
    </fbc:listOfObjectives>
  </model>
</sbml>
```

5 Best practices

In this section, we recommend a number of practices for using and interpreting various constructs in the Flux Balance Constraints package. These recommendations are non-normative, but we advocate them strongly; ignoring them will not render a model invalid, but may reduce interoperability between software and models.

5.1 Examples contrasting the current SBML L2 encoding with L3 and FBC

These examples contrast some elements of an existing model, iJR904 from the BiGG Database encoded in the COBRA format Becker et al. (2007); Reed et al. (2003); Schellenberger et al. (2010) that have been translated into SBML Level 3 Version 1 using the CBMPy implementation of the FBC package Olivier (2012); Olivier et al. (2005) and libSBML experimental ver. 5.6.0 Bornstein et al. (2008).

Objective function definition

Old style SBML Level 2 objective

```
<reaction id="R_BiomassEcoli" name="BiomassEcoli" reversible="false">
<kineticLaw>
<math xmlns="http://www.w3.org/1998/Math/MathML">
<ci>FLUX_VALUE</ci>
</math>
<listOfParameters>
<parameter id="LOWER_BOUND" value="0" units="mmol_per_gDW_per_hr"/>
<parameter id="UPPER_BOUND" value="999999" units="mmol_per_gDW_per_hr"/>
<parameter id="OBJECTIVE_COEFFICIENT" value="1" />
<parameter id="FLUX_VALUE" value="0" units="mmol_per_gDW_per_hr"/>
</listOfParameters>
</kineticLaw>
</reaction>
```

New SBML Level 3 style objective

```
<fbc:listOfObjectives fbc:activeObjective="obj1">
 <fbc:objective fbc:id="obj1" fbc:type="maximize">
  <fbc:listOfFluxes>
   <fbc:fluxObjective fbc:reaction="R_BiomassEcoli" fbc:coefficient="1"/>
  </fbc:listOfFluxes>
 </fbc:objective>
</fbc:listOfObjectives>
```

Species definition

It is particularly useful to contrast the differences in **Species** definition as used in constraint based, genome scale models.

Classic SBML Level 2 Species annotation

To begin with we let's examine the **SBML** Level 2 Version 1 species definition used by the BiGG database and COBRA Becker et al. (2007); Schellenberger et al. (2010). Note how the `name` attribute is overloaded with the chemical formula.

```
<species id="M_atp_c" name="ATP_C10H12N5O13P3"
    compartment="Cytosol" charge="-4" />
```

An alternate SBML Level 2 Species annotation

A newer variation of the above, probably necessitated by the discontinuation of the `charge` attribute in **SBML** and libSBML

```
<species id="M_atp_c" name="ATP" compartment="c">
  <notes>
    <body xmlns="http://www.w3.org/1999/xhtml">
      <p>FORMULA: C10H12N5O13P3</p>
      <p>CHARGE: -4</p>
    </body>
  </notes>
</species>
```

New SBML Level 3 Species attributes

Hopefully, with the adoption of **SBML** FBC these species properties can be unified into a common format.

