NOT OFFICIAL - Legacy work on future CellML Specification Documentation

Release

Authors and Contributors

July 29, 2014
THIS IS PROBABLY NOT WHAT YOU ARE LOOKING FOR. It is an RST generated version of some legacy work on the future CellML 1.2 spec.

This document is an unofficial working draft. The below describes the intended status of the specification, and not the current status right now.

This document specifies CellML 1.2, an XML-based language for describing and exchanging mathematical models.

This is the *normative* specification of CellML. It is intended to provide the minimum amount of information needed to accurately describe CellML. An alternative version is available that is annotated with much more explanatory material.
The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in ?.

The key phrase “information item”, as well as any specific type of information item such as an “element information item”, are to be interpreted as described in ?.

**CellML Infoset** An XML Information Set containing a hierarchy of information items conforming to the rules described in this document.

**CellML Model** A mathematical model represented by a hierarchy of one or more CellML Infosets, according to the rules described in this document.

**CellML Processing Software** Software that processes CellML in accordance with the rules of this document.

**Namespace** An XML namespace, as defined in ?.

**CellML Namespace** Any namespace starting with http://www.cellml.org/cellml/

**CellML 1.2 Namespace** The namespace http://www.cellml.org/cellml/1.2#

**MathML Namespace** The namespace http://www.w3.org/1998/Math/MathML

**RDF Namespace** The namespace http://www.w3.org/1999/02/22-rdf-syntax-ns#

**CellML Metadata Namespace** The namespace http://www.cellml.org/metadata/1.0#


**Basic Latin alphabetic character** A Unicode character in the range U+0041 to U+005A or in the range U+0061 to U+007A.

**European numeric character** A Unicode character in the range U+0030 to U+0039.

**Basic Latin alphanumeric character** A Unicode character that is either a basic Latin alphabetic character or a European numeric character.

**Basic Latin underscore** The Unicode character U+005F.

**Whitespace character** Any one of the Unicode characters U+0020, U+0009, U+000D or U+000A.

**RDF Triple** As defined in ?.

**Independent variable set** A set of zero variables in the model (each represented by a set of one or more connected CellML variables) that are defined externally to the model.
2.1 CellML and XML

1. Every CellML Infoset SHALL be represented in an XML document that conforms with the well-formedness requirements of the XML 1.0 specification.

2. In this document, the remaining provisions relating to CellML Infosets shall be interpreted as being constraints on the XML Information Set represented by that CellML Infoset.

2.2 Equivalent CellML Infosets

1. Two CellML Infosets shall be equivalent if one can be transformed to another by making zero or more of the following changes:
   (a) Changing the representation the XML file in ways that do not change the XML Information Set represented
   (b) Adding, removing, and/or modifying comment information items.
   (c) Changing (inserting, removing, and/or modifying) one or more namespace information items, and/or modifying the prefix of one or more information items, without changing the namespace that any information item is in.
   (d) The following paragraph applies only to character information items for which all ancestor element information items are in the CellML, the MathML or the RDF namespace.
      Inserting or removing character information items that consist entirely of whitespace characters, changing the number of whitespace characters in such an information item, or changing the number of whitespace characters at the beginning or end of a character information item.

2. CellML Processing Software MUST treat CellML Infosets that are equivalent according to the above rules in an identical fashion.

2.3 Character information items

An element information item in the CellML namespace MUST NOT contain any character information items, except for character information items that consist entirely of whitespace characters.
2.4 Use of namespaces

1. CellML Infosets MUST NOT contain any element or attribute information items, except as permitted in this specification.

2. CellML Infosets MUST NOT contain any character information items that are children of element information items in a CellML namespace, except as permitted in this specification.

3. CellML Processing Software SHOULD NOT attempt to process CellML Infosets that contain information items in a CellML namespace other than the CellML 1.2 namespaces, unless they have been designed to support the version of CellML corresponding to the namespace.

4. CellML Infosets MUST NOT contain any element information items in the RDF namespace, unless:
   (a) the element information item or one of its ancestors is an element information item in the RDF namespace, with local name RDF (the RDF element information item), and,
   (b) the RDF element information item forms the top-level of a valid RDF/XML tree, per production 7.2.9 in .

5. CellML Infosets MUST NOT contain any element information items in the MathML namespace, unless:
   (a) the element information item or one of its ancestors is an element information item in the MathML namespace, with local name math (the math element information item), and,
   (b) the math element information item forms the top-level of a valid MathML tree, as described in .

2.5 Extension information items

1. CellML Infosets MAY contain zero or more element or attribute information items in an extension namespace (extension information items), as children of CellML Information Items.

2. Information items in the empty namespace, that appear as children of extension element information items, SHALL also be treated as extension information items.

3. Extension information items MAY contain extension information items as their children. In addition, extension elements MAY contain presentation MathML children. For the avoidance of doubt, it is noted that extension information items MAY also contain any other information items not disallowed by this specification or a referenced normative specification.

4. CellML Processing Software MUST NOT raise an error solely because it encounters an unrecognised extension element.

5. CellML Processing Software that reads CellML and then writes a modified version back out SHOULD preserve unrecognised extension information items when the parent information item is not modified.

