SBML Level 3 Package Specification

Qualitative Models

Claudine Chaouiya

chaouiya@igc.gulbenkian.pt

IGC Rua da Quinta Grande 6
P-2780-156 Oeiras
Portugal

Sarah M Keating

skeating@ebi.ac.uk

European Bioinformatics Institute
Cambridgeshire
UK

Duncan Berenguier

TAGC INSERM U928
13288 Marseille
France

Aurelien Naldi

Center for Integrative Genomics
CH-1015 Lausanne
Switzerland

Denis Thieffry

IBENS
75005 Paris
France

Martijn P. van Iersel

European Bioinformatics Institute
Cambridgeshire
UK

14th Aug 2012

Version 1.0 (Draft)

The latest release, past releases, and other materials related to this specification are available at

This release of the specification is available at

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 practical 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.

The qualitative models contemplated in this package are essentially 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 encoding qualitative models in SBML.

1.2 Package dependencies

The QualitativeModels package has no dependencies on other SBML Level 3 packages. (If you find incompatibilities with other packages, please contact the Package Working Group. Contact information is shown on the front page of this document.)

1.3 Document conventions

Following the precedent set by the SBML Level 3 Core specification document (Hucka et al., 2010), 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 never instantiated directly, but rather serve as parents of other 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 and context

It is possible to represent some qualitative models using SBML Level 2 or indeed SBML Level 3 Core. However, after several attempts, experience showed that the possible confusion caused by the presence of irrelevant attributes and the need to reinterpret the semantics of some SBML elements could lead to ambiguity. At this point the decision was made to develop an SBML Level 3 package that captured the nature of qualitative models.

A first proposal was written in August 2008 by Duncan Berenguier and Nicolas Le Novère and discussed during a dedicated meeting on the 12th and 13th of August 2008. This meeting brought together a number of people who specialised in qualitative modelling. A summary of the meeting is available at `http://www.ebi.ac.uk/compneur/xwiki/bin/view/SBML/L3F` which also provides a link to the revised proposal document that was produced as a result of this meeting.

A secondary, but very valuable, outcome of this meeting was the formation of the Common Logical Modelling Toolbox (CoLoMoTo) community. A community that focuses on logical modelling but who are committed to making their models exchangeable and reusable as widely as possible. This small focussed community then took control of developing the SBML L3 Qualitaive Models package.

The first CoLoMoTo meeting was held at Oeiras, Portugal 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 and a formal SBML L3 proposal document was written and circulated as a result of these and other discussions. This document is available at `http://sbml.org/images/6/61/SBML-L3-qual-proposal_2.1.pdf`.

The proposal was voted on and accepted by the SBML community (June 2011) and a dedicated discussion list set up (`https://lists.sourceforge.net/lists/listinfo/sbml-qual`). The package was presented at COMBINE 2011.

A second CoLoMoTo meeting took place in March 2012 (see `http://co.mbine.org/colomoto/meetings/2012`). During this meeting it was decided that there were parts of the proposal that had been introduced in anticipation of the future development of models. Whilst these are valuable aspects of the proposal there is no software supporting these features as yet. It was therefore decided to remove these features from a version 1 specification and reconsider them in the future for subsequent versions of the Qualitative Models package. A summary of these features is given in Appendix Section B of this document.

3 Package syntax and semantics

In this section, we define the syntax and semantics of the Qualitative Models 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 14, 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:

<div align="center">

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

</div>

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 (Biron and Malhotra, 2000). We assume and use some of them in the rest of this specification, specifically `boolean`, `ID`, `SId`, `SIdRef`, 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`, `dual` and `unknown`.

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` and `assignmentLevel`.

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

The entities being modelled have a *level* associated with them that indicates 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* that alter the levels of entities involved in the model, depending on the level of some other entities. This may involve the level of an entity being increased or decreased by a fixed amount; the level remaining unchanged; or the level being reassigned to an alternate value. Transitions occur when a set of conditions is met. These conditions may involve the levels 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 affected by 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 next 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 on page 9 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 part in 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 a `level` (an integer representing e.g. an activity state, or a functional level of concentration, etc) These objects classes are defined in Figure 2 on the following page.