```
<species metaid="meta_M_atp_c␣" id="M_atp_c" name="ATP"
    compartment="Cytosol" boundaryCondition="false"
    fbc:charge="-4" fbc:chemicalFormula="C10H12N5O13P3"
    initialConcentration="0" hasOnlySubstanceUnits="false"
/>
```

Reaction definition and flux bounds

Old SBML Level 2 style reaction

```
<reaction id="R_GTHS" name="glutathione synthetase" reversible="false">
<notes>
<html:p>Abbreviation: R_GTHS</html:p>
<html:p>Synonyms: _0</html:p>
<html:p>EC Number: 6.3.2.3</html:p>
<html:p>SUBSYSTEM: Cofactor and Prosthetic Group Biosynthesis</html:p>
<html:p>Equation: [c] : atp + glucys + gly --&gt; adp + gthrd + h + pi</html:p>
<html:p>Confidence Level: 0</html:p>
<html:p>NCD</html:p>
<html:p>genes:</html:p>
<html:p>LOCUS:b2947#ABBREVIATION:gshB#ECNUMBERS:6.3.2.3#</html:p>
<html:p>proteins:</html:p>
<html:p>NAME:glutathione synthase#ABBREVIATION:GshB#</html:p>
<html:p>GENE ASSOCIATION: (b2947)</html:p>
</notes>
<listOfReactants>
<speciesReference species="M_atp_c" stoichiometry="1"/>
<speciesReference species="M_glucys_c" stoichiometry="1"/>
<speciesReference species="M_gly_c" stoichiometry="1"/>
</listOfReactants>
<listOfProducts>
<speciesReference species="M_adp_c" stoichiometry="1"/>
<speciesReference species="M_gthrd_c" stoichiometry="1"/>
<speciesReference species="M_h_c" stoichiometry="1"/>
<speciesReference species="M_pi_c" stoichiometry="1"/>
</listOfProducts>
<kineticLaw>
<math xmlns="http://www.w3.org/1998/Math/MathML">
<ci>FLUX_VALUE</ci>
</math>
<listOfParameters>
<parameter id="LOWER_BOUND" value="0" units="mmol_per_gDW_per_hr"/>
<parameter id="UPPER_BOUND" value="999999" units="mmol_per_gDW_per_hr"/>
<parameter id="OBJECTIVE_COEFFICIENT" value="0" />
<parameter id="FLUX_VALUE" value="0" units="mmol_per_gDW_per_hr"/>
</listOfParameters>
</kineticLaw>
</reaction>
```

New SBML Level 3 style reaction and flux bound

Please note that in order to maintain all the annotation encoded in the **SBML** L2 reaction `notes` an additional (tool specific) annotation is introduced i.e. 'KeyValueData'. This should be considered a transitional step until a generally accepted annotation system is adopted by the constraint based modelling community. Nevertheless, where (unambiguously) possible, the L2 annotation has been converted into a MIRIAM compliant form e.g. the 'EC number'.

```
<reaction metaid="meta_R_GTHS" id="R_GTHS" name="glutathione␣synthetase" reversible="false">
 <annotation>
  <listOfKeyValueData xmlns="http://pysces.sourceforge.net/KeyValueData">
   <data id="subsystem" type="string" value="Cofactor␣and␣Prosthetic␣Group␣Biosynthesis"/>
   <data id="name" type="string" value="glutathione␣synthase#ABBREVIATION:GshB#"/>
   <data id="gene_association" type="string" value="(b2947)"/>
   <data id="equation" type="string" value="[c]␣:␣atp␣+␣glucys␣+␣gly␣--&gt;␣adp␣+␣gthrd␣+␣h␣+␣pi"/>
   <data id="genes" type="string"/>
   <data id="proteins" type="string"/>
   <data id="locus" type="string" value="b2947#ABBREVIATION:gshB#ECNUMBERS:6.3.2.3#"/>
   <data id="abbreviation" type="string" value="R_GTHS"/>
   <data id="synonyms" type="string" value="_0"/>
   <data id="confidence_level" type="string" value="0"/>
  </listOfKeyValueData>
  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dcterms="http://purl.org/dc/terms/"
        xmlns:vCard="http://www.w3.org/2001/vcard-rdf/3.0#"
        xmlns:bqbiol="http://biomodels.net/biology-qualifiers/"
        xmlns:bqmodel="http://biomodels.net/model-qualifiers/">
   <rdf:Description rdf:about="#meta_R_GTHS">
    <bqbiol:is>
     <rdf:Bag>
      <rdf:li rdf:resource="http://identifiers.org/ec-code/6.3.2.3"/>
     </rdf:Bag>
    </bqbiol:is>
   </rdf:Description>
  </rdf:RDF>
 </annotation>
</reaction>
```

```
<fbc:listOfFluxBounds>
 <fbc:fluxBound fbc:id="R_GTHS_lower_bnd" fbc:reaction="R_GTHS" fbc:operation="greaterEqual"
    fbc:value="0"/>
 <fbc:fluxBound fbc:id="R_GTHS_upper_bnd" fbc:reaction="R_GTHS" fbc:operation="lessEqual"
    fbc:value="999999"/>
</fbc:listOfFluxBounds>
```

A Validation of SBML documents

A.1 Validation and consistency rules

This section summarizes all the conditions that must (or in some cases, at least *should*) be true of an SBML Level 3 Version 1 model that uses the Flux Balance Constraints package. We use the same conventions as are used in the SBML Level 3 Version 1 Core specification document. In particular, there are different degrees of rule strictness. Formally, the differences are expressed in the statement of a rule: either a rule states that a condition *must* be true, or a rule states that it *should* be true. Rules of the former kind are strict SBML validation rules—a model encoded in SBML must conform to all of them in order to be considered valid. Rules of the latter kind are consistency rules. To help highlight these differences, we use the following three symbols next to the rule numbers:

☑ A checked box indicates a *requirement* for SBML conformance. If a model does not follow this rule, it does not conform to the Flux Balance Constraints specification. (Mnemonic intention behind the choice of symbol: "This must be checked.")

