
CellML Specification Documentation

Release

The CellML Editorial Board

May 23, 2013

CONTENTS

1	Definitions	3
2	General matters	5
2.1	CellML and XML	5
2.2	Equivalent CellML infosets	5
2.3	Character information items	5
2.4	Use of namespaces	6
2.5	Extension information items	6
2.6	Identifiers	6
2.7	Specific information items	7
2.8	RDF element information items	7
3	Data representation formats in CellML	9
4	The <code>model</code> element information item	11
4.1	Top-level of CellML infosets	11
4.2	Specific information items	11
5	The <code>import</code> element information item	13
5.1	Specific information items	13
6	The <code>import units</code> element information item	15
6.1	Specific information items	15
7	The <code>import component</code> element information item	17
7.1	Specific information items	17
8	The <code>units</code> element information item	19
8.1	Specific information items	19
9	The <code>unit</code> element information item	21
9.1	Specific information items	21
10	The <code>component</code> element information item	23
10.1	Specific information items	23
11	The <code>variable</code> element information item	25
11.1	Specific information items	25
12	The <code>group</code> element information item	27
12.1	Specific information items	27

13 The <code>relationship_ref</code> element information item	29
13.1 Specific information items	29
14 The <code>component_ref</code> element information item	31
14.1 Specific information items	31
15 The <code>connection</code> element information item	33
15.1 Specific information items	33
16 The <code>map_components</code> element information item	35
16.1 Specific information items	35
17 The <code>map_variables</code> element information item	37
17.1 Specific information items	37
18 Interpretation of CellML models	39
18.1 Interpretation of imports	39
18.2 Units reference	39
18.3 Interpretation of units	41
18.4 Component reference	42
18.5 Variable reference	43
18.6 Interpretation of initial values	43
18.7 Effect of units on variables	43
18.8 Interpretation of mathematics	44
18.9 Interpretation of grouping	44
18.10 Variable equivalence networks	45
19 References	47
20 Summary of all TODO's	49

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

This document specifies version 1.2 of **CellML**, 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 describe CellML. An *informative specification* is available which is annotated with much more explanatory material.

Todo

Provide a ‘proper’ URL for the informative version of the specification.

DEFINITIONS

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 [RFC 2119](#).

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 [XML Information Set](#).

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 which processes CellML in accordance with the rules of this document.

Namespace An XML namespace, as defined in [Namespaces in XML](#).

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

CellML 1.0 namespace The namespace `http://www.cellml.org/cellml/1.0#`.

CellML 1.1 namespace The namespace `http://www.cellml.org/cellml/1.1#`.

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#`.

Extension namespace Any namespace which is not a CellML namespace, and is not the RDF namespace, the MathML namespace, `http://www.w3.org/XML/1998/namespace`, `http://www.w3.org/2000/xmlns/`, `http://www.w3.org/1999/xlink`, or the empty namespace.

CellML information item Any information item in a CellML namespace.

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 which 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 [RDF Concepts and Abstract Syntax](#).

GENERAL MATTERS

2.1 CellML and XML

1. Every CellML infoset SHALL be represented in an XML document which conforms with the well-formedness requirements of the [XML 1.0](#).
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 of the XML file in ways which 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 which are the direct child of an element information item in a CellML namespace, or in the MathML or 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 which 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 which 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 which 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 which contain information items in a CellML namespace other than the CellML 1.0, CellML 1.1 and CellML 1.2 namespaces.
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 RDF/XML Syntax Specification](#).
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 [MathML 2.0](#).

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, which 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 which 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.
7. For the avoidance of doubt, extension information items **MUST NOT** contain CellML information items as descendants.

2.6 Identifiers

1. Any element information item in the CellML namespace **MAY** contain an attribute information item with local 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 XML 1.0](#).

2.7 Specific information items

1. A specific information item is an information item which is not:
 - (a) A comment information item;
 - (b) A character information item;
 - (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 RDF/XML Syntax Specification](#).
3. An RDF element information item **MUST NOT** appear in a CellML infoSet except as allowed by rule 1 above.

