SBML Level 3 Package Specification

Qualitative Models

Claudine Chaouiya
chaouiya@igc.gulbenkian.pt
IGC Rua da Quinta Grande 6
P-2780-156 Oeiras
Portugal

Martijn P. van Iersel
mvpi@ebi.ac.uk
European Bioinformatics Institute
Cambridgeshire
UK

Sarah M Keating

skeating@ebi.ac.uk
European Bioinformatics Institute
Cambridgeshire
UK

29 Feb 2012

Version 1.0 (Draft)

This is a working draft of the specification for the SBML Level 3 package "qual". It is not a normative document. Please send comments and other feedback to the Package Working Group mailing list, sbml-qual@lists.sourceforge.net.

The latest release, past releases, and other materials related to this specification are available at
http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Qualitative_Models

This release of the specification is available at
http://sbml.org/images/a/a7/SBML-L3-qual-specification_0.1.pdf

Contents

1 Introduction

1.1 Motivation

Quantitative methods for modelling biological networks require an in-depth knowledge of the biochemical reactions and their stoichiometric and kinetic parameters. In many cases, this knowledge is missing. This has led to the development of several qualitative modelling methods using information such as gene expression data coming from functional genomic experiments. Qualitative models are typically based on the definition of *regulatory* or *influence graph*. The components of these models differ from species and reactions used in current SBML models. For example, qualitative models typically associate discrete levels of activities with entity pools; the processes involving them cannot be described as reactions per se but rather as transitions between states. Boolean networks, logical models and some Petri nets are the most used qualitative formalisms in biology. Despite differences from traditional SBML models, it is desirable to bring these classes of models under a common format scheme. The purpose of this Qualitative Models package for SBML Level 3 is to support qualitative models into SBML.

2 Background and context

After several attempts to use the existing SBML L2 format, a decision was made to develop an extension for SBML L3. A first proposal written in August 2008 by Duncan Berenguier and Nicolas Le Novère was discussed during a meeting on the 12th and 13th of August 2008[1]. This meeting led to the release of a document (L3F_extention_draft_1.2.pdf) which is a revision of a previous proposal for this package. A summary of the meeting is available at http://www.ebi.ac.uk/compneur/xwiki/bin/view/SBML/L3F, and a document A second meeting was held at in November 2010 (see http://compbio.igc.gulbenkian.pt/nmd/node/30, for the program and participants). A revised version of the proposal was discussed during this meeting. This document accounts for the outcomes of the meeting discussions and of following exchanges.

[1] Nicolas Le Novère (SBML), Sarah Keating (SBML), Nicolas Rodriguez (SBML), Denis Thieffry (GINsim), Duncan Berenguier (GINsim), Aurélien Naldi (GINsim), Claudine Chaouiya (GINsim, Petri nets), Tomas Helikar (Chemchains), Ioannis Xenarios (SQUAD), Alessandro Di Cara (SQUAD), Mathias John (PiML), Dagmar Koehn (PiML)

3 Package syntax and semantics

In this section, we define the syntax and semantics of the Hierarchical Model Composition package for SBML Level 3 Version 1. We expound on the various data types and constructs defined in this package, then in Section 4, 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 Qualitative Models package for SBML Level 3 Version 1:

"`http://www.sbml.org/sbml/level3/version1/qual/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 Qualitative Models 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 Qualitative Models package:

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

3.2 Primitive data types

Section 3.1 of the SBML Level 3 specification defines a number of primitive data types and also uses a number of XML Schema 1.0 data types (**?**). We assume and use some of them in the rest of this specification, specifically `boolean`, `ID`, `SId`, `SIdRef`, `UnitSId`, `UnitSIdRef`, and `string`. The Qualitative Model package defines other primitive types; they are described below.

3.2.1 Type `temporisationType`

The `temporisationType` is an enumeration of values used to indicate the updating policy used by a **Transition**. The possible values are `timer`, `priority`, `sustain`, `proportion` and `rate`.