▲ A triangle indicates a *recommendation* for model consistency. If a model does not follow this rule, it is not considered strictly invalid as far as the Flux Balance Constraints specification is concerned; however, it indicates that the model contains a physical or conceptual inconsistency. (Mnemonic intention behind the choice of symbol: "This is a cause for warning.")

★ A star indicates a strong recommendation for good modeling practice. This rule is not strictly a matter of SBML encoding, but the recommendation comes from logical reasoning. As in the previous case, if a model does not follow this rule, it is not strictly considered an invalid SBML encoding. (Mnemonic intention behind the choice of symbol: "You're a star if you heed this.")

The validation rules listed in the following subsections are all stated or implied in the rest of this specification document. They are enumerated here for convenience. Unless explicitly stated, all validation rules concern objects and attributes specifically defined in the Flux Balance Constraints package.

☞ For convenience and brievity, we use the shorthand "`fbc:x`" to stand for an attribute or element name `x` in the namespace for the Flux Balance Constraints package, using the namespace prefix `fbc`. In reality, the prefix string may be different from the literal "`fbc`" used here (and indeed, it can be any valid XML namespace prefix that the modeler or software chooses). We use "`fbc:x`" because it is shorter than to write a full explanation everywhere we refer to an attribute or element in the Flux Balance Constraints package namespace.

General rules about this package

fbc-10101 ☑ To conform to the Flux Balance Constraints package specification for SBML Level 3 Version 1, an SBML document must declare the use of the following XML Namespace: *[-25]*"`http://www.sbml.org/sbml/level3/version1/fbc/version1`". (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.1 on page 6.)

fbc-10102 ☑ Wherever they appear in an SBML document, elements and attributes from the Flux Balance Constraints package must be declared either implicitly or explicitly to be in the XML namespace "`http://www.sbml.org/sbml/level3/version1/fbc/version1`". (References: SBML Level 3 Package Specification for Flux Balance Constraints , Version 1, Section 3.1 on page 6.)

General rules about identifiers

fbc-10301 ☑ (Extends validation rule #10301 in the SBML Level 3 Version 1 Core specification.) Within a **Model** the values of the attributes `id` and `fbc:id` on every instance of the following classes of objects must be unique across the set of all `id` and `fbc:id` attribute values of all such objects in a model: the **Model** itself, plus all contained **FunctionDefinition**, **Compartment**, **Species**, **Reaction**, **SpeciesReference**, **ModifierSpeciesReference**, **Event**, and **Parameter** objects, plus

the **FluxBound** , **Objective** and **FluxObjective** objects defined by the Flux Balance Constraints package. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.2 on page 6.)

Rules for the extended SBML class

fbc-20101 ☑ In all SBML documents using the Flux Balance Constraints package, the **SBML** object must include a value for the attribute `fbc:required` attribute. (References: SBML Level 3 Version 1 Core, Section 4.1.2.)

fbc-20102 ☑ The value of attribute `fbc:required` on the **SBML** object must be of the data type `boolean`. (References: SBML Level 3 Version 1 Core, Section 4.1.2.)

fbc-20103 ☑ The value of attribute `fbc:required` on the **SBML** object must be set to "`false`". (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.1 on page 6.)