Todo

Find out which rule 1 we are we talking about here. Andre: it is the rule 1 just here (2.8.1 currently) but need to work out how to properly reference other rules. Probably shouldn't rely on section numbers being consistent.

4. CellML processing software **MUST NOT** allow the mathematical interpretation of a CellML model to be altered by any information present in RDF data.
5. 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 Syntax Specification](#).
6. Two CellML infoSets which differ only by the addition, removal, or modification of RDF element information items (or information items descended from them), but which have the same set of all RDF triples, **SHALL** be termed RDF-equivalent CellML infoSets.
7. CellML processing software **MUST NOT** treat RDF-equivalent CellML infoSets differently.

DATA REPRESENTATION FORMATS IN CELLML

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 under-scores.
 - (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.
 - (b) SHALL, when the integer being represented is negative, 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.
 - (c) SHALL, when the integer being represented is non-negative, 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) SHALL, when the basic real number being represented is negative, 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) SHALL, other than the sign indicator and the decimal point separator, consist only of European numeric characters.
5. A real number string:

- (a) SHALL be a base 10 representation of a real number $r = s \cdot 10^e$, 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 value of the exponent. The exponent separator character SHALL be either the basic Latin 'E' character U+0045 or the basic Latin 'e' character U+0065.

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 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) A `group` element;
 - (d) An `import` element; or
 - (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 namespace with local name equal to `import`.

5.1 Specific information items

1. Every `import` element MUST contain 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.
2. 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 units` element; or
 - (b) An `import component` element.
3. The imported CellML infoset SHALL NOT be equivalent to this CellML infoset. Any CellML infoset imported, directly or indirectly, by the imported CellML infoset SHALL NOT be equivalent to this CellML infoset.

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 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 `units` element or `import 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` element 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 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 `component element` or `import 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 element` 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 namespace with local name equal to `units`, and with either a `model` element or a `component` 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.
 - (a) 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 info set.
 - (b) 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.
 - (c) In any case, the value of the `name` attribute MUST NOT be equal to a cell in the name column of the *Built-in units* table.
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 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 reference* section.
 - (a) The `units` element inclusion digraph SHALL be a conceptual digraph defined for the purpose of the constraint in the next paragraph, and SHALL contain 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*.
 - (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 the *Interpretation of units* section.
 - (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 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 `units element`;
 - (b) A `variable element`; or
 - (c) 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 [MathML 2.0](#).

THE VARIABLE ELEMENT INFORMATION ITEM

`variable` element information items (referred to in this specification as `variable` elements) are element information items in the CellML namespace with local name equal to `variable`, and 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 `units` attribute. The value of the `units` attribute MUST be a valid CellML Identifier, and MUST meet the constraints described in the *Effect of units on variables* section.
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 `in`, `out`, or `none`.
 - (b) The `private_interface` attribute. If the attribute is present, it MUST have value `in`, `out`, or `none`. It MUST NOT have value `in` if the `public_interface` attribute also has the value `in`.
 - (c) The `initial_value` attribute. If the attribute is present, it MUST meet the requirements described **HERE**.

Todo

Need to provide a link to where the requirements for the `initial_value` attribute are.

THE GROUP ELEMENT INFORMATION ITEM

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

12.1 Specific information items

1. Every group element MUST contain one or more `relationship_ref` elements.
2. Every group element MUST contain one or more `component_ref` elements.

THE RELATIONSHIP_REF ELEMENT INFORMATION ITEM

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

13.1 Specific information items

1. Every `relationship_ref` element MUST contain exactly one attribute information item with local name `relationship`. This limit of one attribute information item SHALL apply without regard to the namespace in which the attribute information item is, and SHALL, to the extent that it conflicts, override the ordinary rules for extension information items.
2. In the event that the `relationship` attribute is in the empty namespace, it MUST either take the value `encapsulation` or `containment`.
3. There MUST NOT exist a sibling `relationship_ref` element such that the namespace of the `relationship` attribute on the sibling element is identical to the namespace of the `relationship` attribute on this element, and the value of the `relationship` attribute on the sibling element is identical to the value of the `relationship` attribute on this element.
4. A `relationship_ref` element MAY contain an attribute in the empty namespace with local name `name`. In this case, it MUST NOT have a `relationship` attribute in the empty namespace with value `encapsulation`. The value of the `name` attribute MUST be a valid CellML identifier.