6. CellML Processing Software MUST NOT allow the mathematical interpretation of a CellML Model to be altered by any information present in extension information items.

2.6 Identifiers

1. Any element information item in the CellML namespace MAY contain an attribute information item with name id, in the CellML Metadata Namespace. This attribute information item SHALL be treated as having attribute type ID, as defined in section 3.3.1 of ?.
2.7 Specific information items

1. A specific information item is an information item that is not:
   (a) a comment information item, or,
   (b) a character information item, or,
   (c) an extension information item or a descendant of such an information item, or,
   (d) a namespace information item.

2. Specific information items MUST NOT appear in a CellML Infoset except where explicitly allowed by this specification, or where allowed by a normative specification referenced by this specification.

3. The order in which specific information items appear as children of an element information item defined in this specification SHALL NOT affect the interpretation of the CellML Model.

2.8 RDF Element Information Items

1. Every element information item in the CellML namespace MAY contain zero or more RDF Element Information Item children.

2. An RDF Element Information Item SHALL be an element item in the RDF namespace, with local name RDF (the RDF element information item), and MUST form the top-level of a valid RDF/XML tree, per production 7.2.9 in ?, and SHALL be a child of an element information item in the CellML namespace.

3. CellML Processing Software MUST NOT allow the mathematical interpretation of a CellML Model to be altered by any information present in RDF data.

4. The “set of all RDF triples” associated with a CellML Infoset SHALL refer to the union of all sets of RDF triples produced by parsing all the RDF Element Information Items according to the RDF/XML specification.

5. Two CellML Infosets that differ only by the addition, removal, or modification of RDF Element Information Items (or Information Items descended from them), but that have the same set of all RDF triples, SHALL be termed RDF-equivalent CellML Infosets.

The following data representation formats are defined for use in this specification:

1. A CellML identifier:
   (a) SHALL be a sequence of Unicode characters.
   (b) SHALL NOT contain any characters except basic Latin alphanumeric characters and basic Latin underscores.
   (c) SHALL contain one or more basic Latin alphabetic characters.
   (d) SHALL NOT begin with a European numeric character.
   (e) SHALL, when comparing two identifiers, be considered identical to another identifier if and only if both identifiers have identical sequences of Unicode character codes.

2. A non-negative integer string:
   (a) SHALL be a base 10 representation of a non-negative integer.
   (b) SHALL consist entirely of European numeric characters.

3. An integer string:
   (a) SHALL be a base 10 representation of an integer.
      When the integer being represented is negative, SHALL consist of the Basic Latin hyphen-minus character U+002D, followed by the non-negative integer string representation of the absolute value of the integer.
      When the integer being represented is non-negative, SHALL consist of the non-negative integer string representation of the integer.

4. A basic real number string:
   (a) SHALL be a base 10 representation of a real number.
   (b) Where the number is negative, SHALL begin with the Basic Latin hyphen-minus character U+002D as the sign indicator.
   (c) MAY contain a single decimal point separator, which SHALL be the basic Latin full stop character U+002E.
   (d) Other than the sign indicator and the decimal point separator, SHALL consist only of European numeric characters.

5. A real number string:
   (a) SHALL be a base 10 representation of a real number \( r = s \times 10^e \), where \( s \) is the significand, a real number, and \( e \) is the exponent, an integer.
(b) The representation of the number SHALL be the representation of the significand followed immediately by the representation of the exponent.

(c) The significand SHALL be represented as a basic real number string.

(d) If the exponent is zero, the exponent MAY be represented by an empty string, or MAY be represented according to the following rule. If the exponent is non-zero, it MUST be represented according to the following rule.

(e) An exponent SHALL be represented by an exponent separator character, followed by the integer string representation of the exponent’s value. The exponent separator character SHALL be either the Unicode Basic Latin character ‘E’ or the Unicode Basic Latin character ‘e’.
CHAPTER 4

The model element information item

4.1 Top-level of CellML Infosets

The top-level element information item in a CellML Infoset MUST be an element information item in the CellML 1.2 namespace, with local name model. This element information item is referred to in this specification as the model element.

4.2 Specific information items

1. Every model element MUST contain a name attribute in the empty namespace. The value of the name attribute MUST be a valid CellML identifier, and SHALL be interpreted as a unique identifier for the CellML Infoset.

2. A model element MAY contain zero or more additional specific information item children, each of which MUST be of one of the following types:
   (a) a component element (?),
   (b) a connection element (?),
   (c) an encapsulation element (?),
   (d) an import element (?),
   (e) a units element (?)
The import element information item

Import element information items (referred to in this specification as import elements) are element information items in the CellML 1.2 namespace with local name equal to import.

5.1 Specific information items

1. Every import element MUST contain the following attribute information items:
   (a) An attribute information item in the namespace http://www.w3.org/1999/xlink, and with local name href. The value of this attribute SHALL be a valid locator href, as defined in section 5.4 of the XLink specification (?). The href attribute SHALL be treated according to the XLink specification, by applying the rules for simple-type elements. When describing an import element or one of its children, the phrase “imported CellML Infoset” SHALL refer to the CellML Infoset obtained by parsing the XML document referenced by the href attribute.
   (b) An attribute information with local name name, in the CellML 1.2 namespace. The value of the name attribute MUST be a valid CellML Identifier. The value of the name attribute MUST NOT be identical to the name attribute on any sibling import element.