Rules for extended Model object

fbc-20201 ☑ There may be at most one instance of each of the following kinds of objects within a **Model** object using Flux Balance Constraints: **ListOfFluxBounds** and **ListOfObjectives** . (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.3 on page 7.)

fbc-20202 ☑ The various **ListOf**__ subobjects with an **Model** object are optional, but if present, these container object must not be empty. Specifically, if any of the following classes of objects are present on the **Model**, it must not be empty: **ListOfFluxBounds** and **ListOfObjectives** . (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.3 on page 7.)

fbc-20203 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfFluxBounds** container object may only contain **FluxBound** objects. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.3 on page 7.)

fbc-20204 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfObjectives** container object may only contain **Objective** objects. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.3 on page 7.)

fbc-20205 ☑ A **ListOfFluxBounds** object may have the optional `metaid` and `sboTerm` defined by SBML Level 3 Core. No other attributes from the SBML Level 3 Core namespace or the Flux Balance Constraints namespace are permitted on a **ListOfFluxBounds** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.3 on page 7.)

fbc-20206 ☑ A **ListOfObjectives** object may have the optional attributes `metaid` and `sboTerm` defined by SBML Level 3 Core. Additionally the **ListOfObjectives** must contain the attribute `activeObjective`. No other attributes from the SBML Level 3 Core namespace or the Flux Balance Constraints namespace are permitted on a **ListOfObjectives** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.3 on page 7.)

fbc-20207 ☑ The value of attribute `fbc:activeObjective` on the **ListOfObjectives** object must be of the data type `FbcSIdRef`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.2.2 on page 6).

fbc-20208 ☑ The value of attribute `fbc:activeObjective` on the **ListOfObjectives** object must be set to the `FbcSId` of an existing **Objective** . (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.2.2 on page 6.)

Rules for extended Species object

fbc-20301 ☑ A **SBML Species** object may have the optional attributes `fbc:charge` and `fbc:chemicalFormula`. No other attributes from the Flux Balance Constraints namespaces are permitted on a **Species**. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.4 on page 8)

fbc-20302 ☑ The value of attribute `fbc:charge` on the **SBML Species** object must be of the data type `integer`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.4 on page 8).

fbc-20303 ☑ The value of attribute `fbc:chemicalFormula` on the **SBML Species** object must be set to the `string`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.4 on page 8.)

Rules for FluxBound object

fbc-20401 ☑ A **FluxBound** object may have the optional SBML Level 3 Core attributes `metaid` and `sboTerm`. No other attributes from the SBML Level 3 Core namespaces are permitted on a **FluxBound** . (References: SBML Level 3 Version 1 Core, Section 3.2.)

fbc-20402 ☑ A **FluxBound** object may have the optional SBML Level 3 Core subobjects for notes and annotations. No other elements from the SBML Level 3 Core namespaces are permitted on a **FluxBound** . (References: SBML Level 3 Version 1 Core, Section 3.2.)

fbc-20403 ☑ A **FluxBound** object must have the required attributes `fbc:reaction`, `fbc:operation` and `fbc:value`, and may have the optional attributes `fbc:id` and `fbc:name`. No other attributes from the SBML Level 3 Flux Balance Constraints namespace are permitted on a **FluxBound** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, **??** on page ??.)

fbc-20404 ☑ The attribute `fbc:reaction` in **FluxBound** must be of the data type `SIdRef`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.5 on page 8.)

fbc-20405 ☑ The attribute `fbc:name` in **FluxBound** must be of the data type `string`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.5 on page 8.)

fbc-20406 ☑ The attribute `fbc:operation` in **FluxBound** must be of the data type `enum`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.5 on page 8.)

fbc-20407 ☑ The attribute `fbc:value` in **FluxBound** must be of the data type `double`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.5 on page 8.)

fbc-20408 ☑ The value of the attribute `fbc:reaction` in a **FluxBound** object must be the identifier of an existing **Reaction** object defined in the enclosing **Model** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.5 on page 8.)

fbc-20409 ☑ The value of the attribute `fbc:operation` in a **FluxBound** object must be one of "`lessEqual`", "`greaterEqual`", "`less`","`greater`" or "`equal`". (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.5 on page 8.)