The `id` attribute

The `id` attribute takes a required value of type `SId`. The `id` is used as an identifier for the particular **QualitativeSpecies**. It can be used as a <ci> element within MathML, in which case it it interpreted as the *level* of this **QualitativeSpecies**.
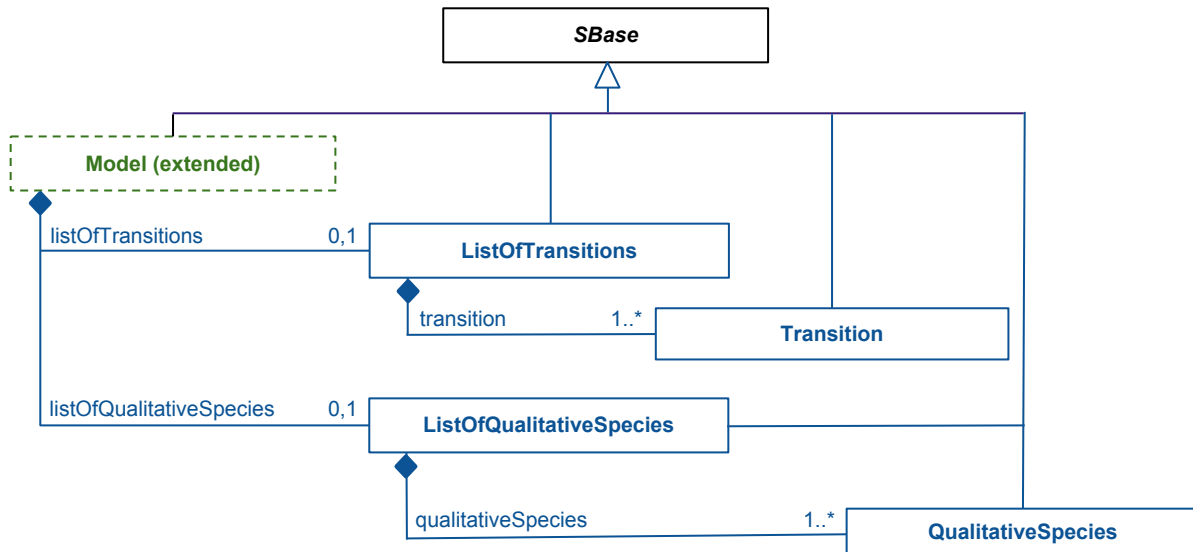
Figure 1: *The definitions of the extended **Model** class. In other respects, **Model** remains defined as in the SBML Level 3 Core specification.*
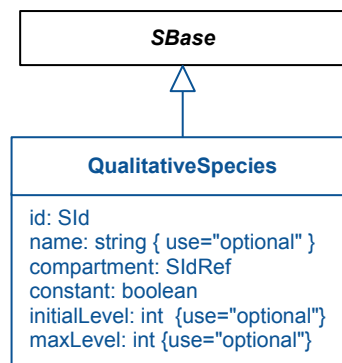
Figure 2: *The definitions of the **QualitativeSpecies** class.*

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*   1

The required attribute `constant`, of type `boolean`, is used to indicate that the `level` of the qualitativeSpecies is   2
fixed or can be varied. This attribute is comparable with the `constant` attribute on the **Species** element.   3

Typically, a qualitativeSpecies that only appears as Input in the model (thus is never modified by a transition,   4
meaning that it receives no interaction) will have this attribute set to true.   5

The `initialLevel` *attribute*   6

The `initialLevel` is an `integer` that defines the initial *level* of the **QualitativeSpecies** in its **Compartment**. This   7
attribute is optional but if set it cannot exceed the value of the `maxlevel` attribute, if this has been set.   8

The `maxLevel` *attribute*   9

The `maxLevel` is an `integer` that sets the maximal *level* of the **QualitativeSpecies**. This attribute is optional.   10

3.6   The Transition class   11

A **Transition** element contains at most one **ListOfInputs** and one **ListOfOutputs** and exactly one **ListOfFunctionTerms**.   12
These objects classes are defined in Figure 3 on the next page.   13

The `id` *attribute*   14

A **Transition** element has an optional `id` attribute of type `SId`. In constrast to most SBML classes the `id` attribute on   15
a **Transition** has no mathematical interpretation.   16

The `name` *attribute*   17

There is an optional `name` attribute of type `string` that should be used in the same manner as on SBML Level 3 Core   18
objects; see Section 3.3.2 of the SBML Level 3 Version 1 Core specification for more information.   19

3.6.1   The Input class   20

The **ListOfInputs** contains zero or more elements of type **Input**. A transition with zero inputs can be useful for   21
defining an initial assignment, where the state of an output depends on a function but not on any input values.   22
An empty list is allowed, and useful for e.g. adding annotations. Each **Input** refers to a **QualitativeSpecies** that   23
participates in the corresponding **Transition**.   24

The `id` *attribute*   25

An **Input** element has an optional `id` attribute of type `SId`. The identifier of an **Input** can be used as a <ci> element   26
within MathML, in which case it it interpreted as the `thresholdLevel`.   27

The `name` *attribute*   28

There is an optional `name` attribute of type `string` that should be used in the same manner as on SBML Level 3 Core   29
objects; see Section 3.3.2 of the SBML Level 3 Version 1 Core specification for more information.   30

The `qualitativeSpecies` *attribute*   31

The required attribute `qualitativeSpecies`, of type `SIdRef`, is used to identify the **QualitativeSpecies** that is the   32
input of this **Transition**. The attribute's value must be the identifier of an existing **QualitativeSpecies** object in the   33
model. This attribute is comparable with the `species` attribute on the **SpeciesReference** element.   34