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 namespace with local name equal to `component_ref`.

14.1 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` element or an `import component` element in the CellML infoset.
2. Every `component_ref` element MAY in turn contain zero or more `component_ref` element children.
3. In addition, `component_ref` elements which are children of `group` elements MUST contain at least one `component_ref` element child.

THE CONNECTION ELEMENT INFORMATION ITEM

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

15.1 Specific information items

1. Every `connection` element MUST contain exactly one `map_components` element.
2. Every `connection` element MUST contain one or more `map_variables` elements.

THE `MAP_COMPONENTS` ELEMENT INFORMATION ITEM

`map_components` element information items (referred to in this specification as `map_components` elements) are element information items in the CellML namespace with local name equal to `map_components`, and which appear as a child of a `connection` element.

16.1 Specific information items

1. Each `map_components` 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. Each `map_components` 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. It MUST NOT be equal to the value of the `component_1` attribute.

THE `MAP_VARIABLES` ELEMENT INFORMATION ITEM

`map_variables` element information items (referred to in this specification as `map_variables` elements) are element information items in the CellML namespace with local name equal to `map_variables`, and which appear as a child of a `connection` element.

17.1 Specific information items

1. Each `map_variables` element MUST contain a `variable_1` attribute in the empty namespace. 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 `map_components` element which is a sibling of this element.
2. Each `map_variables` element MUST contain a `variable_2` attribute in the empty namespace. 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 `map_components` element which is a sibling of this element.

INTERPRETATION OF CELLML MODELS

18.1 Interpretation of imports

1. Each `import` element present in a CellML infoset (the importing infoset) SHALL define a new and separate instance of the CellML infoset referenced by the `href` attribute (the imported infoset).
2. The following component elements SHALL be “pertinent component elements”:
 - (a) All component elements in the top-level CellML infoset for the CellML model;
 - (b) All component elements referenced by `import` component elements in the top-level CellML infoset; and
 - (c) All component elements which are descendants in the encapsulation digraph of a pertinent component element.

18.2 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 which 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 which 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 which 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 the *Built-in units* table, then the units reference SHALL be a reference to the built-in unit corresponding to that row of the table.

Table 18.1: Built-in units

Name	Base unit?	Multiplier and dimensions in terms of base units	Offset from base units
ampere	yes	-	-
becquerel	no	$1 \cdot \text{second}^{-1}$	0
candela	yes	-	-
celsius	no	$1 \cdot \text{kelvin}$	273.15
coulomb	no	$1 \cdot \text{second} \cdot \text{ampere}$	0
dimensionless	no	1	0
farad	no	$1 \cdot \text{metre}^{-2} \cdot \text{kilogram}^{-1} \cdot \text{second}^4 \cdot \text{ampere}^2$	0
gram	no	$10^{-3} \cdot \text{kilogram}$	0
gray	no	$1 \cdot \text{metre}^2 \cdot \text{second}^{-2}$	0
henry	no	$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-2} \cdot \text{ampere}^{-2}$	0
hertz	no	$1 \cdot \text{second}^{-1}$	0
joule	no	$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-2}$	0
katal	no	$1 \cdot \text{second}^{-1} \cdot \text{mole}$	0
kelvin	yes	-	-
kilogram	yes	-	-
liter	no	$10^{-3} \cdot \text{metre}^3$	0
litre	no	$10^{-3} \cdot \text{metre}^3$	0
lumen	no	$1 \cdot \text{candela}$	0
lux	no	$1 \cdot \text{metre}^{-2} \cdot \text{candela}$	0
meter	no	$1 \cdot \text{metre}$	0
metre	yes	-	-
mole	yes	-	-
newton	no	$1 \cdot \text{metre} \cdot \text{kilogram} \cdot \text{second}^{-2}$	0
ohm	no	$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-3} \cdot \text{ampere}^{-2}$	0
pascal	no	$1 \cdot \text{metre}^{-1} \cdot \text{kilogram} \cdot \text{second}^{-2}$	0
radian	no	1	0
second	yes	-	-
siemens	no	$1 \cdot \text{metre}^{-2} \cdot \text{kilogram}^{-1} \cdot \text{second}^3 \cdot \text{ampere}^2$	0
sievert	no	$1 \cdot \text{metre}^2 \cdot \text{second}^{-2}$	0
steradian	no	1	0
tesla	no	$1 \cdot \text{kilogram} \cdot \text{second}^{-2} \cdot \text{ampere}^{-1}$	0
volt	no	$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-3} \cdot \text{ampere}^{-1}$	0
watt	no	$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-3}$	0
weber	no	$1 \cdot \text{metre}^2 \cdot \text{kilogram} \cdot \text{second}^{-2} \cdot \text{ampere}^{-1}$	0