3.2.2 Type `sign`

The `sign` is an enumeration of values used to indicate direction of an **Input** within the system. The possible values are `positive`, `negative` and `dual`.

3.2.3 Type `transitionInputEffect`

The `transitionInputEffect` is an enumeration of values used to indicate the effect of an **Input Transition** within the system. The possible values are `none` and `consumption`.

3.2.4 Type `transitionOutputEffect`

The `transitionOutputEffect` is an enumeration of values used to indicate the effect of an **Output Transition** within the system. The possible values are `production`, `assignmentLevel` and `assignmentSymbol`.

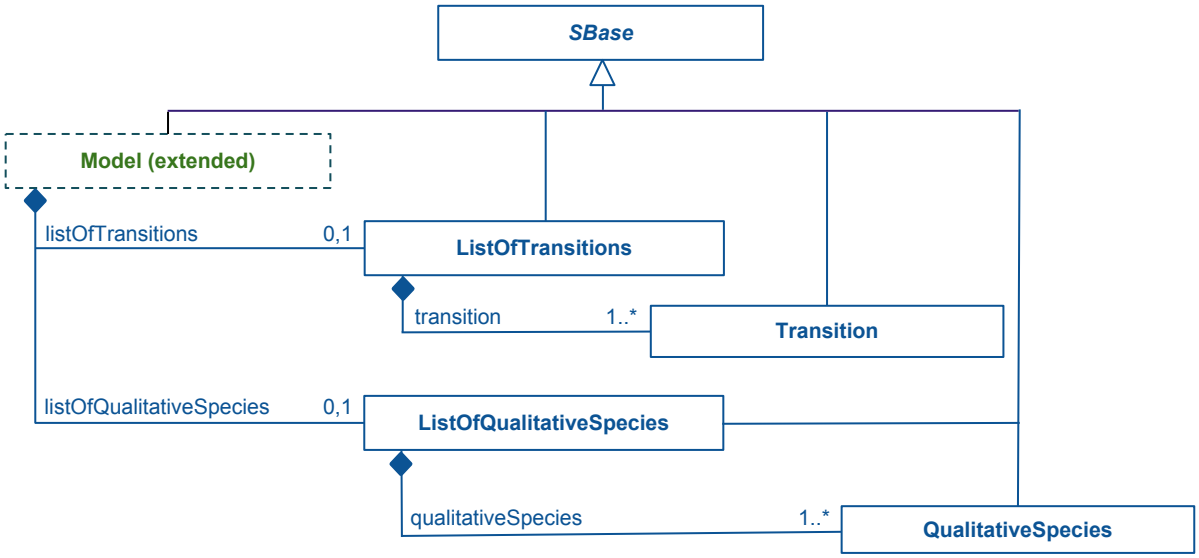

Figure 1: *The definitions of the extended* ***Model*** *class. In other respects,* ***Model*** *remains defined as in the SBML Level 3 Core specification.*

3.3 The extended Model class

The extension of SBML Level 3 Core's **Model** class is relatively straightforward: the Qualitative Models Package adds two lists, one for holding qualitativeSpecies (**listOfQualitativeSpecies**, of class **ListOfQualitativeSpecies**), and the other for holding transitions (**listOfTransitions**, of class **ListOfTransitions**). Figure **??** on page ?? provides the UML diagram. The **Model** element may contain at most one **ListOfQualitativeSpecies**, which must contain at least one **QualitativeSpecies**. It may also contain at most one **ListOfTransitions** which must contain at least one **Transition**.The **QualitativeSpecies** class and the **Transition** class are defined in Sections Section 3.4 and Section 3.5 respectively.

3.4 The QualitativeSpecies class

Similarly to the **Species** in SBML, the components of qualitative models refer to pools of entities that are considered indistinguishable and are each located in a specific **Compartment**. However, here components are characterised by their qualitative influences rather than by taking parts into reactions. Therefore, we define the **QualitativeSpecies** element to represent such pools of entities.