The `thresholdLevel` *attribute*   35

The `thresholdLevel` is a `integer` that can be used to set the threshold level of the particular input. This attribute   36
relates to the contribution of this input for the transition to take place. In logical regulatory models, it would   37
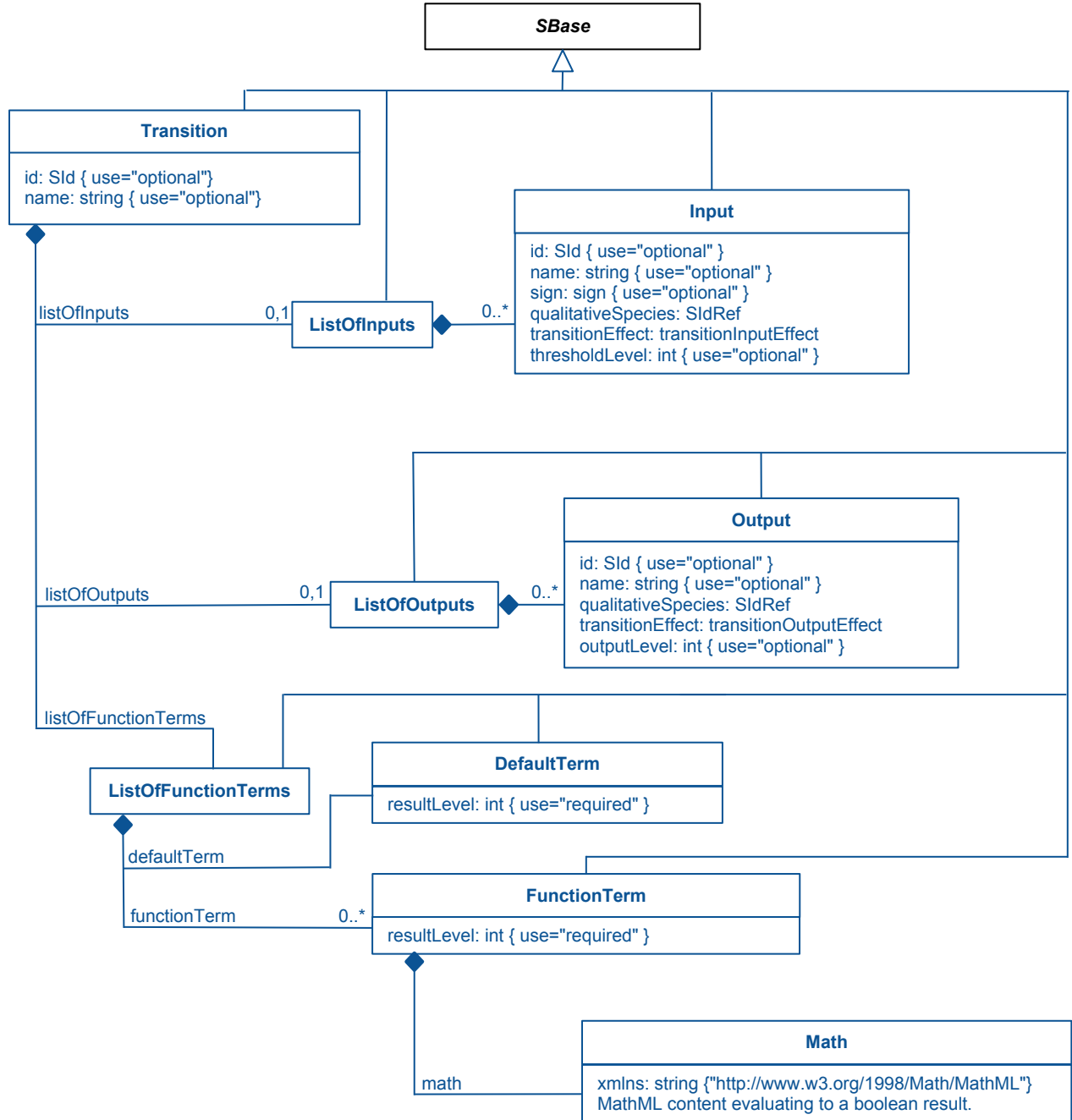
*Figure 3: The definitions of **Transition**, **Input**, **Output**, **DefaultTerm** and **FunctionTerm** classes. Note that the **Default-Term** class is not derived from SBase.*

represent the threshold level above which the regulation takes place, while in a Petri net, it would represent the number of tokens required to enable to transition.

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 following page shows the possible values with the interpretation of each value.

TransitionInputEffect	Interpretation	
none	The level associated with the `qualitativeSpecies` is not modified.	2
consumption	The level of the `qualitativeSpecies` is decreased by the `resultLevel` of the selected term possibly modified by the `thresholdLevel` of the **Input**.	3

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

The following example illustrates the interpretation of the transitionEffect attribute.

```
<listOfInputs>
    <input qualitativeSpecies="A"   transitionEffect="none"              thresholdLevel="2" />
    <input qualitativeSpecies="B"   transitionEffect="consumption"/>
    <input qualitativeSpecies="C"   transitionEffect="consumption" thresholdLevel="2" />
</listOfInputs>
```

In the case of qualitativeSpecies "`A`" the `level` is unaltered by the **Transition** and hence the `transitionEffect` attribute is set to "`none`". The `level` of qualitativeSpecies "`B`" is reduced; hence the `transitionEffect` is "`consumption`". The `level` is reduced by the value of the `resultLevel` from the whichever **FunctionTerm** is applicable (see Section 3.6.5). Similarly, the `level` of "`C`" is also reduced, but on this occasion by 2 (the `threholdLevel`) times the appropriate `resultLevel` . The Petri net example in Section 4 provides further example of the use of the `transitionEffect` and `thresholdLevel` attributes.