Table 18.2: Prefix values

Name	Value
yotta	24
zetta	21
exa	18
peta	15
tera	12
giga	9
mega	6
kilo	3
hecto	2
deka	1
deci	-1
centi	-2
milli	-3
micro	-6
nano	-9
pico	-12
femto	-15
atto	-18
zepto	-21
yocto	-24

18.3 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 the *Built-in units* table having ‘yes’ in the ‘Base unit?’ column).
2. There SHALL be one user defined base unit for every `units` element in the CellML model which 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 which is a base unit, then the base unit reduction of the units reference SHALL have offset 0.0, 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 which 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` attribute information item, the prefix term SHALL have value 0.0. If the `prefix` attribute information item has a value which 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 'Name' column of the *Prefix values* table, and the prefix term SHALL have the value taken from the 'Value' 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 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 0.0. 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 1.0, 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 0.0.
- 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.

18.4 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 which 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 which is identical to the `name` attribute on a `component` element SHALL be treated as a reference to that `component` element.
4. A component reference which 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 which in turn references another `import component` element.

18.5 Variable reference

1. When present in an information item which 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.

18.6 Interpretation of initial values

1. This section applies to the interpretation of the `initial_value` attribute, when it appears as an attribute information item on a `variable` element.
2. The `initial_value` attribute MUST either be a real number string, or a variable reference.
3. Where the `initial_value` attribute has a real number value, it SHALL be interpreted as a statement that the variable on which the attribute appears is equal to that real number value, under the conditions when the initial value holds.
4. Where the `initial_value` attribute is a variable reference, it SHALL be interpreted as a statement that the variable on which the attribute appears is equal to the referenced variable under the conditions when the initial value holds.

Todo

Need a reasonable definition of the conditions when the initial value holds, because we can't really use the 1.1 version because it is problematic for a number of reasons.

18.7 Effect of units on variables

1. 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 element 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.

18.8 Interpretation of mathematics

1. Every MathML element in the CellML model, which appears as a direct child information item of the MathML `math` element information item, which in turn appears as a child information item of a pertinent component element, SHALL be treated as a statement which holds true unconditionally.
2. Every MathML element which appears as a direct child information item of the MathML `math` element information item SHALL be treated as a statement which holds true unconditionally.
3. Every variable name given using the MathML `ci` element SHALL be treated as a variable reference within the component element ancestor the MathML is contained within.
4. Every such variable reference SHALL be treated as a linear expression $\frac{m_V}{m_S} \cdot x - o_V + \frac{m_S}{m_V} \cdot o_S$. In this equation, x represents the variable in the mathematical model, in the units of the source variable element, while the remaining variables SHALL be interpreted as specified in the *Effect of units on variables* section.
5. Every MathML `cn` element MUST have an attribute information item in the CellML namespace, with local name `units`. 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.