fbc-20410 ☑ The combined set of all **FluxBound** 's with identical value for `fbc:reaction` must be consistent. That is while it is possible to define a lower and an upper bound for a reaction, it is not possible to define multiple lower or uppoer bounds. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.5 on page 8.)

Rules for Objective object

fbc-20501 ☑ A **Objective** object may have the optional SBML Level 3 Core attributes `metaid` and `sboTerm`. No other attributes from the SBML Level 3 Core namespaces are permitted on a **Objective** . (References: SBML Level 3 Version 1 Core, Section 3.2.)

fbc-20502 ☑ A **Objective** object may have the optional SBML Level 3 Core subobjects for notes and annotations. No other elements from the SBML Level 3 Core namespaces are permitted on a **Objective** . (References: SBML Level 3 Version 1 Core, Section 3.2.)

fbc-20503 ☑ A **Objective** object may have the required attribute `fbc:id` and `fbc:type` and the optional attribute `fbc:name`. No other attributes from the SBML Level 3 Flux Balance Constraints namespace are permitted on a **Objective** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

fbc-20504 ☑ The attribute `fbc:name` in **Objective** must be of the data type `string`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

fbc-20505 ☑ The attribute `fbc:type` in **Objective** must be of the data type `enum`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

fbc-20406 ☑ The value of the attribute `fbc:type` in a **Objective** object must be one of "`minimize`" or "`maximize`". (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

fbc-20507 ☑ A **Objective** must have one and only one instance of the **ListOfFluxes** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

fbc-20508 ☑ The **ListOfFluxes** subobject with a **Objective** object must not be empty. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

fbc-20509 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfFluxes** container object may only contain **Objective** objects. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

fbc-20510 ☑ A **ListOfFluxes** object may have the optional `metaid` and `sboTerm` defined by SBML Level 3 Core. No other attributes from the SBML Level 3 Core namespace or the Flux Balance Constraints namespace are permitted on a **ListOfFluxes** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.6 on page 9.)

Rules for FluxObjective object

fbc-20601 ☑ A **FluxObjective** object may have the optional SBML Level 3 Core attributes `metaid` and `sboTerm`. No other attributes from the SBML Level 3 Core namespaces are permitted on a **FluxObjective** . (References: SBML Level 3 Version 1 Core, Section 3.2.)

fbc-20602 ☑ A **FluxObjective** object may have the optional SBML Level 3 Core subobjects for notes and annotations. No other elements from the SBML Level 3 Core namespaces are permitted on a **FluxObjective** . (References: SBML Level 3 Version 1 Core, Section 3.2.)

fbc-20603 ☑ A **FluxObjective** object must have the required attributes `fbc:reaction` and `fbc:coefficient`, and may have the optional attributes `fbc:id` and `fbc:name`. No other attributes from the SBML Level 3 Flux Balance Constraints namespace are permitted on a **FluxObjective** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.7 on page 11.)

fbc-20604 ☑ The attribute `fbc:name` in **FluxObjective** must be of the data type `string`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.7 on page 11.)

fbc-20605 ☑ The value of the attribute `fbc:reaction` of a **FluxObjective** object must conform to the syntax of the SBML data type `SIdRef`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.7 on page 11.)

fbc-20606 ☑ The value of the attribute `fbc:reaction` of a **FluxObjective** object must be the identifier of an existing **Reaction** object defined in the enclosing **Model** object. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.7 on page 11.)

fbc-20607 ☑ The value of the attribute `fbc:coefficient` of a **FluxObjective** object must conform to the syntax of the SBML data type `double`. (References: SBML Level 3 Package Specification for Flux Balance Constraints, Version 1, Section 3.7 on page 11.)

Acknowledgments

We thank especially Herbert Sauro (University of Washington, USA), Ursula Kummer (Heidelberg University, Germany) and Michael Hucka (CalTech, USA) as well as Neil Swainston, Keiran Smallbone for invaluable discussions and while developing the Flux Balance Constraints package proposal and specification. We also thank all those who contributed ideas and engaged with us in discussions over the course of its development.