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, both (dual) or unknown. The sign is usually used for visualization purposes only. This attribute is optional.

3.6.2   The Output class

The **ListOfOutputs** contains zero or more elements of type **Output**. A transition with zero outputs can be useful for modelling the effect of the environment. For example, in Petri nets, a sink transition (with no output) will consume all tokens arriving in its input places.

Each **Output** refers to a **QualitativeSpecies** that participates in (in affected by) the corresponding **Transition**.

The `id` attribute

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

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` to specify the effect of the **Transition** on the corresponding **QualitativeSpecies**. This attribute is optional.

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 shows the possible values with the interpretation of each value.

The following example illustrates the interpretation of the transitionEffect attribute. In the case of qualitativeSpecies "A" the `level` is assigned the `resultLevel` from the whichever **FunctionTerm** is applicable, whereas the `level` of qualitativeSpecies "B" is increased by `resultLevel`. Similarly, the `level` of "C" is increased by 2 (`outputlevel`) times `resultLevel` (see also Petri net example in Section 4).

```
<listOfOutputs>
    <output qualitativeSpecies="A"   transitionEffect="assignmentLevel"/>
    <output qualitativeSpecies="B"   transitionEffect="production"/>
    <output qualitativeSpecies="C"   transitionEffect="production"  outputLevel="2" />
</listOfInputs>
```

TransitionOutputEffectInterpretation	
production	The level of the `qualitativeSpecies` is increased by the `resultLevel` of the selected term possibly modified by the `outputLevel` 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****.*

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 **Transition**.

3.6.4   The DefaultTerm class

The **DefaultTerm** defines the default result of a **Transition**. This term is used if there are no other **FunctionTerm** elements or if the none of the **Math** elements of the **FunctionTerm** elements evaluates to "`true`".

The* `resultLevel` *attribute

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

The `resultLevel` is an `integer` describing a level. The `resultLevel` is used; possibly together with the `thresholdLevel` or `outputLevel` to determine the level of a **QualitativeSpecies** resulting from the **Transition**.

3.6.5   The FunctionTerm class

Each **FunctionTerm** is also associated with a result 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` *attribute

The result of the term is described by a the required attribute `resultLevel`.

The `resultLevel` is an `integer` describing a level. The `resultLevel` is used; possibly together with the `thresholdLevel` or `outputLevel` to determine the level of a **QualitativeSpecies** resulting from the **Transition**.

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

This proposal mainly covers logical models but it can also handle standard Petri nets. We provide one example of each.

4.1 Simple Logical Regulatory Graph

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. Moreover, A positively regulates B, which positively regulates C, which positively regulates A. In turn A activates itself at level 1 but inhibits itself at a higher level (2) as illustrated by Figure 4.
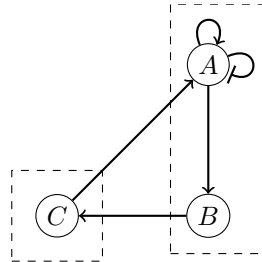
Figure 4: *A simple Logical Regulatory Network.*

The logical functions are the following:

$$\text{B} := 1 \quad \text{if } A => 1$$
$$\text{C} := 1 \quad \text{if } B => 1$$