A **QualitativeSpecies** describes a pool of indistinguishable entities in a **Compartment**. It is associated with either a `level` or a `symbolValue` from its **ListOfSymbolicValues**. TODO: NEED EXPLANATION OF WHAT THESE STAND FOR These objects classes are defined in Figure 2.

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

The `id` attribute takes a required value of type `SId`. The `id` is used as an identifier for the particluar **QualitativeSpecies**. A **QualitativeSpecies** also has an optional `name` attribute of type `string`. The `id` and `name` attributes may be used in the same manner as on SBML Level 3 Core objects; see Section 3.3.2 of the SBML Level 3 Version 1 Core specification for more information.

The `compartment` *attribute*

The required attribute `compartment`, of type `SIdRef`, is used to identify the compartment in which the qualitativeSpecies is located. The attribute's value must be the identifier of an existing **Compartment** objecyt in the model. This attribute is comparable with the `compartment` attribute on the **Species** element.

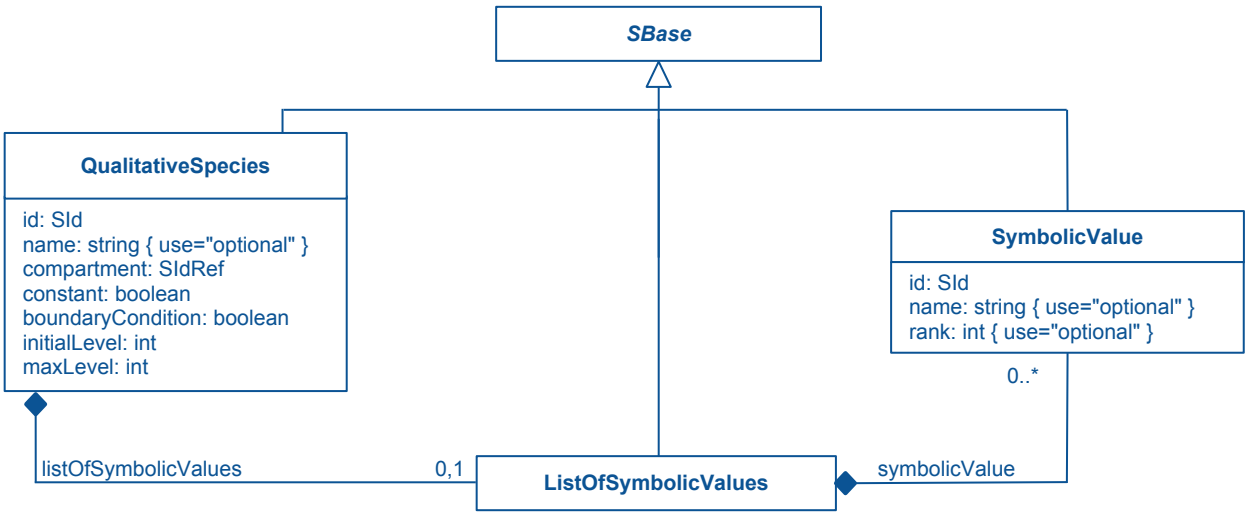

Figure 2: *The definitions of the* ***QualitativeSpecies*** *class.*

The `constant` ***and*** `boundaryCondition` ***attributes***

These attributes are treated as in **Species** elements.

The `initialLevel` ***attribute:***

The `initialLevel` is an `integer` that defines the initial level of the **QualitativeSpecies** in its **Compartment**. This attribute is optional.

The `maxLevel` ***attribute:***

The `maxLevel` is an `integer` that sets the maximal level of the **QualitativeSpecies**. This attribute is optional.

3.4.1 The SymbolicValue class

3.5 The Transition class

3.5.1 The Input class

3.5.2 The Output class

3.5.3 The FunctionTerm class

3.6 Not yet worked on

This proposal defines the following new main elements: **QualitativeSpecies**, **SymbolicValue**, **Transition**, **Input**, **Output**, **FunctionTerm**, **DefaultTerm**. All inherit from **SBase** and all, except **DefaultTerm**, are contained into the corresponding **ListOf***ElementName* element, which inherits from **ListOf**.

