© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

CalConnect TC UnitsML

AsciiUnits syntax specification

Committee Draft Standard

Warning for drafts

This document is not a CalConnect Standard. It is distributed for review and comment, and is subject to change without notice and may not be referred to as a Standard. Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

The Calendaring and Scheduling Consortium, Inc. 2025

© 2025 The Calendaring and Scheduling Consortium, Inc.

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from the address below.

The Calendaring and Scheduling Consortium, Inc.

4390 Chaffin Lane
McKinleyville
California 95519
United States of America

copyright@calconnect.org
www.calconnect.org

Contents

Foreword v

Introduction vi
General vi
Purpose vi
Differences with other technologies vi

1. Scope 1

2. Normative references 1

3. Terms and definitions 1

4. Principles 2

5. Representable elements 2
5.1. General 2
5.2. Unit systems 2
5.3. Units 3
5.4. Prefixes 3
5.5. Dimensions 4
5.6. Quantities 4

6. Architecture 4
6.1. General 4
6.2. Statement 4
6.3. Units 5
6.4. Values of units 5

7. AsciiUnits prefixes 5

8. AsciiUnits dimensions 6
8.1. General 6
8.2. Basic dimensions 6
8.3. Dimensions with mathematical operations 7
8.4. Composed dimensions 7

9. AsciiUnits units 8
9.1. General 8
9.2. Basic units 8
9.3. Prefixed units 9
9.4. Composite units 11
9.5. Symbol ambiguity resolution 17

10. Unit metadata 18
10.1. General 18
10.2. Quantity 19
10.3. Symbol representation 23
10.4. Name 25
10.5. Multiplier 26

11. Using AsciiUnits in AsciiMath 28
11.1. General 28
11.2. Notation 28
11.3. Representation in UnitsML in MathML 28
11.4. Representing alternative unit symbols 30

Appendix A (normative) EBNF grammar 31
A.1. General 31
A.2. EBNF notation 31
A.3. Grammar definition 31
A.4. Examples with grammar breakdown 33
A.5. Railroad diagrams 34
A.6. Grammar validation 35

Appendix B (informative) Fixed symbols 36
B.1. General 36
B.2. Dimension symbols 36

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

B.3. Non-SI units who values in SI units that are experimentally determined.....36
B.4. Natural units and atomic units.....36

Appendix C (informative) Ambiguous unit symbols..... 37

Appendix D (normative) Test suite.....39
D.1. General..... 39
D.2. Basic units..... 39
D.3. Prefixed units..... 41
D.4. Composite units..... 43
D.5. Dimension expressions..... 47
D.6. Units with metadata..... 48

Bibliography..... 50

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Foreword

The Calendaring and Scheduling Consortium (“CalConnect”) is a global non-profit organization with the aim to facilitate interoperability of collaborative technologies and tools through open standards.

CalConnect works closely with international and regional partners, of which the full list is available on our website (https://www.calconnect.org/about/liaisons-and-relationships).

The procedures used to develop this document and those intended for its further maintenance are described in the CalConnect Directives.

In particular the different approval criteria needed for the different types of CalConnect documents should be noted. This document was drafted in accordance with the editorial rules of the CalConnect Directives.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CalConnect shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be provided in the Introduction.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

This document was prepared by Technical Committee *UnitsML*.

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Introduction

General

UnitsML is a suite of technologies that facilitate digital processing of units of measurement and measurement quantities.

AsciiUnits is a text-based syntax for representing units of measurement in a compact and human-readable format, designed to be fully compatible with and interoperable with the UnitsML XML language.

Purpose

Key features of AsciiUnits include:

Automatic prefix parsing	Prefixes attached to unit codes are automatically recognized and processed.
Automatic dimension calculation	Dimensions are automatically calculated based on the prefixes and units represented.
Mathematical operations	Support for mathematical operations on units and dimensions such as addition, subtraction, division, exponentiation, square root, grouping and parentheses.
Metadata support	Ability to add additional information such as quantity references, custom symbols, and names.
UnitsML UnitsDB integration	The UnitsML UnitsDB database is the basis used by AsciiUnits to define the units and prefixes available for various applications.

Differences with other technologies

UCUM

UCUM is a compact coding system for the representation of units of measurement, primarily designed for electronic data interchange in clinical and healthcare contexts. Originally developed by the Regenstrief Institute and adopted by HL7, UCUM is widely used in electronic health records and clinical information systems.