$$A := \begin{cases} 2 & \text{if } (1 <= A < 2)\, or\, (C >= 1) \\ 1 & \text{if } A < 1\, and\, C >= 1 \\ 0 & \text{otherwise} \end{cases}$$

```xml
<?xml version="1.0" encoding="UTF8"?>
<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">
    <model id="example">
        <listOfCompartments>
            <compartment id="cytosol" name="cytosol" constant="true"/>
            <compartment id="nucleus" name="nucleus" constant="true"/>
        </listOfCompartments>
        <qual:listOfQualitativeSpecies>
            <qual:qualitativeSpecies qual:compartment="cytosol" qual:constant="false"
                            qual:id="A" qual:maxLevel="2"/>
            <qual:qualitativeSpecies qual:compartment="cytosol" qual:constant="false"
                            qual:id="B" qual:maxLevel="1"/>
            <qual:qualitativeSpecies qual:compartment="nucleus" qual:constant="false"
                            qual:id="C" qual:maxLevel="1"/>
        </qual:listOfQualitativeSpecies>
        <qual:listOfTransitions>
            <qual:transition qual:id="tr_B">
                <qual:listOfInputs>
                    <qual:input qual:id="theta_B_A"    qual:qualitativeSpecies="A"
                            qual:thresholdLevel="1"  qual:transitionEffect="none"
                            qual:sign="positive"/>
```

```
            </qual:listOfInputs>                                                    1
            <qual:listOfOutputs>                                                    2
                <qual:output qual:transitionEffect="assignmentLevel"               3
                             qual:qualitativeSpecies="B"/>                          4
            </qual:listOfOutputs>                                                   5
            <qual:listOfFunctionTerms>                                              6
                <qual:functionTerm qual:resultLevel="1">                            7
                    <math xmlns="http://www.w3.org/1998/Math/MathML">              8
                        <!-- A >= 1-->                                              9
                        <apply>                                                    10
                            <geq/>                                                 11
                            <ci>A</ci>                                             12
                            <ci>theta_B_A</ci>                                     13
                        </apply>                                                   14
                    </math>                                                        15
                </qual:functionTerm>                                               16
                <qual:defaultTerm qual:resultLevel="0"/>                           17
            </qual:listOfFunctionTerms>                                            18
        </qual:transition>                                                         19
        <qual:transition qual:id="tr_A">                                           20
            <qual:listOfInputs>                                                    21
                <qual:input qual:id="theta_A_A1"    qual:qualitativeSpecies="A"    22
                            qual:thresholdLevel="1"  qual:transitionEffect="none"  23
                            qual:sign="positive"/>                                 24
                <qual:input qual:id="theta_A_A2"    qual:qualitativeSpecies="A"    25
                            qual:thresholdLevel="2"  qual:transitionEffect="none"  26
                            qual:sign="negative"/>                                 27
                <qual:input qual:id="theta_A_C"     qual:qualitativeSpecies="C"    28
                            qual:thresholdLevel="1"  qual:transitionEffect="none"  29
                            qual:sign="positive"/>                                 30
            </qual:listOfInputs>                                                   31
            <qual:listOfOutputs>                                                   32
                <qual:output qual:qualitativeSpecies="A"                           33
                             qual:transitionEffect="assignmentLevel"/>             34
            </qual:listOfOutputs>                                                  35
            <qual:listOfFunctionTerms>                                             36
                <qual:functionTerm qual:resultLevel="2">                           37
                    <math xmlns="http://www.w3.org/1998/Math/MathML">             38
                        <!-- (A >= 1 and A < 2) or C < 1 -->                       39
                     <apply>                                                       40
                         <or/>                                                     41
                         <apply>                                                   42
                             <and/>                                               43
                             <apply>                                              44
                                 <geq/>                                           45
                                 <ci>A</ci>                                       46
                                 <ci>theta_A_A1</ci>                              47
                             </apply>                                             48
                             <apply>                                              49
                                 <lt/>                                            50
                                 <ci>A</ci>                                       51
                                 <ci>theta_A_A2</ci>                              52
                             </apply>                                             53
                         </apply>                                                 54
                         <apply>                                                  55
                             <lt/>                                                56
```

```
                            <ci>C</ci>                                                    1
                            <ci>theta_A_C</ci>                                            2
                        </apply>                                                          3
                    </apply>                                                              4
                </math>                                                                   5
            </qual:functionTerm>                                                          6
            <qual:functionTerm qual:resultLevel="1">                                      7
                <math xmlns="http://www.w3.org/1998/Math/MathML">                         8
                    <!-- (A < 1) and C >= 1 -->                                           9
                    <apply>                                                              10
                        <and/>                                                           11
                        <apply>                                                          12
                            <lt/>                                                        13
                            <ci>A</ci>                                                   14
                            <ci>theta_A_A1</ci>                                          15
                        </apply>                                                         16
                        <apply>                                                          17
                            <geq/>                                                       18
                            <ci>C</ci>                                                   19
                            <ci>theta_A_C</ci>                                           20
                        </apply>                                                         21
                    </apply>                                                             22
                </math>                                                                  23
            </qual:functionTerm>                                                         24
            <qual:defaultTerm qual:resultLevel="0"/>                                     25
        </qual:listOfFunctionTerms>                                                      26
    </qual:transition>                                                                   27
    <qual:transition qual:id="tr_C">                                                     28
        <qual:listOfInputs>                                                              29
            <qual:input qual:id="theta_C_B"      qual:qualitativeSpecies="B"             30
                        qual:thresholdLevel="1"  qual:transitionEffect="none"            31
                        qual:sign="positive"/>                                           32
        </qual:listOfInputs>                                                             33
        <qual:listOfOutputs>                                                             34
            <qual:output qual:qualitativeSpecies="C"                                     35
                         qual:transitionEffect="assignmentLevel"/>                       36
        </qual:listOfOutputs>                                                            37
        <qual:listOfFunctionTerms>                                                       38
            <qual:functionTerm qual:resultLevel="1">                                     39
                <math xmlns="http://www.w3.org/1998/Math/MathML">                        40
                    <!-- B >= 1-->                                                       41
                    <apply>                                                              42
                        <geq/>                                                           43
                        <ci>B</ci>                                                       44
                        <ci>theta_C_B</ci>                                               45
                    </apply>                                                             46
                </math>                                                                  47
            </qual:functionTerm>                                                         48
            <qual:defaultTerm qual:resultLevel="0"/>                                     49
        </qual:listOfFunctionTerms>                                                      50
    </qual:transition>                                                                   51
    </qual:listOfTransitions>                                                            52
    </model>                                                                             53
</sbml>                                                                                  54
                                                                                        55
```