2. Imported CellML Infosets make up part of the model, so for the entire model to be valid, all imported CellML Infosets MUST also be valid under this specification.

3. Every import element MAY contain zero or more specific information children, each of which MUST be of one of the following types:
   (a) an import component element (?).
   (b) an import units element (?).
The import units element information item

Import units element information items (referred to in this specification as import units elements) are element information items in the CellML 1.2 namespace with local name equal to units which appear as children of import elements.

6.1 Specific information items

1. Every import units element MUST contain a name attribute in the empty namespace. The value of the name attribute MUST be a valid CellML identifier. The value of the name attribute MUST NOT be identical to the name attribute of any other import units or units element in the CellML Infoset.

2. Every import units element MUST contain a units_ref attribute in the empty namespace. The value of the units_ref attribute MUST be a valid CellML identifier. The value of the units_ref attribute MUST match the value of the name attribute on a units or import units element in the imported CellML Infoset. The value of the units_ref attribute MUST NOT match the value of the units_ref attribute on any sibling import units element.
The import component element information item

Import component element information items (referred to in this specification as import component elements) are element information items in the CellML 1.2 namespace with local name equal to component which appear as children of import elements.

7.1 Specific information items

1. Every import component element MUST contain a name attribute in the empty namespace. The value of the name attribute MUST be a valid CellML identifier. The value of the name attribute MUST NOT be identical to the name attribute of any other import component or component element in the CellML Infoset.

2. Every import component element MUST contain a component_ref attribute in the empty namespace. The value of the component_ref attribute MUST be a valid CellML identifier. The value of the component_ref attribute MUST match the value of the name attribute on a component or import component element in the imported CellML Infoset. The value of the component_ref attribute MUST NOT match the value of the component_ref attribute on any sibling import component element.
The units element information item

Units element information items (referred to in this specification as units elements) are element information items in the CellML 1.2 namespace with local name equal to units, and with either a component or model element as their parent.

8.1 Specific information items

1. Every units element MUST contain a name attribute in the empty namespace. The value of the name attribute MUST be a valid CellML identifier. Where the parent of the units element is a model element, the value of the name attribute MUST NOT be identical to the name attribute of any other units element child of that model element, or of any import units element in the CellML Infoset. Where the parent of the units element is a component element, the value of the name attribute MUST NOT be identical to the name attribute of any other units element child of that component element. In any case, the value of the name attribute MUST NOT be equal to a cell in the name column of ?.

2. A units element MAY contain a base_units attribute in the empty namespace. If present, the value of the base_units attribute MUST be equal to either yes or no. If the attribute is present and equal to yes, then the following paragraph does not apply.

3. A units element MAY contain one or more unit element children (?).
The unit element information item

Unit element information items (referred to in this specification as unit elements) are element information items in the CellML 1.2 namespace with local name equal to unit, and with a units element as their parent.

9.1 Specific information items

1. Every unit element MUST contain a units attribute information item in the empty namespace. The value of the units attribute MUST be a valid units reference, as defined in ?. The units element inclusion digraph SHALL be a conceptual digraph defined for the purpose of the constraint in this paragraph, and which contains one node for every units element in the CellML Model. The units element inclusion digraph SHALL contain an arc from units element A to units element B if and only if units element A contains a unit element with units attribute value that is a units reference to units element B. The value of the units attribute MUST NOT be such that the units element inclusion digraph contains one or more cycles.

2. A unit element MAY contain any of the following attribute information items in the empty namespace:

   (a) The prefix attribute. If present, the value of the attribute MUST meet the constraints specified in ?.

   (b) The offset attribute. If present, the value of the attribute MUST be a real number string. If the attribute is present and has value other than a real number string representing 0, then this unit element MUST NOT have any sibling unit elements, and the value of the exponent attribute, if present, MUST be a real number string representing 1.

   (c) The multiplier attribute. If present, the value of the attribute MUST be a real number string.

   (d) The exponent attribute. If present, the value of the attribute MUST be a real number string.
The component element information item

Component element information items (referred to in this specification as component elements) are element information items in the CellML 1.2 namespace with local name component, and which appear as a child of a model element.

10.1 Specific information items

1. Every component element MUST contain a name attribute in the empty namespace. The value of the name attribute MUST be a valid CellML identifier. The value of the name attribute MUST NOT be identical to the name attribute on any other component element or import component element in the CellML Infoset.

2. A component element MAY contain zero or more specific information item children, each of which MUST be of one of the following types:

   (a) a variable element (?).

   (b) an element information item in the MathML namespace, and with local name math, which MUST be the top-level of a content MathML tree, as described in ?.

   (c) a units element (?).
CHAPTER 11

The variable element information item

Variable element information items (referred to in this specification as variable elements) are element information items in the CellML 1.2 namespace with local name equal to variable which appear as a child of a component element.

11.1 Specific information items

1. Every variable element MUST have each of the following attribute information items in the empty namespace:
   
   (a) The name attribute. The value of the name attribute MUST be a valid CellML Identifier. The value of the name attribute MUST NOT be identical to the name attribute on any sibling variable element.
   