While AsciiUnits and UCUM (Unified Code for Units of Measure) are both text-based systems for representing units of measurement, they differ in several key aspects.

Table 1

Feature	AsciiUnits	UCUM
Syntax approach	Uses intuitive mathematical operators (`*`, `/`, `^`)	Uses special characters (`.` for multiplication, `/` for division)
Readability	Designed for human readability and ease of typing	Optimized for compactness and machine processing
XML integration	Direct mapping to UnitsML XML	No built-in XML representation
Metadata support	Supports additional metadata (quantity references, custom symbols)	No metadata support
Dimensional analysis	Automatic dimension calculation	No built-in dimensional analysis

EXAMPLE — Example comparison between AsciiUnits and UCUM

Unit	AsciiUnits	UCUM
Meter	`m`	`m`
Square meter	`m^2`	`m2`
Meter per second	`m/s`	`m/s`

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Unit	AsciiUnits	UCUM
Kilogram meter per second squared	`kg*m/s^2`	`kg.m/s2`
Millimole per liter	`mmol/L`	`mmol/l`

QUDT

QUDT (Quantities, Units, Dimensions and Types) is a semantic web framework for representing quantities, units, dimensions, and types in a consistent and interoperable manner based on RDF (Resource Description Framework) and OWL (Web Ontology Language).

QUDT is designed to support the semantic web and linked data applications, enabling interoperability between different systems and applications.

While AsciiUnits and QUDT share some similarities, they differ in their focus and implementation.

QUDT relies on RDF and OWL for its representation and does not provide a text-based syntax for representing units of measurement. AsciiUnits allows for accessible encoding of units in documents and text files, making it possible for authors to represent measurement quantities in a clear and concise manner.

AsciiUnits being a text-based syntax, is allows for conversion to and from UnitsML XML, which is a more structured and machine-readable format. This makes AsciiUnits suitable for applications where human readability and ease of typing are important, while QUDT is more focused on semantic web applications.

Digital SI framework

The Digital SI framework is an initiative to modernize the International System of Units (SI) by providing a digital representation of SI units and their relationships. It aims to enhance the accessibility and usability of SI units in digital applications, including scientific computing, data analysis, and machine learning.

The Digital SI framework provides a comprehensive set of tools and resources for representing SI units, including a digital representation of SI base and derived units, as well as a set of guidelines for using SI units in digital applications.

While AsciiUnits is designed to be a text-based syntax for representing units of measurement, the Digital SI framework focuses on providing a digital representation of SI units and their relationships. AsciiUnits can be used in conjunction with the Digital SI framework to provide a human-readable representation of SI units and their relationships, making it easier for users to work with SI units in digital applications.

Other technologies

AsciiUnits is designed to be compatible with and interoperable with other technologies for representing units of measurement, including UCUM, QUDT, and the Digital SI framework. It can be used in conjunction with these technologies to provide a comprehensive solution for representing units of measurement in scientific and engineering applications.

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

AsciiUnits syntax specification

1. Scope

AsciiUnits syntax is designed to be simple and intuitive, allowing users to express complex unit expressions using a straightforward notation.

2. Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

UnitsML XML schema, https://www.unitsml.org

W3C TR mathml-units, https://www.w3.org/TR/mathml-units/

3. Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1.
unit
standard quantity used to express a physical quantity

3.2.
basic unit
unit not derived from other units

3.3.
composite unit
unit derived from two or more basic units conjoined by multiplication or division

3.4.
prefix
modifier to a unit with a fixed factor

3.5.
quantity
type of measurement that is represented by a unit

EXAMPLE distance, moment of force, temperature, energy.

3.6.
dimension
basic physical quantity that can be expressed in terms of fundamental quantities

EXAMPLE length, mass, time, plane angle.

3.7.
unit name
name of a unit

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

3.8.
unit symbol
abbreviated representation of a unit

3.9.
unit reference
reference identifier to a unit in a database

3.10.
unit dimensions
fundamental quantities to which the unit applies

4. Principles

AsciiUnits follows the following key principles to ensure clarity and usability in representing units.

