

Proposal title

Qualitative Models (qual)

Proposal authors^a

Duncan Berenguier
TAGC INSERM U928 and IML CNRS UMR 6206,
Luminy, 163 av. de Luminy 13288 Marseille, France

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

Aurelien Naldi
Center for Integrative Genomics , University of Lausanne
Genopode Building, CH-1015 Lausanne, Switzerland

Denis Thieffry
IBENS, 46 rue d'ULM, 75005 Paris, France

Proposal tracking number

TODO: ask for a tracking number

Version information

Version number and date of public release

Version 2.1 released on May, 12, 2011

URL for this version of the proposal

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

URL for the previous versions of this proposal

http://sbml.org/Events/Other_Events/Alternative_modelling_Frameworks_2008/L3F_Proposal. TODO: add previous proposal

^aHave also collaborated to this proposal: T. Helikar, A. von Kamp, S. Klamt, N. Le Novère

Introduction and motivation

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.

Past work on this problem or similar topics

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 2008¹. 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.

Graphical and typographical conventions

Throughout this document, the following typographical conventions are used :

- **Classes** names begin with a capital letter a capital letter and are written in bold, sans-serif typeface. In addition, if a class describes elements introduced in this extension, these are written in light blue and are hyperlinked to the section where they are defined, *e.g.* [QualitativeSpecies](#).
- **attributes** are written in sans-serif typeface.
- **constants** and **types** are written in typewriter typeface.

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.

¹Nicolas 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 ([Chem-chains](#)), Ioannis Xenarios ([SQUAD](#)), Alessandro Di Cara (SQUAD), Mathias John ([PiML](#)), Dagmar Koehn (PiML)

XML Namespace

The XML-namespace used for elements defined by sbml-qual follows the convention of all SBML packages. This means that the namespace is "http://www.sbml.org/sbml/level3/version1/qual/version1" (in the case of SBML Level 3 version 1, SBML-qual version 1).

Proposed syntax and semantics

Like **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, and the **Transition** element to represent their qualitative influences.

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

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

The overall structure of this extension is described in Figure 1.

The following sections define the new elements and their attributes. The section **Interpreting transitions** further indicates how these elements should be combined.

Extension of the **Model** element

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

Extension of the **AssignmentRule** element

The definition of the attribute **variable** of the SBML element **AssignmentRule** is extended to possibly refer to a **QualitativeSpecies** (in addition to **Species**, **SpeciesReference**, **Compartment**, or **Parameter**).

Extension of the **EventAssignment** element

The definition of the attribute **variable** of the SBML element **EventAssignment** is extended to possibly refer to a **QualitativeSpecies** (in addition to **Species**, **SpeciesReference**, **Compartment**, or **Parameter**).

Definition of **QualitativeSpecies**

The **Model** element may contain (at most) one **ListOfQualitativeSpecies** that contains at least one **QualitativeSpecies**.

A **QualitativeSpecies** describes a pool of indistinguishable entities in a **Compartment**. It is associated with either a *level* or a *symbol* from its **ListOfSymbolicValues**.

