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 NEED CITATION. 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 `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`.

3.2.2 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`.

3.2.3 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`.

3.3 Qualitative modelling

Before describing the classes and their attributes that have been used by this Qualitative Models Specification it is worth clarifying the intended meaning of some of the terms used.

3.3.1 Levels and symbols

The entities being modelled have a *level* associated with them that indicate the current state of the entity.

A *level* may be a boolean but may also represent more than two states and thus is considered to be an integer. In the case of the entity being a boolean its allowed levels would be “`0`” or “`1`”; but in other cases it may have any number of levels i.e. integer values up to and including a maximum.

3.3.2 Transitions

Qualitative Models consider *transitions* between the levels/symbols of the entities involved in the model. This may involve the level of an entity being increased or decreased by a fixed amount; the level/symbol remaining unchanged; or the level/symbol being reassigned to an alternate value. Transitions occur when a set of conditions is met. These conditions may involve the levels/symbols falling above or below a given *threshold*. A simple example of this is the case where there are two entities A and B and the model states that when the level of A exceeds “`1`” (the threshold), the level of B is increased by “`1`”.

3.3.3 FunctionTerms

The resulting value of an entity following a transition may have several possibilities that are governed by a number of conditions. Each transition can have a list of conditional functions *functionTerms*, each associated with a result that allow the user to specify sets of piecewise conditions. For example a model may wish to encode the following

$$B = \begin{cases} B+1 & \text{if } A < 1 \\ B & \text{if } 1 <= A < 3 \\ B+2 & \text{otherwise} \end{cases}$$

In this case the **Transition** would have a **FunctionTerm** for each of the first two conditions and a **DefaultTerm** for the otherwise component.

3.4 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 1 on the following 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 Section 3.5 and Section 3.6 respectively.

3.5 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 **SymbolicValue**. These objects classes are defined in Figure 2 on the next page.

The `id` attribute

The `id` attribute takes a required value of type `SId`. The `id` is used as an identifier for the particluar **QualitativeSpecies**. It can be used as a <ci> element within MathML, in which case it it interpreted as the *level* or *symbol* of this **QualitativeSpecies**.

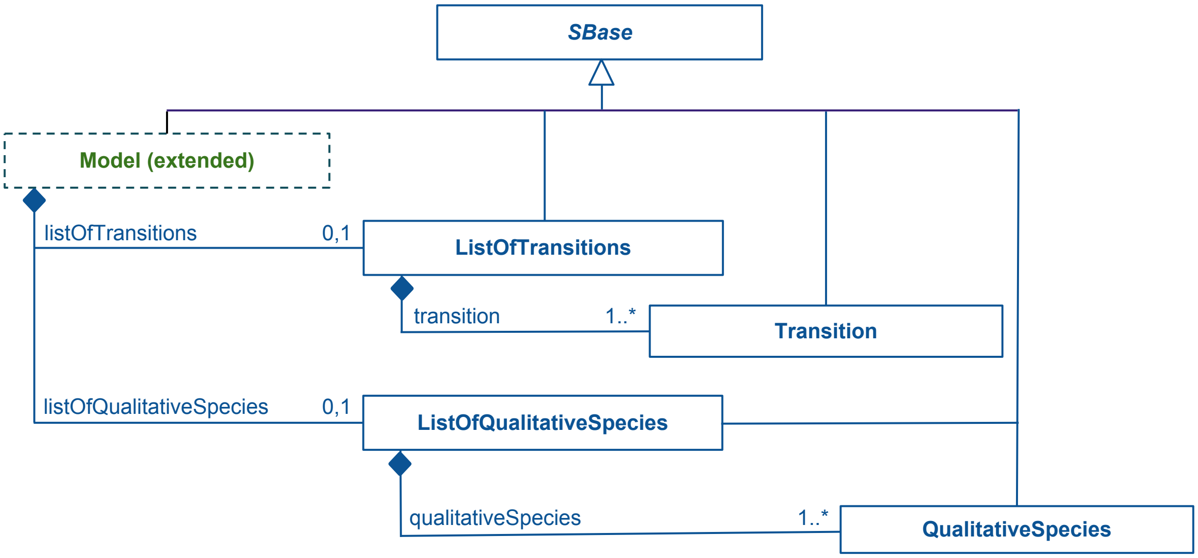

Figure 1: *The definitions of the extended* ***Model*** *class. In other respects,* ***Model*** *remains defined as in the SBML Level 3 Core specification.*

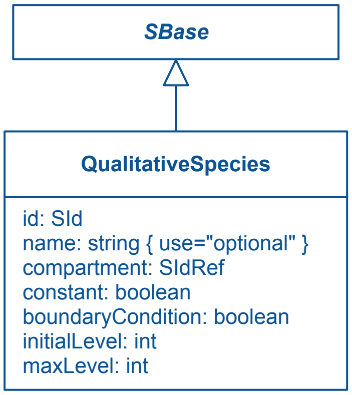

Figure 2: *The definitions of the* ***QualitativeSpecies*** *class.*

The `name` *attribute*

A **QualitativeSpecies** also has an optional `name` attribute of type `string`. The `name` attribute should 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** object in the model. This attribute is comparable with the `compartment` attribute on the **Species** element.