   (b) The type attribute. The primary specification does not place any restrictions on the value of the type attribute, but secondary specifications SHOULD define the types supported in that secondary specification.

   A type attribute value of real SHALL be interpreted as indicating that the variable is real valued. A type attribute value of boolean SHALL be interpreted as indicating that there are two possible values, true and false.

2. Every variable element MAY contain one or more of the following attribute information items in the empty namespace:
   
   (a) The public_interface attribute. If the attribute is present, it MUST have value yes or no.
   
   (b) The private_interface attribute. If the attribute is present, it MUST have value yes or no.
   
   (c) The units attribute. If the attribute is present, the value MUST be a valid CellML Identifier, and MUST meet the constraints described in ?. This attribute MUST be present if the type attribute has the value real.
Encapsulation element information items (referred to in this specification as encapsulation elements) are element information items in the CellML 1.2 namespace with local name equal to encapsulation.

### 12.1 Specific information items

1. Every encapsulation element MUST contain one or more component_ref elements (?).
The component_ref element information item

component_ref element information items (referred to in this specification as component_ref elements) are element information items in the CellML 1.2 namespace with local name equal to component_ref.

13.1 Specific information items

1. Every component_ref element MUST contain an attribute information item in the empty namespace and with local name component. The value of this attribute MUST be a valid CellML Identifier, and MUST match the name attribute on a component or import component element in the CellML Infoset.

2. Every component_ref element MAY in turn contain zero or more component_ref element children. In addition, component_ref elements that are children of encapsulation elements MUST contain at least one component_ref element child.
Connection element information items (referred to in this specification as connection elements) are element information items in the CellML 1.2 namespace with local name equal to connection.

### 14.1 Specific information items

1. Every connection element MUST contain a component_1 attribute in the empty namespace. The value of the component_1 attribute MUST be a valid CellML Identifier. The value of this attribute MUST be equal to the name attribute on a component or import component element in the CellML Infoset.

2. Every connection element MUST contain a component_2 attribute in the empty namespace. The value of the component_2 attribute MUST be a valid CellML Identifier. The value of this attribute MUST be equal to the name attribute on a component or import component element in the CellML Infoset.

3. Every connection element MUST contain one or more map_variables elements (?).
map_variables element information items (referred to in this specification as map_variables elements) are element information items in the CellML 1.2 namespace with local name equal to map_variables which appear as a child of a connection element.

15.1 Specific information items

1. Each map_variables element MUST contain a variable_1 attribute. The value of the variable_1 attribute MUST be a valid CellML Identifier. The value of this attribute MUST be equal to the name attribute on a variable element child of the component element referenced by the component_1 attribute on the connection element parent.

2. Each map_variables element MUST contain a variable_2 attribute. The value of the variable_2 attribute MUST be a valid CellML Identifier. The value of this attribute MUST be equal to the name attribute on a variable element child of the component element referenced by the component_2 attribute on the connection element parent.
16.1 Component reference

1. A component reference SHALL be the name of a component, and SHALL be interpreted based on the context within the CellML Model in which it occurs.

2. A component reference present in an information item that is a descendant of a model element SHALL be identical to either the name attribute on a component element or to the name attribute on an import component element.

3. A component reference that is identical to the name attribute on a component element SHALL be treated as a reference to that component element.

4. A component reference that is identical to the name attribute on an import component element SHALL be treated as if the component reference appeared in the imported model, and referred to the name specified in the component_ref attribute of the import component element.

5. It is noted for the avoidance of doubt that CellML Models MAY apply the previous rule recursively, to reference an import component element that in turn references another import component element.

16.2 Variable reference

1. When present in an information item that is a descendant of a component element, a variable reference SHALL be the name of a variable, and SHALL refer to the variable element in the same component with a name attribute identical to the variable reference.

2. In all other cases, a variable reference SHALL consist of a component reference and a variable name. In this case, the variable reference SHALL be treated as if it was present in the component element referenced by the component reference.

16.3 Interpretation of encapsulation

1. For the purposes of this specification, there SHALL be a conceptual encapsulation digraph in which there is one node for every component in the CellML model.

2. Where a component_ref element appears as a child of another component_ref element, then there SHALL be an arc in the encapsulation digraph, and that arc SHALL be from the component referenced by the parent component_ref element, and to the component referenced by the child component_ref element.
3. The encapsulation digraph MUST NOT contain any loops or cycles, and the in-degree of a node MUST NOT exceed one (that is, it must be a forest).

4. The encapsulated set for a component A SHALL be the set of all components to which there exists an arc in the encapsulation digraph from the node corresponding to A.

5. The encapsulation parent for a component A SHALL be the component corresponding to the node that is the parent node in the encapsulation digraph of the node corresponding to A.

6. The sibling set for a component A SHALL be the set of all components that have the same encapsulation parent as A, or in the case that A has no encapsulation parent, SHALL be the set of all components that do not have an encapsulation parent.

7. The hidden set for a component A SHALL be the set of all components B where component B is not in the encapsulated set for component A, and component B is not the encapsulation parent of component A, and component B is not in the sibling set for component A.