The SBML element **Model** is extended to include the new elements **ListOfQualitativeSpecies** and **ListOfTransitions**. The SBML elements **EventAssignment** and **AssignmentRule** are extended to refer to **QualitativeSpecies**.

The overall structure of this extension is described in Figure **??**.

Extension of the Model element

The SBML element **Model** is extended to contain at most one **ListOfQualitativeSpecies** and at most one **ListOfTransitions**.

Definition of QualitativeSpecies

Definition of SymbolicValue

The **QualitativeSpecies** element may contain at most one **ListOfSymbolicValues** that contains zero or more **SymbolicValue**s. An empty list is allowed, and useful for e.g. adding annotations. The **SymbolicValue** element defines a non instantiated parameter. Such symbols may represent the different solutions of piecewise linear differential equations, along with different thresholds.

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

These attributes are used according to the SBML L3.1 Section 3.3. The attribute `id` is mandatory and `name` is optional.

The `rank` *attribute*

The `rank` is an `integer` that defines the position of the symbol in the **ListOfSymbolicValues**. This attribute is optional.

Definition of Transition

A **Transition** element contains at most one **ListOfInputs**, exactly one **ListOfOutputs** and one **ListOfFunctionTerms**.

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

These attributes are used according to the SBML L3.1 Section 3.3. They are both optional.

The `temporisationType` *attribute:*

The `temporisationType` is an `enumeration` the "temporisation" of the **Transition**, that is the updating policy associated with the **Transition**. It can be set to `timer`, `priority`, `sustain`, `proportion` or `rate`. This attribute is optional.

Definition of Input

The **ListOfInputs** contains zero or more **Input**s. A transition with zero inputs can be useful to define an initial assignment, where the state of an output depends on a function but not on any input values. An empty list is allowed, and useful for e.g. adding annotations. Each **Input** refers to a **QualitativeSpecies** that participates to the corresponding **Transition**.

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

These attributes are used according to the SBML L3.1 Section 3.3. They are both optional.

The `qualitativeSpecies` *attribute:*

The `qualitativeSpecies` is a `SIdRef` referring to a **QualitativeSpecies**. This attribute is mandatory.

The `thresholdLevel` *and* `thresholdSymbol` *attributes:*

The `thresholdLevel` is an `integer` and `thresholdSymbol` is a `SIdRef`. They are optional and exclusive.

The `transitionEffect` *attribute:*

The `transitionEffect` is an `enumeration` describing how the `qualitativeSpecies` is affected by the **Transition**. On inputs, the value of `transitionEffect` can be either `none` or `consumption`. (See section Interpreting transitions). This attribute is mandatory.

The `sign` *attribute*

The `sign` is an `enumeration` that can be used as an indication on whether the contribution of this input is positive, negative, or both. Thus, possible values can be either `positive`, `negative` or `dual`. The sign is usually used for visualization purposes only. This attribute is optional.

Definition of Output

The **ListOfOutputs** contains at least one **Output**. Each **Output** refers to a **QualitativeSpecies** that participates to the corresponding **Transition**.

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

These attributes are used according to the SBML L3.1 Section 3.3. They are both optional.

The `qualitativeSpecies` *attribute:*

The `qualitativeSpecies` is a `SIdRef` referring to a **QualitativeSpecies**. This attribute is mandatory.

The `outputLevel` *attribute:*

The `outputLevel` is an `integer` used along with the `transitionEffect` set to `production` to specify the effect of the **Transition** on the corresponding **QualitativeSpecies**. This attribute is optional.

The `transitionEffect` *attribute:*

The `transitionEffect` is an `enumeration` describing how the `qualitativeSpecies` is affected by the **Transition**. On outputs, the value of `transitionEffect` can be `production`, `assignmentLevel` or `assignmentSymbol`. (See section Interpreting transitions). This attribute is mandatory.