Typability	Symbols are meant to be easily typed on standard keyboards, avoiding complex characters or symbols.
Non-ambiguous	Unit representation and composure syntax should lead to exactly one interpretation, ensuring that each expression has an unambiguous meaning.
Extensible	New units and new prefixes can be added to the system without changing the syntax.

5. Representable elements

5.1. General

AsciiUnits allows for the encoding of the following types of elements:

— Unit systems
— Units
— Dimensions
— Quantities
— Prefixes in conjunction with units

5.2. Unit systems

A unit system is a collection of units of measure that forms a coherent framework for converting between different units of measurement.

UnitsML defines the following unit systems.

Table 3 — UnitsML unit systems

ID	Name	Acceptable to be used in the SI system
`SI_base`	SI	true
`SI_derived_special`	SI-derived special	true
`SI_derived_non-special`	SI-derived non-special	true
`non-SI_acceptable`	non-SI but acceptable to be used together	true

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

ID	Name	Acceptable to be used in the SI system
non-SI_not_acceptable	non-SI and unacceptable to be used together	false

Each unit in UnitsML belongs to exactly one unit system.

5.3. Units

The AsciiUnits syntax is designed to represent a wide range of units of measurement and their combinations.

It includes:

— Basic units: Simple units and prefixed units
— Composite units: Units formed by combining basic units through mathematical operations

Basic units available to AsciiUnits are sourced from UnitsDB.

UnitsDB is a comprehensive database of units of measurement and their relationships, which is used to define the units and prefixes available in AsciiUnits for various applications.
NOTE UnitsDB is a part of the UnitsML project and is maintained by CalConnect TC UNITS.

5.4. Prefixes

A prefix is a modifier to a unit that indicates a fixed factor by which the unit is multiplied.

Prefixes can be used in conjunction with units to create prefixed units.

The BIPM SI prefixes are used in AsciiUnits, which include the following:

Table 4 — SI prefixes in base 10

Prefix	Symbol	Factor	Representations				
			ASCII	MathML	HTML	LaTeX	Unicode
quetta	<mi>Q</mi>	Q	10^30	Q	Q	Q	Q
ronna	<mi>R</mi>	R	10^27	R	R	R	R
yotta	<mi>Y</mi>	Y	10^24	Y	Y	Y	Y
zetta	<mi>Z</mi>	Z	10^21	Z	Z	Z	Z
exa	<mi>E</mi>	E	10^18	E	E	E	E
peta	<mi>P</mi>	P	10^15	P	P	P	P
tera	<mi>T</mi>	T	10^12	T	T	T	T
giga	<mi>G</mi>	G	10^9	G	G	G	G
mega	<mi>M</mi>	M	10^6	M	M	M	M
kilo	<mi>k</mi>	k	10^3	k	k	k	k
hecto	<mi>h</mi>	h	10^2	h	h	h	h
deka	<mi>da</mi>	da	10^1	da	da	da	da
deci	<mi>d</mi>	d	10^-1	d	d	d	d
centi	<mi>c</mi>	c	10^-2	c	c	c	c
milli	<mi>m</mi>	m	10^-3	m	m	m	m
micro	<mi>u</mi>	u	10^-6	u	µ	mu	µ
nano	<mi>n</mi>	n	10^-9	n	n	n	n
pico	<mi>p</mi>	p	10^-12	p	p	p	p
femto	<mi>f</mi>	f	10^-15	f	f	f	f
atto	<mi>a</mi>	a	10^-18	a	a	a	a
zepto	<mi>z</mi>	z	10^-21	z	z	z	z
yocto	<mi>y</mi>	y	10^-24	y	y	y	y
ronto	<mi>r</mi>	r	10^-27	r	r	r	r
quecto	<mi>q</mi>	q	10^-30	q	q	q	q

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Table 5 — SI prefixes in binary

Prefix	Symbol	Factor	Representations				
			ASCII	MathML	HTML	LaTeX	Unicode
kibi	`<mi>Ki</mi>`	Ki	2^10	Ki	Ki	Ki	Ki
mebi	`<mi>Mi</mi>`	Mi	2^20	Mi	Mi	Mi	Mi
gibi	`<mi>Gi</mi>`	Gi	2^30	Gi	Gi	Gi	Gi
tebi	`<mi>Ti</mi>`	Ti	2^40	Ti	Ti	Ti	Ti
pebi	`<mi>Pi</mi>`	Pi	2^50	Pi	Pi	Pi	Pi
exbi	`<mi>Ei</mi>`	Ei	2^60	Ei	Ei	Ei	Ei
zebi	`<mi>Zi</mi>`	Zi	2^70	Zi	Zi	Zi	Zi
yobi	`<mi>Yi</mi>`	Yi	2^80	Yi	Yi	Yi	Yi

5.5. Dimensions

A dimension is a physical quantity that can be expressed in terms of fundamental quantities.