8. CellML models MUST NOT contain connection elements such that the component referenced by the component_1 attribute is in the hidden set of the component referenced by the component_2 attribute.

### 16.4 Variable equivalence networks

1. A variable equivalence network SHALL be a graph with one node for every variable element in the CellML model.

2. For every map_variables element present in the CellML Model, there SHALL be an edge in the variable equivalence network.

3. One endpoint of the edge in the variable equivalence network SHALL be the node corresponding to the variable (A) referenced by the component_1 and variable_1 attributes.

4. One endpoint of the edge in the variable equivalence network SHALL be the node corresponding to the variable (B) referenced by the component_2 and variable_2 attributes.

5. CellML models MUST NOT contain any pair of map_variables elements that each make reference to the same sets of variables (without regard to whether the variable_1 attribute of one map_variables element references the variable referenced by the variable_1 or variable_2 attribute of the other).

6. When the component parent element of variable A is in the sibling set of the component parent element of variable B, the applicable interface for variables A and B SHALL be the public interface.

7. Where the component parent element of variable A is in the encapsulated set of the component parent element of variable B, the applicable interface for variable A SHALL be the public interface, and the applicable interface for variable B SHALL be the private interface.

8. Where the component parent element of variable B is in the encapsulated set of the component parent element of variable A, the applicable interface for variable A SHALL be the private interface, and the applicable interface for variable B SHALL be the public interface.

9. For a given variable, if the applicable interface is the public interface, the applicable interface attribute SHALL be the public_interface attribute information item on the corresponding variable element. If the applicable interface is the private interface, the applicable interface attribute SHALL be the private_interface attribute information item on the corresponding variable element.

10. In any case, if the applicable interface attribute is absent, the following rules in this section SHALL still apply as if the applicable interface attribute was present, and had the value no.

11. CellML models MUST NOT contain map_variables elements unless the value of the applicable interface attributes on variables A and B are both yes.
12. The variable equivalence network MUST NOT contain any cycles in the underlying graph (that is, it must be a tree).

13. For the purposes of this specification, the variable elements in a CellML Model SHALL be treated as belonging to one of several disjoint sets of connected variables. Each set of connected variables is a set of all variable elements for which the corresponding nodes in the variable equivalence network form a connected set. Each set of connected variables represents one variable in the mathematical model.

14. For the purposes of this specification, for every set of connected variables, one variable SHALL arbitrarily be selected as the source variable. Within this specification, the variable in the mathematical model is described as if it was in the units specified on the source variable element.

15. All variables in a variable equivalence network MUST have the same type.

16. If any of the variables in a variable equivalence network have a units attribute, all variables in the variable equivalence network MUST contain units that are dimensionally compatible with each other.

17. Two units are dimensionally compatible with each other if their canonical forms are made up of the same base units with the same exponents. The canonical form of a unit is the form obtained by repeated expansion of units into their component units, and simplification to add exponents of identical base units.

### 16.5 Units reference

1. A units reference SHALL be a CellML identifier, and SHALL be interpreted based on the context within the CellML Model in which it occurs.

2. A CellML Infoset MUST NOT contain a units reference to which all scoping rules are inapplicable.

3. Where more than one of the units scoping rules apply, the applicable rule that appears first in this specification SHALL be used.

4. The units scoping rules are as follows:

   (a) Where a units reference appears in an information item that is descended from a component element, and there is a units element child of that component element with a name attribute identical to the units reference, then the units reference SHALL refer to that units element.

   (b) Where a units reference appears in an information item that is descended from the model element, and there is a units element child of that model element with a name attribute identical to the units reference, then the units reference SHALL refer to that units element.

   (c) Where there is an import units element in the CellML Infoset, such that the import units element has a name attribute identical to the units reference, then the units reference SHALL be treated as if the units reference appeared in the imported model, and referred to the name specified in the units_ref attribute of the import units element.

   (d) Where the units reference is equal to a cell in the name column of ?, then the units reference SHALL be a reference to the built-in unit corresponding to that row of the table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Is base unit?</th>
<th>Multiplier and dimensions in terms of base units</th>
<th>Offset from base units</th>
</tr>
</thead>
<tbody>
<tr>
<td>ampere</td>
<td>yes</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>becquerel</td>
<td>no</td>
<td>1 · second -1</td>
<td>0</td>
</tr>
</tbody>
</table>