Definitions of FunctionTerm and DefaultTerm

The **ListOfFunctionTerms** may contain any number of **FunctionTerm** elements, and exactly one **DefaultTerm**. Each term is associated with a result (symbolic or level) and a **FunctionTerm** is associated with a Boolean function inside a **Math** element. The disjunction of the terms defines the *qualitative function* associated with a **Transition**.

The `resultLevel` *and* `resultSymbol` *attributes:*

The result of the term is described by a `resultLevel` or a `resultSymbol`. Both are optional, but one of them must be defined.

The `resultLevel` is an `integer` describing a level. The `resultSymbol` is a `SIdRef` referring to a **SymbolicValue**.

The `temporisationValue` *attribute and the TemporisationMath element:*

The attribute `temporisationValue` and the element **TemporisationMath** allow the specification of the "temporisation" of the **Transition** under the corresponding **FunctionTerm**. Both are optional. Depending on the value of the `temporisationType`, either one or both could be used.

The `temporisationValue` is a `double`. The element **TemporisationMath** holds a MathML function returning a `double`.

The Math element:

Each **FunctionTerm** holds a Boolean function encoded in a **Math** element, using the subset of MathML 2.0 as defined in SBML L3v1 Section 3.4.6. This element encodes the conditions under which the **FunctionTerm** is selected.

Interpreting transitions

Determining the result of a qualitative function:

The qualitative function associated with a **Transition** is encoded by a **ListOfFunctionTerms**. The **Transition** contains exactly one **DefaultTerm** describing the result of the function by default. A **FunctionTerm** in a **Transition** defines a result (`resultLevel` or `resultSymbol`) as well as the conditions (**Math** element) under which this **FunctionTerm** is selected. The conditions are encoded in MathML as a Boolean function that returns `true` if the conditions are fulfilled. Several **FunctionTerm** elements can have the same result; the qualitative function is then defined as the disjunction of their conditions.

Encoding the conditions:

To encode the conditions of the qualitative function, one can use **`ci`** elements of MathML to refer to SBML elements. A **`ci`** referring to the `id` of a **QualitativeSpecies** then refers to the level or the symbol of this **QualitativeSpecies**, while a **`ci`** referring to the `id` of an **Input** then refers to the `thresholdLevel` or the `thresholdSymbol` of this **Input**.

The transitionEffect:

The **Input** and **Output** elements refer to a **QualitativeSpecies** using the attribute `qualitativeSpecies`. They are defined with a `transitionEffect` attribute that takes one of the following values :

- `none`: Neither the level nor the symbol associated to the `qualitativeSpecies` is modified.
- `consumption`: The level of the `qualitativeSpecies` is decreased by the `resultLevel` of the selected term possibly modified by the `thresholdLevel` of the **Input**.
- `production`: The level of the `qualitativeSpecies` is increased by the `resultLevel` of the selected term possibly modified by the `level` of the **Output**.
- `assignmentLevel`: The level of the `qualitativeSpecies` is set to the `resultLevel` of the selected term.
- `assignmentSymbol`: The symbol associated to the `qualitativeSpecies` is set to the `resultSymbol` of the selected term.

4 Examples

4.1 Graphical and typographical conventions

As this proposal covers various formalisms, the examples are labeled with a token indicating the corresponding formalism : **All** all formalisms, **PN** Petri nets, **LRG** logical regulatory networks or **SYM** symbolic relationships.