Listing 1: Logical Regulatory Graph example

4.2   Simple Petri net

The following example shows a simple, standard Petri net, with 4 places A, B, C and D and one transition $t1$ as depicted in Figure 5.
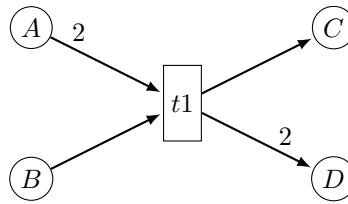

Figure 5: *A Petri net model.*

```
<?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">
    <model id="PN_exemple">
        <listOfCompartments>
            <compartment id="default" constant="true"/>
        </listOfCompartments>
        <qual:listOfQualitativeSpecies>
            <qual:qualitativeSpecies qual:id="A"        qual:compartment="default"
                                     qual:initialLevel="2"  qual:constant="false"/>
            <qual:qualitativeSpecies qual:id="B"        qual:compartment="default"
                                     qual:initialLevel="4"  qual:constant="false"/>
            <qual:qualitativeSpecies qual:id="C"        qual:compartment="default"
                                     qual:initialLevel="2"  qual:constant="false"/>
            <qual:qualitativeSpecies qual:id="D"        qual:compartment="default"
                                     qual:initialLevel="3"  qual:constant="false"/>
        </qual:listOfQualitativeSpecies>
        <qual:listOfTransitions>
            <qual:transition qual:id="t1">
                <qual:listOfInputs>
                    <qual:input qual:id="t1_A"        qual:qualitativeSpecies="A"
                                qual:thresholdLevel="2" qual:transitionEffect="consumption" />
                    <qual:input qual:id="t1_B"        qual:qualitativeSpecies="B"
                                qual:thresholdLevel="1" qual:transitionEffect="consumption" />
                </qual:listOfInputs>
                <qual:listOfOutputs>
                    <qual:output qual:qualitativeSpecies="C" qual:outputLevel="1"
                                qual:transitionEffect="production" />
                    <qual:output qual:qualitativeSpecies="D" qual:outputLevel="2"
                                qual:transitionEffect="production" />
                </qual:listOfOutputs>
                <qual:listOfFunctionTerms>
                    <qual:functionTerm qual:resultLevel="1">
                        <math xmlns="http://www.w3.org/1998/Math/MathML">
                            <!-- A >= 2 and B >= 1 -->
                            <apply>
                                <and />
                                <apply>
                                    <geq />
```

```
                                      <ci>A</ci>                                          1
                                      <ci>t1_A</ci>                                       2
                                  </apply>                                                3
                                  <apply>                                                 4
                                      <geq />                                             5
                                      <ci>B</ci>                                          6
                                      <ci>t1_B</ci>                                       7
                                  </apply>                                                8
                              </apply>                                                    9
                          </math>                                                        10
                      </qual:functionTerm>                                               11
                      <qual:defaultTerm qual:resultLevel="0" />                          12
                  </qual:listOfFunctionTerms>                                            13
              </qual:transition>                                                         14
          </qual:listOfTransitions>                                                      15
      </model>                                                                           16
</sbml>                                                                                  17
                                                                                         18
```

Listing 2: Petri net example

5 Best practices

All [1] 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, meaning that the initialLevel must be defined.

PN To represent unbounded places, the maxLevel should be not specified.

LRG Discussions are still ongoing about the possible (but sometimes convenient to avoid cumbersome descriptions) incoherency of the **FunctionTerm** elements. For now, 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 Any qualitativeSpecies which attribute **constant** is set to "false" should appear as the output of a transition (meaning there is a process governing its evolution). Conversely, any qualitativeSpecies that appears as the output of a transition should have its attribute **constant** set to false.

PN The **transitionEffect** of an **Input** is set to "consumption", unless this input is connected to the transition by a test arc (meaning the transition has no effect on its marking). The **thresholdLevel** indicates the weight of the arc from this place to the transition (when not specified, it is set to 1). It is used to specify the enabling conditions of the transition (and to indicate the number of tokens consumed by the firing of this transition). The **sign** attribute should not be defined.

LRG The **transitionEffect** of an **Input** is set to "none". The **thresholdLevel** indicates the level for which the species participates to the transition (when not defined, its default value is 1). The **sign** attribute indicates the type of effect on the output of the transition (the regulated species): "positive" (activation), "negative" (inhibition), "dual" (positive or negative depending e.g. on co-factors) or "unknown".

PN The **transitionEffect** of an **Output** is set to "production". The **outputLevel** indicates the weight of the arc from the transition to this place (when not defined, its default value is 1). It is used to define the number of tokens produced by the firing of this transition.

LRG The **transitionEffect** of an **Output** is set to "assignmentLevel". The **outputLevel** should not be defined, the level assigned to this species being defined by the **resultLevel** of the transition.

---

[1] **All** all, **PN** Petri nets, **LRG** logical regulatory networks

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 Qualitative Models 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 Qualitative Models 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 Qualitative Models 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 Qualitative Models package.