Continued on next page
<table>
<thead>
<tr>
<th>Name</th>
<th>Is base unit?</th>
<th>Multiplier and dimensions in terms of base units</th>
<th>Offset from base units</th>
</tr>
</thead>
<tbody>
<tr>
<td>candela</td>
<td>yes</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>celsius</td>
<td>no</td>
<td>1 • kelvin</td>
<td>273.15</td>
</tr>
<tr>
<td>coulomb</td>
<td>no</td>
<td>1 • second • ampere</td>
<td>0</td>
</tr>
<tr>
<td>dimensionless</td>
<td>no</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>farad</td>
<td>no</td>
<td>1 • metre-2 • kilogram-1 • second4 • ampere2</td>
<td>0</td>
</tr>
<tr>
<td>gram</td>
<td>no</td>
<td>10 • 3 • kilogram</td>
<td>0</td>
</tr>
<tr>
<td>gray</td>
<td>no</td>
<td>1 • metre2 • second-2</td>
<td>0</td>
</tr>
<tr>
<td>henry</td>
<td>no</td>
<td>1 • metre2 • kilogram-1 • second-2 • ampere-2</td>
<td>0</td>
</tr>
<tr>
<td>hertz</td>
<td>no</td>
<td>1 • second-1</td>
<td>0</td>
</tr>
<tr>
<td>joule</td>
<td>no</td>
<td>1 • metre2 • kilogram-1 • second-2</td>
<td>0</td>
</tr>
<tr>
<td>katal</td>
<td>no</td>
<td>1 • second-1 • mole</td>
<td>0</td>
</tr>
<tr>
<td>kelvin</td>
<td>yes</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>kilogram</td>
<td>yes</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>liter</td>
<td>no</td>
<td>10 • 3 • metre 3</td>
<td>0</td>
</tr>
<tr>
<td>litre</td>
<td>no</td>
<td>10 • 3 • metre 3</td>
<td>0</td>
</tr>
<tr>
<td>lumen</td>
<td>no</td>
<td>1 • candela</td>
<td>0</td>
</tr>
<tr>
<td>lux</td>
<td>no</td>
<td>1 • metre-2 • candela</td>
<td>0</td>
</tr>
<tr>
<td>meter</td>
<td>no</td>
<td>1 • metre</td>
<td>0</td>
</tr>
<tr>
<td>metre</td>
<td>yes</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>mole</td>
<td>yes</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>newton</td>
<td>no</td>
<td>1 • metre • kilogram • second-2</td>
<td>0</td>
</tr>
<tr>
<td>ohm</td>
<td>no</td>
<td>1 • metre2 • kilogram • second-3 • ampere-2</td>
<td>0</td>
</tr>
<tr>
<td>pascal</td>
<td>no</td>
<td>1 • metre-1 • kilogram • second-2</td>
<td>0</td>
</tr>
<tr>
<td>radian</td>
<td>no</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>second</td>
<td>yes</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td>siemens</td>
<td>no</td>
<td>1 • metre-2 • kilogram-1 • second3 • ampere2</td>
<td>0</td>
</tr>
<tr>
<td>sievert</td>
<td>no</td>
<td>1 • metre2 • second-2</td>
<td>0</td>
</tr>
<tr>
<td>steradian</td>
<td>no</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>tesla</td>
<td>no</td>
<td>1 • kilogram • second-2 • ampere-1</td>
<td>0</td>
</tr>
</tbody>
</table>

Continued on next page
### Table 16.1 – continued from previous page

<table>
<thead>
<tr>
<th>Name</th>
<th>Is base unit?</th>
<th>Multiplier and dimensions in terms of base units</th>
<th>Offset from base units</th>
</tr>
</thead>
<tbody>
<tr>
<td>volt</td>
<td>no</td>
<td>$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-3} \cdot \text{ampere}^{-1}$</td>
<td>0</td>
</tr>
<tr>
<td>watt</td>
<td>no</td>
<td>$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-3}$</td>
<td>0</td>
</tr>
<tr>
<td>weber</td>
<td>no</td>
<td>$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-2} \cdot \text{ampere}^{-1}$</td>
<td>0</td>
</tr>
</tbody>
</table>

Table: built-in units

<table>
<thead>
<tr>
<th>Prefix Name</th>
<th>Prefix Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>yotta</td>
<td>24</td>
</tr>
<tr>
<td>zetta</td>
<td>21</td>
</tr>
<tr>
<td>exa</td>
<td>18</td>
</tr>
<tr>
<td>peta</td>
<td>15</td>
</tr>
<tr>
<td>tera</td>
<td>12</td>
</tr>
<tr>
<td>giga</td>
<td>9</td>
</tr>
<tr>
<td>mega</td>
<td>6</td>
</tr>
<tr>
<td>kilo</td>
<td>3</td>
</tr>
<tr>
<td>hecto</td>
<td>2</td>
</tr>
<tr>
<td>deka</td>
<td>1</td>
</tr>
<tr>
<td>deci</td>
<td>-1</td>
</tr>
<tr>
<td>centi</td>
<td>-2</td>
</tr>
<tr>
<td>milli</td>
<td>-3</td>
</tr>
<tr>
<td>micro</td>
<td>-6</td>
</tr>
<tr>
<td>nano</td>
<td>-9</td>
</tr>
<tr>
<td>pico</td>
<td>-12</td>
</tr>
<tr>
<td>femto</td>
<td>-15</td>
</tr>
<tr>
<td>atto</td>
<td>-18</td>
</tr>
<tr>
<td>zepto</td>
<td>-21</td>
</tr>
<tr>
<td>yocto</td>
<td>-24</td>
</tr>
</tbody>
</table>

Table: Prefix values

### 16.6 Interpretation of units

1. The base units SHALL consist of the user defined base units, and the built-in base units (those units defined in rows of ? having ‘yes’ in the ‘Is base unit?’ column).

2. There SHALL be one user defined base unit for every units element that has a base_units attribute in the empty namespace, having value yes.