Basic dimensions are described with only one quantity. More complex dimensions require multiple quantities to be expressed.

Dimensions are represented in AsciiUnits using the `dim_` prefix followed by the dimension name.

The collection of dimensions is defined in the UnitsDB database. Each dimension is represented by a unique identifier in the database, of the pattern `NISTd{n}`.

5.6. Quantities

A quantity is a type of measurement that is represented by a unit.

Each quantity is associated with one or more quantity names, exactly one dimension, and one or more units of measurement.

The collection of quantities is defined in the UnitsDB database. Each quantity is represented by a unique identifier in the database, of the pattern `NISTq{n}`.

6. Architecture

6.1. General

AsciiUnits is a text-based syntax for representing units of measurement.

There are two types of representable forms in AsciiUnits:

— Units
— Values of units

6.2. Statement

An AsciiUnits statement is a sequence of characters that represents a unit or a value of a unit.

EXAMPLE

```
kg # kilogram
um # micrometer (meter + prefix of micro)
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
degK # degree kelvin
```

6.3. Units

A unit statement represented in the AsciiUnits language is composed of up to 3 types of elements in the UnitsML model structure.

— `Unit`, which represents the unit itself
— `Dimension`, which represents the dimension of the unit
— `Quantity`, which represents the quantity being described

6.4. Values of units

A value of a unit is a number that represents a measurement in that unit.

A value consists of a numerical component and a unit. For example, in “2.3 kg,” “2.3” is the value representing the quantity and “kg” is the unit.

EXAMPLE

```
1.5 m # 1.5 meters
2.3 kg # 2.3 kilograms
3.0 m/s # 3.0 meters per second
```

AsciiUnits 1.0 does not support encoding of values of units. This may be added in a future version.

7. AsciiUnits prefixes

AsciiUnits statements support the use of prefixes as standalone elements.

Prefixes elements are indicated with a prefix of `-` after the prefix code.

Syntax:

```
{prefix-code}-
```

Figure 1

Where,

`{prefix-code}`	is the prefix code
`-`	indicates that there is no unit associated with the prefix

EXAMPLE 1

AsciiUnits statement (for deci):

```
d-
```

XML:

```
<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="-1" xml:id="NISTp10_-1">
  <PrefixName xml:lang="en">deci</PrefixName>
  <PrefixSymbol type="ASCII">d</PrefixSymbol>
  <PrefixSymbol type="unicode">d</PrefixSymbol>
  <PrefixSymbol type="LaTeX">d</PrefixSymbol>
  <PrefixSymbol type="HTML">d</PrefixSymbol>
```

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Prefix>
```

EXAMPLE 2

AsciiUnits statement (for deca):

```
da-
```

XML:

```
<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="1" xml:id="NISTp10_1">
  <PrefixName xml:lang="en">deka</PrefixName>
  <PrefixSymbol type="ASCII">da</PrefixSymbol>
  <PrefixSymbol type="unicode">da</PrefixSymbol>
  <PrefixSymbol type="LaTeX">da</PrefixSymbol>
  <PrefixSymbol type="HTML">da</PrefixSymbol>
</Prefix>
```

8. AsciiUnits dimensions

8.1. General

AsciiUnits statements support the use of dimensions as standalone elements, that are not associated with any unit.

8.2. Basic dimensions

Dimensions elements are indicated with a prefix of `dim_`.

Syntax:

```
dim_{dimension-code}
```

Figure 2

Where,

`{dimension-code}` is the dimension code

EXAMPLE 1

```
dim_L       # length dimension
dim_Theta   # temperature dimension
dim_T       # time dimension
```

EXAMPLE 2

AsciiUnits statement (for length):

```
dim_L       # length dimension
```

XML:

```
<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L">
  <Length symbol="L" powerNumerator="1"/>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Dimension>
```

8.3. Dimensions with mathematical operations

Dimension elements can be expressed with mathematic operation adjustments.

The supported operations are:

— * for addition
— / for subtraction
— ^ for power
— `sqrt()` for square root

EXAMPLE 1

```
dim_L^2         # squared length dimension
dim_Theta^10    # temperature dimension to power 10
sqrt(dim_Theta) # square root of temperature dimension
```

EXAMPLE 2

AsciiUnits statement (for squared length):

```
dim_L^2
```

XML:

```
<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L2">
  <Length symbol="L" powerNumerator="2"/>
</Dimension>
```

EXAMPLE 3

AsciiUnits statement (for square root of temperature):

```
sqrt(dim_Theta)
```

XML:

```
<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_
Theta0.5">
  <ThermodynamicTemperature symbol="Theta" powerNumerator="0.5"/>
</Dimension>
```

8.4. Composed dimensions

Composed dimensions are formed by combining more than one basic dimensions through mathematical operations.

EXAMPLE 1

```
dim_Theta*dim_L^2
```

EXAMPLE 2

AsciiUnits statement (for temperature times squared length):

```
dim_Theta*dim_L^2
```

XML:

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_
L2Theta">
  <Length symbol="L" powerNumerator="2"/>
  <ThermodynamicTemperature symbol="Theta" powerNumerator="1"/>
</Dimension>
```

9. AsciiUnits units

9.1. General

AsciiUnits supports a wide range of units, including basic units, prefixed units, and composite units.

9.2. Basic units

Basic units are the fundamental units of measurement that are not derived from other units.

These units have the following characteristics:

— They are defined as individual entries in the UnitsDB database.
— Their `RootUnit` element contains a single `EnumeratedRootUnit` element, which represents the unit itself, with the `unit` attribute set to the unit name, with `powerNumerator` and `powerDenominator` set to 1.

Syntax:

```
{unit-code}
```

Figure 3

Where,

`{unit-code}`	is the unit code

EXAMPLE 1 — Examples of basic units

```
m       # meter
s       # second
Hz      # hertz
K       # kelvin
rad     # radian
```

The XML representation of an AsciiUnits basic unit is demonstrated below.

— Basic units will not have a `<Prefix>` element since they are not prefixed units.
— The `<Dimension>` element is automatically generated based on the unit code.

EXAMPLE 2 — AsciiUnits representation for `m`

AsciiUnits statement:

```
m
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu1"
dimensionURL="#NISTd1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">meter</UnitName>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
  <UnitSymbol type="HTML">m</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">m</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd1">
  <Length symbol="L" powerNumerator="1"/>
</Dimension>
```

EXAMPLE 3 — AsciiUnits representation for Hz

AsciiUnits statement:

```
Hz
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu31"
dimensionURL="#NISTd24">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">hertz</UnitName>
  <UnitSymbol type="HTML">Hz</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">Hz</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd24">
  <Time symbol="T" powerNumerator="-1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq45"
quantityType="base" dimensionURL="#NISTd24">
  <QuantityName xml:lang="en-US">frequency</QuantityName>
</Quantity>
```

9.3. Prefixed units

A prefixed unit is a unit that has a prefix attached to it, indicating a fixed factor by which the unit is multiplied.

Prefixes are defined in the UnitsDB database and are represented in the XML representation of the unit.

NOTE AsciiUnits only support prefixes defined by the BIPM in the SI Brochure, as described in Clause 5.4.

Prefixes are automatically parsed in AsciiUnits.

Syntax:

```
{prefix-code}{unit-code}
```

Figure 4

Where,

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

`{prefix-code}`	is the prefix code
`{unit-code}`	is the unit code

```
kg      # kilogram
mm      # millimeter
mW      # milliwatt
```

Figure 5

The XML representation of an AsciiUnits prefixed unit is demonstrated below.

— Prefixed units will have a `<Prefix>` element since they are prefixed.
— The `<Dimension>` element is automatically generated based on the unit and the prefix.