18.9 Interpretation of grouping

1. Two `relationship_ref` elements SHALL be considered to refer to the same relationship if and only if all of the following conditions hold:
 - (a) The attribute information item with local name `relationship` is in an identical namespace on both element information items;
 - (b) The attribute information item with local name `relationship` has identical values on each of the element information items; and
 - (c) Either the attribute information item in the empty namespace and with local name `name` is absent on both element information items, or, it is present on both element information items and has identical value.
2. For every distinct relationship referred to by a `relationship_ref` element in the CellML model, there SHALL be a conceptual relationship digraph in which there is one node for every component in the CellML model.
3. Where a `component_ref` element appears as a child of another `component_ref` element, then for all `relationship_ref` elements which are children of the ancestral group element, there SHALL be an arc in the relationship digraph corresponding to the relationship referenced by the `relationship_ref` element, 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.
4. The term encapsulation digraph SHALL refer to the relationship digraph for the relationship corresponding to the `relationship_ref` attribute in the empty namespace and with value `encapsulation` (and with no `name` attribute).
5. The encapsulation digraph MUST NOT contain any loops, and MUST NOT contain any cycles in the underlying graph (that is, it must be a tree).
6. 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 .
7. The encapsulation parent for a component A SHALL be the component corresponding to the node which is the parent node in the encapsulation digraph of the node corresponding to A .

8. The sibling set for a component *A* SHALL be the set of all components which have the same encapsulation parent as *A*, or in the case that *A* has no encapsulation parent, SHALL be the set of all components which do not have an encapsulation parent.
9. 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*.
10. CellML models MUST NOT contain `map_components` 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.

18.10 Variable equivalence networks

1. A variable equivalence network SHALL be a directed 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 arc in the variable equivalence network.
3. One endpoint of the arc 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 arc 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 which 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 `none`.
11. CellML models MUST NOT contain `map_variables` elements unless the value of the applicable interface attributes on variables *A* and *B* are both either `in` or `out`, and those two attribute values are different from each other.
12. The direction of the arc SHALL be from the node corresponding to the variable with the applicable interface attribute equal to `in`, and SHALL be to the node corresponding to the variable with the applicable interface attribute equal to `out`.
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 the set of all

`variable` elements for which the corresponding nodes in the variable equivalence network form a weakly connected subgraph. Each set of connected variables represents one variable in the mathematical model.

14. In every set of connected variables, there **MUST** be exactly one `variable` element which has neither a public interface of `in` nor a private interface of `in`. This `variable` element is referred to as the source `variable` element. Within this specification, the variable in the mathematical model is described as if it was in the units specified on the source `variable` element.

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)
Resource Description Framework (RDF): Concepts and Abstract Syntax (10 February 2004)
RDF/XML Syntax Specification (Revised) (10 February 2004)
Mathematical Markup Language (MathML) Version 2.0 (Second Edition) (21 October 2003)
XML Linking Language (XLink) Version 1.0 (27 June 2001)

SUMMARY OF ALL TODO'S

Todo

Provide a 'proper' URL for the informative version of the specification.

(The *original entry* is located in abstract.rst, line 16.)

Todo

Need a reasonable definition of the conditions when the initial value holds, because we can't really use the 1.1 version because it is problematic for a number of reasons.

(The *original entry* is located in /var/build/user_builds/cellml-specifications-sphinx-testing/checkouts/latest/source/model-interpretation.rst, line 345.)

Todo

Need to provide a link to where the requirements for the `initial_value` attribute are.

(The *original entry* is located in /var/build/user_builds/cellml-specifications-sphinx-testing/checkouts/latest/source/variable-element-item.rst, line 37.)

Todo

Find out which rule 1 we are we talking about here. Andre: it is the rule 1 just here (2.8.1 currently) but need to work out how to properly reference other rules. Probably shouldn't rely on section numbers being consistent.

(The *original entry* is located in /var/build/user_builds/cellml-specifications-sphinx-testing/checkouts/latest/source/general-matters.rst, line 162.)