Simple Logical Regulatory Graph

LRG The following example shows a simple LRG with 3 regulators A, B and C, where A can take three values ($A = \{0,1,2\}$), and B,C are Boolean. The logical functions are the following : $B := 1\ if\ A \geq 1$, $C := 1\ if\ B \geq 1$, $A := 2\ if\ (A \geq 1\ and\ A < 2)\ or\ C \geq 1; A := 1\ if\ A < 1\ and\ C \geq 1; A := 0\ otherwise.$

Listing 1: *Logical Regulatory Graph example*

```
<?xml version="1.0" encoding="UTF8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1" level="3" version="1" xmlns:qual="http://www.sbml.org/sbml/leve
  <model id="example">
    <listOfCompartments>
      <compartment id="cytosol" name="cytosol"/>
      <compartment id="nucleus" name="nucleus"/>
    </listOfCompartments>
    <qual:listOfQualitativeSpecies xmlns="http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Qualitative_Models">
      <qualitativeSpecies id="A" maxLevel="2" compartment="cytosol"/>
      <qualitativeSpecies id="B" maxLevel="1" compartment="cytosol"/>
      <qualitativeSpecies id="C" maxLevel="1" compartment="nucleus"/>
    </qual:listOfQualitativeSpecies>
    <qual:listOfTransitions xmlns="http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Qualitative_Models">
      <transition id="tr_B">
        <listOfInputs>
          <input id="theta_B_A" qualitativeSpecies="A" thresholdLevel="1" transitionEffect="none" sboTerm="SBO:0000170
        </listOfInputs>
        <listOfOutputs>
          <output qualitativeSpecies="B" transitionEffect="assignmentLevel"/>
        </listOfOutputs>
        <listOfFunctionTerms>
          <functionTerm resultLevel="1">
            <math> <!-- A >= 1-->
              <apply>
                <geq/>
                <ci>A</ci>
                <ci>theta_B_A</ci>
              </apply>
            </math>
          </functionTerm>
          <defaultTerm resultLevel="0"/>
        </listOfFunctionTerms>
      </transition>
      <transition id="tr_A">
        <listOfInputs>
          <input id="theta_A_A1" qualitativeSpecies="A" thresholdLevel="1" transitionEffect="none" sboTerm="SBO:000017
          <input id="theta_A_A2" qualitativeSpecies="A" thresholdLevel="2" transitionEffect="none" sboTerm="SBO:000017
          <input id="theta_A_C" qualitativeSpecies="C" thresholdLevel="1" transitionEffect="none" sboTerm="SBO:0000170
        </listOfInputs>
        <listOfOutputs>
          <output qualitativeSpecies="A" transitionEffect="assignmentLevel"/>
        </listOfOutputs>
```

```
        <listOfFunctionTerms>
          <functionTerm resultLevel="2">
            <math> <!-- (A >= 1 and A < 2) or C < 1 -->
            <apply>
              <or/>
              <apply>
                <and/>
                <apply>
                  <geq/>
                  <ci>A</ci>
                  <ci>theta_A_A1</ci>
                </apply>
                <apply>
                  <lt/>
                  <ci>A</ci>
                  <ci>theta_A_A2</ci>
                </apply>
              </apply>
              <apply>
                <lt/>
                <ci>C</ci>
                <ci>theta_A_C</ci>
              </apply>
            </apply>
          </math>
        </functionTerm>
        <functionTerm resultLevel="1">
          <math> <!-- A < 1 and C >= 1 -->
            <apply>
              <and/>
              <apply>
                <lt/>
                <ci>A</ci>
                <ci>theta_A_A</ci>
              </apply>
              <apply>
                <geq/>
                <ci>C</ci>
                <ci>theta_A_C</ci>
              </apply>
            </apply>
          </math>
        </functionTerm>
        <defaultTerm resultLevel="0"/>
      </listOfFunctionTerms>
    </transition>
    <transition id="tr_C">
      <listOfInputs>
        <input id="theta_C_B" qualitativeSpecies="B" thresholdLevel="1" transitionEffect="none" sboTerm="SBO:0000169"
      </listOfInputs>
      <listOfOutputs>
        <output qualitativeSpecies="C" transitionEffect="assignmentLevel"/>
      </listOfOutputs>
      <listOfFunctionTerms>
        <functionTerm resultLevel="1">
          <math> <!-- B >= 1 -->
            <apply>
              <geq/>
              <ci>B</ci>
```

```
            <ci>theta_C_B</ci>
          </apply>
        </math>
      </functionTerm>
      <defaultTerm resultLevel="0"/>
    </listOfFunctionTerms>
  </transition>
 </qual:listOfTransitions>
</model>
</sbml>
```

Simple Petri net

PN The following example shows a simple Petri net, containing 4 places A, B, C and D with one transition $t1$.