3. The base unit reduction of a units reference SHALL consist of a real valued offset, a real valued multiplier, and a set of tuples each consisting of a base unit and a real valued exponent. The base unit reduction of a units reference SHALL be determined as follows:

   (a) Where the units reference is to a unit that is a base unit, then the base unit reduction of the units reference SHALL have offset zero, multiplier 1.0, and the set of tuples SHALL have a single member, which SHALL consist of the base units being referenced and the exponent 1.0.

   (b) Where the units reference is to built-in units other than a base unit, then the base unit reduction SHALL be derived from the row of ? for which the value in the ‘Name’ column matches the name of the units.
reference. The offset of the base unit reduction SHALL be equal to the number in the ‘Offset from base units’ column of the row, and the multiplier SHALL be equal to the number at the start of the ‘Multiplier and dimensions in terms of base units’ column of the row. The set of tuples SHALL contain one member for every built-in base unit named in the ‘Multiplier and dimensions in terms of base units’ column of the row, and each of these tuples SHALL contain the built-in unit referenced, and the exponent appearing in superscript immediately after the units name in the table cell.

(c) Where the units reference is to a unit that is neither built-in, nor a base unit, the resultant base unit reduction SHALL be defined as a composition of the base unit reductions referenced from the unit element information items (the operand base unit reductions), in accordance with the following rules:

i. The prefix term is a conceptual property of unit elements, defined here for later use. If the unit element does not have a prefix information item, the prefix term SHALL have value zero. If the prefix attribute information item has a value that is a real number string, then the prefix term SHALL have the corresponding numerical value. Otherwise, the prefix attribute information item MUST have a value taken from the ‘Prefix Name’ column of ?, and the prefix term SHALL have the value taken from the ‘Prefix Term’ column of the same row.

ii. The exponent term is a conceptual property of unit elements, defined here for later use. If a unit element has no exponent attribute information item, the exponent term SHALL have value 1.0. Otherwise, the value of the exponent attribute information item MUST be a real number string, and the value of the exponent term SHALL be the numerical value of that string.

iii. The multiplier term is a conceptual property of unit elements, defined here for later use. The multiplier term SHALL be the real number value of the multiplier attribute information item on the units element (or 1.0 in the absence of such an attribute information item), multiplied by 10.0 raised to the power of the negative of the product of the prefix term and the exponent term.

iv. The offset term is a conceptual property of unit elements, defined here for later use. If a unit element has no offset attribute information item, the offset term SHALL have value zero. Otherwise, the value of the offset attribute information item MUST be a real number string, and the value of the offset term SHALL be the numerical value of that string.

v. Where the units reference is to a units element with a single unit child element, then the resultant base unit reduction SHALL have multiplier equal to the product of the multiplier of the operand base unit reduction and the multiplier term of the unit element. It SHALL have offset equal to the sum of the offset of the operand base unit reduction and the offset term of the unit element.

vi. Where the units reference is to a units element with a number of unit child elements not equal to one, then the resultant base unit reduction SHALL have multiplier equal to the product of the multipliers of each operand base unit reduction, and the multiplier term of each unit element. It SHALL have offset equal to zero.

vii. The set of tuples on the resultant base unit reduction SHALL have one member for every distinct base unit present in the set of tuples for any of the operand base unit reductions. The exponent alongside each of these base units in the resultant base unit reduction SHALL be the sum, across all tuples for the base unit from operand base unit reductions, of pairwise products of the exponent term on the corresponding unit element and the exponent from the tuple.

16.7 The effect of units on variables

1. If present, the units attribute on a variable element MUST be a valid units reference. The target of this units reference is referred to as the variable units, and the corresponding base units reduction is referred to as the variable base unit reduction.

2. The variable base unit reduction of a variable element MUST have an identical set of tuples to the set of tuples on the source variable base units reduction. Two sets of tuples SHALL be considered identical if and only
if neither set contains any tuple not present in the other. Two tuples are considered identical if both the base units and exponent on the tuple are the same.

3. The following symbols are defined for the purposes of the formulae in the ‘Interpretation of Mathematics’ section:
   (a) \( m_V \) is the multiplier on the variable base unit reduction.
   (b) \( o_V \) is the offset on the variable base unit reduction.
   (c) \( m_S \) is the multiplier on the source variable base unit reduction.
   (d) \( o_S \) is the offset on the source variable base unit reduction.

### 16.8 Interpretation of the \textit{type} attribute

1. This section applies to the interpretation of the \textit{type} attribute, a mandatory attribute information item on a \textit{variable} element.

2. The \textit{type} attribute specifies the type that a variable should take. A type \textbf{SHALL} define the type of variables that a variable may take, but \textbf{SHOULD NOT} be used to describe the precise implementation-specific details of how data is represented.

### 16.9 Interpretation of imports

1. Each \textit{import} element present in a CellML Infoset (the importing infoset) \textbf{SHALL} define a new and separate instance of the CellML Infoset referenced by the href attribute (the imported infoset).

2. The following \textit{component} elements \textbf{SHALL} be “pertinent \textit{component} elements”:
   (a) all \textit{component} elements in the top-level CellML Infoset for the CellML Model, and,
   (b) all \textit{component} elements referenced by import components in the top-level CellML Infoset, and,
   (c) all \textit{component} elements that are descendants in the encapsulation digraph of a pertinent \textit{component} element.