References

Becker, S. A., Feist, A. M., Mo, M. L., Hannum, G., Palsson, B. O., and Herrgard, M. J. (2007). Quantitative prediction of cellular metabolism with constraint-based models: the cobra toolbox. *Nat. Protocols*, 2(3):727–738. 1750-2799 10.1038/nprot.2007.99 10.1038/nprot.2007.99.

Biron, P. V. and Malhotra, A. (2000). XML Schema part 2: Datatypes (W3C candidate recommendation 24 October 2000). Available via the World Wide Web at http://www.w3.org/TR/xmlschema-2/.

Bornstein, B. J., Keating, S. M., Jouraku, A., and Hucka, M. (2008). Libsbml: an api library for sbml. *Bioinformatics*, 24(6):880–881.

Eriksson, H.-E. and Penker, M. (1998). *UML Toolkit*. John Wiley & Sons, New York.

Fallside, D. C. (2000). XML Schema part 0: Primer (W3C candidate recommendation 24 October 2000). Available via the World Wide Web at http://www.w3.org/TR/xmlschema-0/.

Hill, E. A. (1900). On a system of indexing chemical literature; adopted by the classification division of the u. s. patent office. *Journal of the American Chemical Society*, 22(8):478–494.

Hill, E. A. (2012). Wikipedia: The hill system. Available via the World Wide Web at http://en.wikipedia.org/wiki/Hill_system.

Hucka, M., Bergmann, F. T., Hoops, S., Keating, S. M., Sahle, S., and Wilkinson, D. J. (2011). Available via the World Wide Web at http://sbml.org/Documents/Specifications.

Hucka, M., Finney, A., Sauro, H., Bolouri, H., Doyle, J., Kitano, H., Arkin, A., Bornstein, B., Bray, D., Cornish-Bowden, A., Cuellar, A., Dronov, S., Gilles, E., Ginkel, M., Gor, V., Goryanin, I., Hedley, W., Hodgman, T., Hofmeyr, J., Hunter, P., Juty, N., Kasberger, J., Kremling, A., Kummer, U., Novère, N. L., Loew, L., Lucio, D., Mendes, P., Minch, E., Mjolsness, E., Nakayama, Y., Nelson, M., Nielsen, P., Sakurada, T., Schaff, J., Shapiro, B., shimizu, T., Spence, H., Stelling, J., Takahashi, K., Tomita, M., Wagner, J., and Wang, J. (2003). The Systems Biology Markup Language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–31.

Oberhardt, M. A., Palsson, B. Ø., and Papin, J. A. (2009). Applications of genome-scale metabolic reconstructions. *Molecular Systems Biology*, 5:320.

Oestereich, B. (1999). *Developing Software with UML: Object-Oriented Analysis and Design in Practice*. Addison-Wesley.

Olivier, B. G. (2012). CBMPy: constraint based modelling in python. Available via the World Wide Web at http://pysces.sf.net/cbm.

Olivier, B. G., Rohwer, J. M., and Hofmeyr, J.-H. S. (2005). Modelling cellular systems with pysces. *Bioinformatics*, 21(4):560–561.

Orth, J. D., Thiele, I., and Palsson, B. Ø. (2010). What is flux balance analysis? *Nature Biotechnology*, 28(3):245–248.

Reed, J. L., Vo, T. D., Schilling, C. H., and Palsson, B. O. (2003). An expanded genome-scale model of escherichia coli k-12 (ijr904 gsm/gpr). *Genome Biol*, 4(9):R54.

SBML Team (2010). The SBML issue tracker. Available via the World Wide Web at http://sbml.org/issue-tracker.

Schellenberger, J., Park, J., Conrad, T., and Palsson, B. (2010). Bigg: a biochemical genetic and genomic knowledgebase of large scale metabolic reconstructions. *BMC Bioinformatics*, 11(1):213.

Thompson, H. S., Beech, D., Maloney, M., and Mendelsohn, N. (2000). XML Schema part 1: Structures (W3C candidate recommendation 24 October 2000). Available online via the World Wide Web at the address http://www.w3.org/TR/xmlschema-1/.