Listing 2: *Petri net example*

```
<?xml version="1.0" encoding="UTF-8"?>
<sbml xmlns="http://www.sbml.org/sbml/level3/version1" level="3" version="1" xmlns:qual="http://www.sbml.org/sbml/leve
  <model id="PN_exemple">
    <listOfCompartments>
      <compartment id="default" />
    </listOfCompartments>
    <qual:listOfQualitativeSpecies xmlns="http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Qualitative_Models">
      <qualitativeSpecies id="A" compartment="default" initialLevel="2" />
      <qualitativeSpecies id="B" compartment="default" initialLevel="4" />
      <qualitativeSpecies id="C" compartment="default" initialLevel="2" />
      <qualitativeSpecies id="D" compartment="default" initialLevel="3" />
    </qual:listOfQualitativeSpecies>
    <qual:listOfTransitions xmlns="http://sbml.org/Community/Wiki/SBML_Level_3_Proposals/Qualitative_Models">
      <transition id="t1">
        <listOfInputs>
          <input id="t1_A" qualitativeSpecies="A" thresholdLevel="2" transitionEffect="consumption" />
          <input id="t1_B" qualitativeSpecies="B" thresholdLevel="1" transitionEffect="consumption" />
        </listOfInputs>
        <listOfOutputs>
          <output qualitativeSpecies="C" level="1" transitionEffect="production" />
          <output qualitativeSpecies="D" level="2" transitionEffect="production" />
        </listOfOutputs>
        <listOfFunctionTerms>
          <functionTerm result="1">
            <math> <!-- A >= 2 and B >= 1 -->
              <apply>
                <and />
                <apply>
                  <geq />
                  <ci>A</ci>
                  <ci>t1_A</ci>
                </apply>
                <apply>
                  <geq />
                  <ci>A</ci>
                  <ci>t1_B</ci>
                </apply>
              </apply>
            </math>
          </functionTerm>
```

```
          <defaultTerm result="0" />
        </listOfFunctionTerms>
      </transition>
    </qual:listOfTransitions>
  </model>
</sbml>
```

5 Best practices

All To be valid, the SBML root element must express the requirement of this package: `<sbml ... qual:required="true" ... >`.

PN In Petri nets the initial conditions are part of the model, the initialLevel must be defined. To represent unbounded places, the maxLevel should be not specified.

LRG Discussions are still ongoing about the possible (but some times convenient to avoid cumbersome descriptions) incoherency of the **FunctionTerm** elements. For the moment, here are the guidelines to ensure coherent definitions:

- The **FunctionTerm** elements of all the transitions targeting the same output should be "coherent": the conditions of two **FunctionTerm** elements, leading to different effects on the level/symbol of the output, should not be fulfilled at the same time(i.e. they should be exclusive).
- If several **FunctionTerm** elements lead to the same effect on the level/symbol of the same output, then the importing tool should consider the disjunction (OR) on the conditions of the terms.

LRG To declare external nodes (ones that have no Boolean expression/truth table associated with them), one should set the attribute boundaryCondition of the **QualitativeSpecies** to TRUE:

```
<qualitativeSpecies id="EGF" maxLevel="1" boundaryCoundition="true"
                                                compartment="extracellular"/>
```

LRG To declare a "delay" node, which is specified to delay its state update for k iterations, one should set, for all the **Transition** elements having this node as their (unique) output, the attribute temporisationType to the value `timer` and the temporisationValue to k.

LRG To declare a "sustain" node, which is specified to sustain (i.e., to remain in) its latest state for the next k iterations, one should set, for all the **Transition** elements having this node as their (unique) output, the attribute temporisationType to the value `sustain` and the temporisationValue to k.

A Validation of SBML documents

Acknowledgments