### 16.10 Interpretation of the mathematics

1. Where full content MathML can be translated into strict content MathML using a rule in the MathML specification, then the rules in this specification \textbf{SHALL} be applied as if the equivalent strict content MathML form had been used.

2. Every MathML element in the CellML Model which appears as a direct child information item of the MathML \textit{math} element information item, which in turn appears as a child information item of a pertinent \textit{component} element, \textbf{SHALL} be treated as a statement which holds true unconditionally.

3. Every variable name given using the MathML \textit{ci} element \textbf{MUST} either be:
   (a) a reference to a variable name bound by a \textit{bind} element ancestor of the \textit{ci} element.
   (b) a reference to a variable within the \textit{component} element ancestor the MathML is contained within.

4. Where the same variable name is bound by more than one \textit{bind} element ancestor of a \textit{ci} element, the \textit{ci} element \textbf{SHALL} refer only to the bound variable from the \textit{bind} element that is the descendant of all those other \textit{bind} elements. Where the same variable name is defined by a \textit{bind} element ancestor of a \textit{ci} element
and a model variable, the variable SHALL be treated as a bound variable, and SHALL NOT be treated as a model variable.

5. Where a MathML ci element refers to a variable that is not in the independent variable set, the variable SHALL be treated as if it was a lexical closure over the independent variables of the application of each independent variable to the function giving the value of the variable in terms of each value of each independent variable.

6. The MathML csymbol element information item with name evaluatedAt in content dictionary cellml1 shall refer to a function of three operands. The first operand MUST be a MathML ci element referring to a variable that is in independent variable set. The application of this csymbol element shall be interpreted as meaning the value of the third operand, modified so that all lexical closures of the independent variable given as the first operand are removed and substituted for the value of the second operand.

7. Every variable reference to a variable of type real SHALL be treated as a linear expression \( \frac{m_U}{m_S} \cdot x - o_V + \frac{m_S}{m_V} \cdot o_S \)

   In this equation, \( x \) represents the variable in the mathematical model, in the units on the source variable element, while the remaining variables SHALL be interpreted as specified in ?.

8. The MathML elements requiring additional type information are:
   
   (a) Every MathML cn element
   
   (b) Every MathML ci element that appears within a bvar element information item under a bind element information item.

9. Every MathML element requiring additional type information (as defined above) MUST have an attribute information item in the CellML 1.2 namespace, with local name type. The value of this attribute information item MUST be a valid type name.

10. A MathML element requiring additional type information (as defined above) MAY have an attribute information item in the CellML 1.2 namespace, with local name units. Where the type attribute takes the value real, the units attribute MUST be present. The value of this attribute information item MUST be a valid units reference. The referenced units SHALL NOT affect the mathematical interpretation of the CellML model. However, CellML Processing Software MAY use this information to assist the user in the detection and correction of units errors in the CellML Model.

### 16.11 Type inference requirement

1. In a valid model, it SHALL be possible to identify a single type for every mathematical expression or sub-expression.

2. The type identified for a MathML ci element for the purposes of this section SHALL be the same type specified by the type attribute of the corresponding variable, or where the ci element refers to a bound variable, the type specified on ci child of the corresponding bvar element.

3. The type identified for a MathML cn element for the purposes of this section SHALL be the same type specified by the type attribute in the CellML 1.2 namespace on the cn element.

4. The MathML element information items true and false SHALL be identified as having type boolean.

5. The MathML element information items exponentiale, pi, and eulergamma SHALL be identified as having type real.

6. The MathML piecewise element SHALL be identified as having the type of the first case value. All case values in the piecewise element, and if present, the otherwise expression MUST also be identified as having the same type. The case conditions MUST be identified as having type boolean.

7. The type identification for all remaining MathML element information items, including for the apply element information item with different operators, are left up to the secondary specifications.
8. Secondary specifications MUST be written so that the type of any valid mathematical expression can be uniquely inferred from the types of the arguments.
Chapter 16. Interpretation of CellML models
CHAPTER 17

Normative References

RFC 2119: Key words for use in RFCs to Indicate Requirement Levels March 1997
Extensible Markup Language (XML) 1.0 (Fourth Edition) 16 August 2006
XML Information Set (Second Edition) 4 February 2004
Namespaces in XML (Second Edition) 16 August 2006
RDF/XML Syntax Specification (revised) 10 February 2004
Mathematical Markup Language (MathML) Version 3.0 21 October 2003
XML Linking Language (XLink) Version 1.0 27 June 2001
Secondary specifications


2. CellML Secondary Specifications MUST NOT conflict with, or override, any of the assertions or requirements in this specification.

3. CellML Processing software MUST either:
   (a) support one or more specific CellML Secondary Specifications, and correctly process all models that comply both with this specification and with one or more of the selected CellML Secondary Specifications, or,
   (b) correctly process all models that comply with this specification.

4. Despite the previous paragraph, CellML Secondary Specifications MAY identify particular unambiguously defined subsets of all CellML Models as being optional for implementors, and CellML Processing Software MAY choose to exercise such options.
CHAPTER 19

Summary of all TODO’s