The `constant` *attribute*

The required attribute `boundaryCondition`, of type `boolean`, is used to indicate that the `level` of the qualitativeSpecies is fixed or can be varied. This attribute is comparable with the `constant` attribute on the **Species** element.

Question: If constant=true can the QS be an Input/Output or neither?

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.

3.6 The Transition class

A **Transition** element contains at most one **ListOfInputs** and exactly one **ListOfOutputs** and one **ListOfFunctionTerms**. These objects classes are defined in Figure 3 on the following page.

The `id` *attribute*

A **Transition** element has an optional `id` attribute of type `SId`.

The `name` *attribute*

There is an optional `name` attribute of type `string` that should 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.

3.6.1 The Input class

The **ListOfInputs** contains zero or more elements of type **Input**. A transition with zero inputs can be useful for defining 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 in the corresponding **Transition**.

The `id` *attribute*

An **Input** element has an optional `id` attribute of type `SId`. The identifier of an **Input** can be used as a <ci> element within MathML, in which case it it interpreted as the `thresholdLevel` or `thresholdSymbol`.

The `name` *attribute*

There is an optional `name` attribute of type `string` that should 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 `qualitativeSpecies` *attribute*

The required attribute `qualitativeSpecies`, of type `SIdRef`, is used to identify the **QualitativeSpecies** that is the *input* of this **Transition**. The attribute's value must be the identifier of an existing **QualitativeSpecies** object in the model. This attribute is comparable with the `species` attribute on the **SpeciesReference** element.

The `thresholdLevel` *attribute*

The `thresholdLevel` is a `integer` that can be used to set the threshold level of the particular input.

TODO: Example

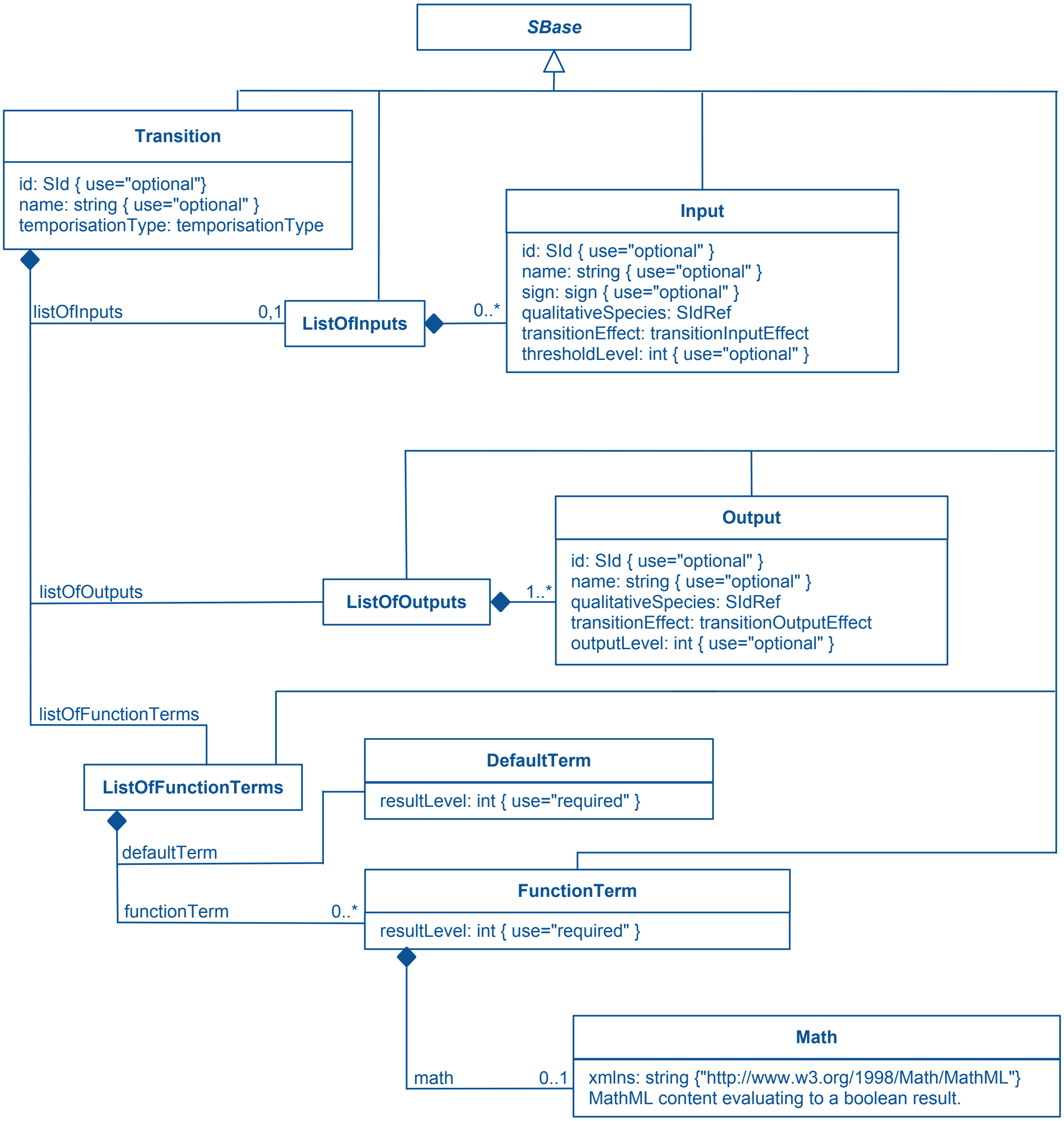
SBase
Transition
id: SId { use="optional"}
name: string { use="optional" }
temporisationType: temporisationType
Input
id: SId { use="optional" }
name: string { use="optional" }
sign: sign { use="optional" }
qualitativeSpecies: SIdRef
transitionEffect: transitionInputEffect
thresholdLevel: int { use="optional" }
listOfInputs
0,1
ListOfInputs
0..*
Output
id: SId { use="optional" }
name: string { use="optional" }
qualitativeSpecies: SIdRef
transitionEffect: transitionOutputEffect
outputLevel: int { use="optional" }
listOfOutputs
ListOfOutputs
1..*
listOfFunctionTerms
ListOfFunctionTerms
DefaultTerm
resultLevel: int { use="required" }
defaultTerm
functionTerm
0..*
FunctionTerm
resultLevel: int { use="required" }
math
0..1
Math
xmlns: string {"http://www.w3.org/1998/Math/MathML"}
MathML content evaluating to a boolean result.