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

General rules about this package

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

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

General rules about identifiers

qual-10301 ☑ (Extends validation rule #10301 in the SBML Level 3 Version 1 Core specification.) Within a **Model** the values of the attributes `id` and `qual:id` on every instance of the following classes of objects must be unique across the set of all `id` and `qual: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 **QualitativeSpecies**, **Transition**, **Input** and **Output** objects defined by the Qualitative Models package. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, **??** on page ??.)

Rules for the extended SBML class

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

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

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

Rules for extended Model object

qual-20201 ☑ There may be at most one instance of each of the following kinds of objects within a **Model** object using Qualitative Models: **ListOfTransitions** and **ListOfQualitativeSpecies**. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.4 on page 7.)

qual-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: **ListOfQualitativeSpecies** and **ListOfTransitions**. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.4 on page 7.)

qual-20203 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfTransitions** container object may only contain **Transition** objects. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.4 on page 7.)

qual-20204 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfQualitativeSpecies** container object may only contain **QualitativeSpecies** objects. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.4 on page 7.)

qual-20205 ☑ A **ListOfQualitativeSpecies** 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 Qualitative Models namespace are permitted on a **ListOfQualitativeSpecies** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.4 on page 7.)

qual-20206 ☑ A **ListOfTransitions** object may have the optional attributes `metaid` and `sboTerm` defined by SBML Level 3 Core. No other attributes from the SBML Level 3 Core namespace or the Qualitative Models namespace are permitted on a **ListOfTransitions** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.4 on page 7.)

Rules for QualitativeSpecies object

qual-20301 ☑ A **QualitativeSpecies** 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 **QualitativeSpecies**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20302 ☑ A **QualitativeSpecies** 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 **QualitativeSpecies**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20303 ☑ A **QualitativeSpecies** object must have the required attributes `qual:id`, `qual:compartment` and `qual:constant`, and may have the optional attributes `qual:name`, `qual:initialLevel` and `qual:maxLevel`. No other attributes from the SBML Level 3 Qualitative Models namespace are permitted on a **QualitativeSpecies** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

qual-20304 ☑ The attribute `qual:constant` in **QualitativeSpecies** must be of the data type `boolean`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

qual-20305 ☑ The attribute `qual:name` in **QualitativeSpecies** must be of the data type `string`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

qual-20306 ☑ The attribute `qual:initialLevel` in **QualitativeSpecies** must be of the data type `integer`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

qual-20307 ☑ The attribute `qual:maxLevel` in **QualitativeSpecies** must be of the data type `integer`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

qual-20308 ☑ The value of the attribute `qual:compartment` in a **QualitativeSpecies** object must be the identifier of an existing **Compartment** object defined in the enclosing **Model** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

qual-20309 ☑ The value of the attribute `qual:initialLevel` in a **QualitativeSpecies** object cannot be greater than the value of the `qual:maxLevel` attribute for the given **QualitativeSpecies** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

qual-20310 ☑ A **QualitativeSpecies** with attribute `qual:constant` set to `true` cannot only be refferred to by an **Input**. It cannot be the subject of an **Output** in a **Transition**. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.5 on page 7.)

Rules for Transition object

qual-20401 ☑ A **Transition** 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 **Transition**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20402 ☑ A **Transition** 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 **Transition**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20403 ☑ A **Transition** object may have the optional attributes `qual:id` and `qual:name`. No other attributes from the SBML Level 3 Qualitative Models namespace are permitted on a **Transition** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20404 ☑ The attribute `qual:name` in **Transition** must be of the data type `string`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20405 ☑ A **Transition** must have one and only one instance of the **ListOfFunctionTerms** objects and may have at most one instance of the **ListOfInputs** and **ListOfOutputs** objects from the Qualitative Models namespace. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20406 ☑ The various **ListOfFunctionTerms** subobject with a **Transition** object must not be empty. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20407 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfInputs** container object may only contain **Input** objects. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20408 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfOutputs** container object may only contain **Output** objects. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20409 ☑ Apart from the general notes and annotation subobjects permitted on all SBML objects, a **ListOfFunctionTerms** container object must contain one and only one **DefaultTerm** object and then may only contain **FunctionTerm** objects. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20410 ☑ A **ListOfInputs** 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 Qualitative Models namespace are permitted on a **ListOfInputs** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20411 ☑ A **ListOfOutputs** object may have the optional attributes `metaid` and `sboTerm` defined by SBML Level 3 Core. No other attributes from the SBML Level 3 Core namespace or the Qualitative Models namespace are permitted on a **ListOfOutputs** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

qual-20412 ☑ A **ListOfFunctionTerms** object may have the optional attributes `metaid` and `sboTerm` defined by SBML Level 3 Core. No other attributes from the SBML Level 3 Core namespace or the Qualitative Models namespace are permitted on a **ListOfFunctionTerms** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6 on page 9.)