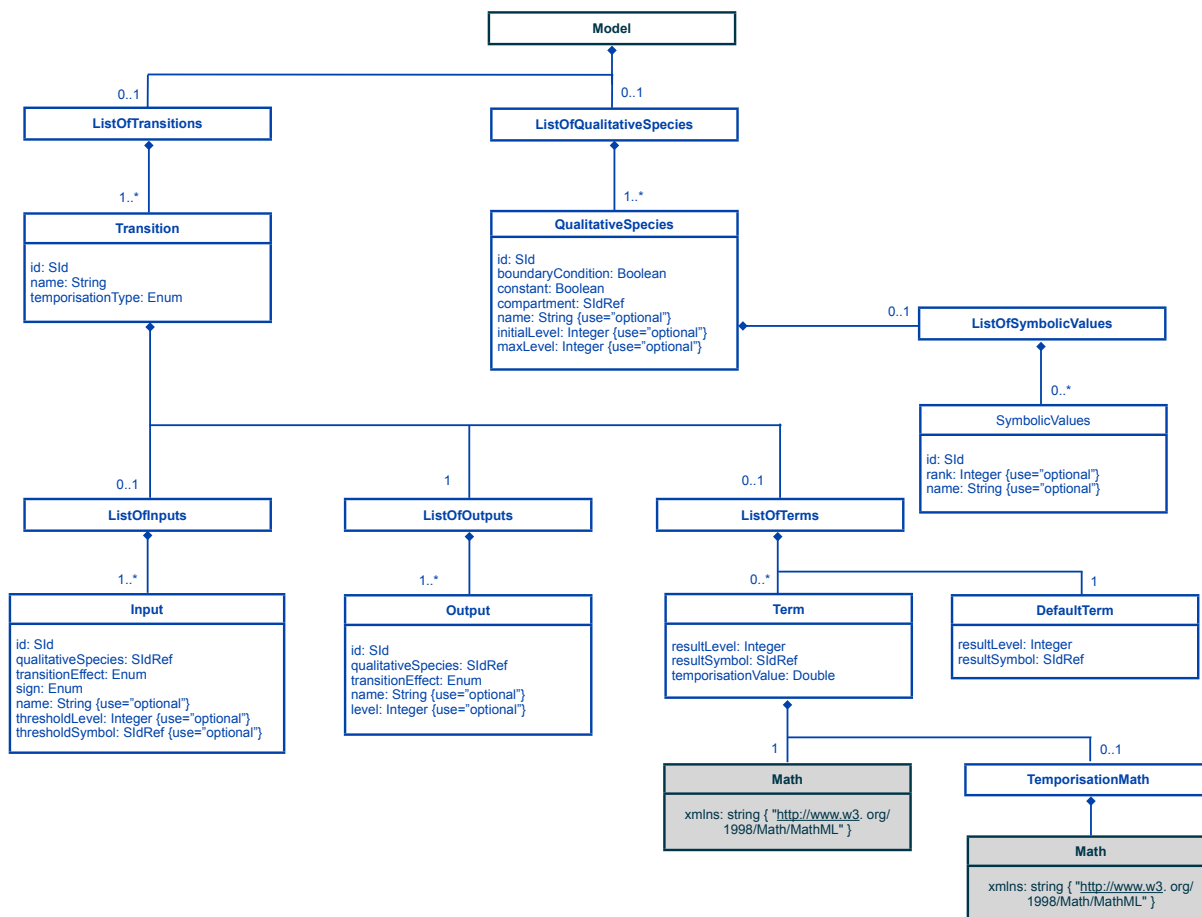


Figure 1: UML diagram of the proposal of Qualitative Models package for SBML Level 3

The id and name attributes: These attributes are used according to the SBML L3v1 Section 3.3. The attribute id is mandatory and name is optional.

The compartment, constant and boundaryCondition attributes: These attributes are treated as in **Species** elements.

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

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

Definition of SymbolicValue

The **QualitativeSpecies** element may contain at most one **ListOfSymbolicValues** that contains at least one **SymbolicValue**. The **SymbolicValue** element defines a non instantiated parameter. Such

symbols may represent the different solutions of piecewise linear differential equations, along with different thresholds.

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

The rank attribute The rank is an `integer` that defines the position of the symbol in the [ListOfSymbolicValues](#). This attribute is optional.

Definition of Transition

The **Model** element may contain at most one [ListOfTransitions](#) that contains at least one [Transition](#). A [Transition](#) element contains at most one [ListOfInputs](#), exactly one [ListOfOutputs](#) and one [ListOfFunctionTerms](#).

The id and name attributes: These attributes are used according to the SBML L3.1 Section 3.3. They are both optional.

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

Definition of Input

The [ListOfInputs](#) contains at least one [Input](#). Each [Input](#) refers to a [QualitativeSpecies](#) that participates to the corresponding [Transition](#).

The id and name attributes: These attributes are used according to the SBML L3.1 Section 3.3. They are both optional.

The qualitativeSpecies attribute: The `qualitativeSpecies` is a `SIIdRef` referring to a [QualitativeSpecies](#). This attribute is mandatory.

The thresholdLevel and thresholdSymbol attributes: The `thresholdLevel` is an `integer` and `thresholdSymbol` is a `SIIdRef`. They are optional and exclusive.

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

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

Definition of Output

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

The id and name attributes: These attributes are used according to the SBML L3.1 Section 3.3. They are both optional.

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

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

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

Definitions of FunctionTerm and DefaultTerm

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

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

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

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

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

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

Interpreting transitions

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

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

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

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

1 Use-cases and examples

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 : $B := 1 \text{ if } A \geq 1$, $C := 1 \text{ if } B \geq 1$, $A := 2 \text{ if } (A \geq 1 \text{ and } A < 2) \text{ or } C \geq 1$; $A := 1 \text{ if } A < 1 \text{ and } C \geq 1$; $A := 0 \text{ otherwise}$.

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/level3/version1/qual/version1"
  qual:required="true">
  <model id="example">
    <listOfCompartments>
      <compartment id="cytosol" name="cytosol"/>
    </listOfCompartments>
  </model>
</sbml>
```

```

    <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 --></math>
                <apply>
                    <geq/>
                    <ci>A</ci>
                    <ci>theta_B_A</ci>
                </apply>
            </functionTerm>
            <defaultTerm resultLevel="0" />
        </listOfFunctionTerms>
    </transition>
    <transition id="tr_A">
        <listOfInputs>
            <input id="theta_A_A1" qualitativeSpecies="A" thresholdLevel="1"
                transitionEffect="none" sboTerm="SBO:0000170" />
            <input id="theta_A_A2" qualitativeSpecies="A" thresholdLevel="2"
                transitionEffect="none" sboTerm="SBO:0000170" />
            <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 --></math>
                <apply>
                    <or/>
                    <apply>
                        <and/>
                        <apply>
                            <geq/>
                            <ci>A</ci>
                            <ci>theta_A_A1</ci>
                        </apply>
                    </apply>
                </apply>
            </functionTerm>
        </listOfFunctionTerms>
    </transition>
</qual:listOfTransitions>

```



```

        </apply>
        <apply>
            <lt />
            <ci>A</ci>
            <ci>theta_A_A2</ci>
        </apply>
    </apply>
    <apply>
        <lt />
        <ci>C</ci>
        <ci>theta_A_C</ci>
    </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>

```

```

    </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/level3/version1/qual/version1"
  qual:required="true">
  <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 --></math>
            <apply>
              <and />
              <apply>
                <geq />
                <ci>A</ci>
                <ci>t1_A</ci>
              </apply>
            </apply>
          </functionTerm>
        </listOfFunctionTerms>
      </transition>
    </qual:listOfTransitions>
  </model>
</sbml>

```

```

        <apply>
          <geq />
          <ci>A</ci>
          <ci>t1.B</ci>
        </apply>
      </apply>
    </math>
  </functionTerm>
  <defaultTerm result="0" />
</listOfFunctionTerms>
</transition>
</qual:listOfTransitions>
</model>
</sbml>

```

2 Package dependencies

This package does not depend on any other SBML Level 3 package.

3 Prototype implementations

Prototype for GINsim.

4 Unresolved issues

5 Recommended 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 .

References

Appendix