*Figure 3: The definitions of **Transition**, **Input**, **Output**, **DefaultTerm** and **FunctionTerm** classes. Note that the **DefaultTerm** class is not derived from SBase.*

The `transitionEffect` *attribute*

Each **Input** has a required attribute `transitionEffect` of type `transitionInputEffect` which describes how the **QualitativeSpecies** referenced by the **Input** is affected by the **Transition**. Table 1 on the next page shows the possible values with the interpretation of each value.

The following example illustrate the interpretation of the transitionEffect attribute. In the case of qualitativeSpecies "`A`" the `level` is unaltered by the ' **Transition**. However the `level` of qualitativeSpecies "`B`" is reduced by `resultLevel` from the whichever **FunctionTerm** is applicable (see Section|3.6.5

TransitionInputEffect	Interpretation
none	Neither the level nor the symbol associated with 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**.

Table 1: *Interpretation of the* transitionEffect *attribute on an* ***Input****.*

```
<listOfInputs>
    <input qualitativeSpecies="A" thresholdLevel="1" transitionEffect="none"/>
    <input qualitativeSpecies="A" transitionEffect="consumption"/>
</listOfInputs>
```

TODO: An example of a resultLevel modified by a thresholdLevel

The sign *attribute*

The sign of type sign can be used as an indication as to whether the contribution of this input is positive, negative, or both. The sign is usually used for visualization purposes only. This attribute is optional.

3.6.2 The Output class

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

The id *attribute*

An **Output** element has an optional id attribute of type SId.

The name *attribute*

There is an optional name attribute of type string that should 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 qualitativeSpecies *attribute*

The required attribute qualitativeSpecies, of type SIdRef, is used to identify the **QualitativeSpecies** that is the *output* of this **Transition**. The attribute's value must be the identifier of an existing **QualitativeSpecies** object in the model. This attribute is comparable with the species attribute on the **SpeciesReference** element.

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. Question: This contradicts the table

The transitionEffect *attribute*

Each **Output** has a required attribute transitionEffect of type transitionOutputEffect which describes how the **QualitativeSpecies** referenced by the **Output** is affected by the **Transition**. Table 2 on the following page shows the possible values with the interpretation of each value.

3.6.3 The ListOfFunctionTerms class

The **ListOfFunctionTerms** may contain any number of **FunctionTerm** elements, and exactly one **DefaultTerm**. Each **FunctionTerm** encodes the conditions under which this term is selected. The **DefaultTerm** describes the results of the **Transition** applied by default. The disjunction of the terms defines the *qualitative function* associated with a

TransitionInputEffect	Interpretation
production	The level of the qualitativeSpecies is increased by the resultLevel of the selected term possibly modified by the thresholdLevel of the **Output**.
assignmentLevel	The level of the qualitativeSpecies is set to the resultLevel of the selected term.

Table 2: *Interpretation of the* transitionEffect *attribute on an* ***Output****.*

Transition.

3.6.4 The DefaultTerm class

The **DefaultTerm** defines the default result of a **Transition**.

The resultLevel attribute

The default result is described by a resultLevel. This attribute is required.

The resultLevel is an integer describing a level.

Question: why is this not derived from SBase - since this means you cannot add an annotation to it

3.6.5 The FunctionTerm class

Each **FunctionTerm** is also associated with a result (symbolic or level) and in addition to a Boolean function inside a **Math** element that can be used to set the conditions under which this term is selected.

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 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.

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