Rules for Input object

qual-20501 ☑ A **Input** 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 **Input**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20502 ☑ A **Input** 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 **Input**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20503 ☑ A **Input** object must have the required attributes `qual:qualitativeSpecies` and `qual:transitionEffect`, and may have the optional attributes `qual:id`, `qual:name`, `qual:sign` and `qual:thresholdLevel`. No other attributes from the SBML Level 3 Qualitative Models namespace are permitted on a **Input** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.1 on page 9.)

qual-20504 ☑ The attribute `qual:name` in **Input** must be of the data type `string`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.1 on page 9.)

qual-20505 ☑ The value of the attribute `qual:sign` of a **Input** object must conform to the syntax of the SBML data type `sign` and may only take on the allowed values of `sign` defined in SBML; that is, the value must be one of the following: "`positive`", "`negative`", "`dual`" or "`unknown`". (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.1 on page 9.)

qual-20506 ☑ The value of the attribute `qual:transitionEffect` of a **Input** object must conform to the
syntax of the SBML data type `transitionInputEffect` and may only take on the allowed
values of `transitionInputEffect` defined in SBML; that is, the value must be one of the
following: "`none`" or "`consumption`". (References: SBML Level 3 Package Specification for
Qualitative Models, Version 1, Section 3.6.1 on page 9.)

qual-20507 ☑ The attribute `qual:thresholdLevel` in **Input** must be of the data type `integer`. (References:
SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.1 on page 9.)

qual-20508 ☑ The value of the attribute `qual:qualitativeSpecies` in an **Input** object must be the identifier
of an existing **QualitativeSpecies** object defined in the enclosing **Model** object. (References:
SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.1 on page 9.)

Rules for Output object

qual-20601 ☑ A **Output** 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 **Output**.
(References: SBML Level 3 Version 1 Core, Section 3.2.)

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

qual-20603 ☑ A **Output** object must have the required attributes `qual:qualitativeSpecies` and `qual:-`
`transitionEffect`, and may have the optional attributes `qual:id`, `qual:name` and `qual:-`
`outputLevel`. No other attributes from the SBML Level 3 Qualitative Models namespace are
permitted on a **Output** object. (References: SBML Level 3 Package Specification for Qualitative
Models, Version 1, Section 3.6.2 on page 11.)

qual-20604 ☑ The attribute `qual:name` in **Output** must be of the data type `string`. (References: SBML Level 3
Package Specification for Qualitative Models, Version 1, Section 3.6.2 on page 11.)

qual-20605 ☑ The value of the attribute `qual:transitionEffect` of a **Output** object must conform to the
syntax of the SBML data type `transitionOutputEffect` and may only take on the allowed
values of `transitionOutputEffect` defined in SBML; that is, the value must be one of the
following: "`production`" or "`assignmentLevel`". (References: SBML Level 3 Package Specifi-
cation for Qualitative Models, Version 1, Section 3.6.2 on page 11.)

qual-20606 ☑ The attribute `qual:outputLevel` in **Output** must be of the data type `integer`. (References:
SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.2 on page 11.)

qual-20607 ☑ The value of the attribute `qual:qualitativeSpecies` in an **Output** object must be the identi-
fier of an existing **QualitativeSpecies** object defined in the enclosing **Model** object. (References:
SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.2 on page 11.)

qual-20608 ☑ The **QualitativeSpecies** referred to by the attribute `qual:qualitativeSpecies` in an **Output**
object must have the value of its `qual:constant` attribute set to `false`. (References: SBML
Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.2 on page 11.)

Rules for DefaultTerm object

qual-20701 ☑ A **DefaultTerm** 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 **DefaultTerm**.
(References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20702 ☑ A **DefaultTerm** 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
DefaultTerm. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20703 ☑ A **DefaultTerm** object must have the required attributes `qual:resultLevel`. No other attributes from the SBML Level 3 Qualitative Models namespace are permitted on a **DefaultTerm** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.4 on page 12.)

qual-20704 ☑ The attribute `qual:resultLevel` in **DefaultTerm** must be of the data type `integer`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.4 on page 12.)

Rules for FunctionTerm object

qual-20801 ☑ A **FunctionTerm** 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 **FunctionTerm**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20802 ☑ A **FunctionTerm** 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 **FunctionTerm**. (References: SBML Level 3 Version 1 Core, Section 3.2.)

qual-20803 ☑ A **FunctionTerm** object must have the required attributes `qual:resultLevel`. No other attributes from the SBML Level 3 Qualitative Models namespace are permitted on a **FunctionTerm** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.5 on page 12.)

qual-20804 ☑ A **FunctionTerm** object may contain exactly one MathML `qual:math` element. No other elements from the SBML Level 3 Qualitative Models namespace are permitted on a **FunctionTerm** object. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.5 on page 12.)

qual-20805 ☑ The attribute `qual:resultLevel` in **FunctionTerm** must be of the data type `integer`. (References: SBML Level 3 Package Specification for Qualitative Models, Version 1, Section 3.6.5 on page 12.)

B Future directions

B.1 Symbols

B.2 Temporisation

Acknowledgments

References

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/`.

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/`.

Hucka, M., Bergmann, F. T., Hoops, S., Keating, S. M., Sahle, S., Schaff, J. C., Smith, L. P., and Wilkinson, D. J. (2010). The Systems Biology Markup Language (SBML): Language Specification for Level 3 Version 1 Core. Available via the World Wide Web at `http://sbml.org/Documents/Specifications`.

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

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/`.