EXAMPLE — AsciiUnits representation for `kg`

AsciiUnits statement:

```
kg
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu2"
dimensionURL="#NISTd2">
  <UnitSystem name="SI" type="SI_base" xml:lang="en-US"/>
  <UnitName xml:lang="en">kilogram</UnitName>
  <UnitSymbol type="HTML">kg</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">kg</mi>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="gram" prefix="k"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="3" xml:id="NISTp10_3">
  <PrefixName xml:lang="en">kilo</PrefixName>
  <PrefixSymbol type="ASCII">k</PrefixSymbol>
  <PrefixSymbol type="unicode">k</PrefixSymbol>
  <PrefixSymbol type="LaTeX">k</PrefixSymbol>
  <PrefixSymbol type="HTML">k</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd2">
  <Mass symbol="M" powerNumerator="1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq2"
quantityType="base" dimensionURL="#NISTd2">
  <QuantityName xml:lang="en-US">mass</QuantityName>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Quantity>
```

9.4. Composite units

9.4.1. General

Composite units are units that are derived from two or more basic units conjoined by a mathematical operation.

Mathematical operations are represented in AsciiUnits using the following symbols:

— * for multiplication
— / for division
— ^ for exponentiation
— `sqrt()` for square root
— `()` for grouping

The XML representation of an AsciiUnits composite unit:

— Prefixed units inside the composite will have a `<Prefix>` element.
— The `<Dimension>` element is automatically generated based on the final mathematical evaluation of the involved units and prefixes.

9.4.2. Multiplication

Units are multiplied using *

EXAMPLE 1

```
kg*m    # kilogram times meter
```

EXAMPLE 2

AsciiUnits statement (for ampere times coulomb):

```
A*C
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_A.C"
dimensionURL="#D_M3I4">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">A*C</UnitName>
  <UnitSymbol type="HTML">A&#x22c5;C</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">A</mi>
      <mo>&#x22c5;</mo>
      <mi mathvariant="normal">C</mi>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="ampere"/>
    <EnumeratedRootUnit unit="coulomb" powerNumerator="1"/>
  </RootUnits>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_M3I4">
  <Mass symbol="M" powerNumerator="1"/>
  <ElectricCurrent symbol="I" powerNumerator="4"/>
```

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Dimension>
```

9.4.3. Division

Units are divided using /

EXAMPLE

AsciiUnits statement (for GHz per volt):

```
GHz//V
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_GHz.V-1"
dimensionURL="#D_L-2M-1T2I">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">GHz*V^-1</UnitName>
  <UnitSymbol type="HTML">GHz&#x22c5;V<sup>&#x2212;1</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">GHz</mi>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">V</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>1</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="hertz" prefix="G"/>
    <EnumeratedRootUnit unit="volt" powerNumerator="-1"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="9" xml:id="NISTp10_9">
  <PrefixName xml:lang="en">giga</PrefixName>
  <PrefixSymbol type="ASCII">G</PrefixSymbol>
  <PrefixSymbol type="unicode">G</PrefixSymbol>
  <PrefixSymbol type="LaTeX">G</PrefixSymbol>
  <PrefixSymbol type="HTML">G</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L-2M-
1T2I">
  <Length symbol="L" powerNumerator="-2"/>
  <Mass symbol="M" powerNumerator="-1"/>
  <Time symbol="T" powerNumerator="2"/>
  <ElectricCurrent symbol="I" powerNumerator="1"/>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Dimension>
```

9.4.4. Grouping

Parentheses define order of operations

```
K/(kg*m)    # kelvin divided by (kilogram times meter)
J/(kg*K)    # joule per (kilogram kelvin)
```

Figure 6

EXAMPLE

AsciiUnits statement (for joule per kilogram kelvin):

```
unitsml(m^(-2))
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_m-2"
dimensionURL="#D_L-2">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">m^-2</UnitName>
  <UnitSymbol type="HTML">m<sup>&#x2212;2</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <msup>
        <mrow>
          <mi mathvariant="normal">m</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>2</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L-2">
  <Length symbol="L" powerNumerator="-2"/>
</Dimension>
```

9.4.5. Exponents

Syntax:

```
{unit-code}^{exponent}
```

Figure 7

Where,

`{unit-code}`	is the unit code
`{exponent}`	is the exponent

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

EXAMPLE 1

```
m^2     # square meter
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_m2"
dimensionURL="#D_L2">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">m^2</UnitName>
  <UnitSymbol type="HTML">m<sup>&#x2212;2</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <msup>
        <mrow>
          <mi mathvariant="normal">m</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>2</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L2">
  <Length symbol="L" powerNumerator="2"/>
</Dimension>
```

Negative exponents are allowed.

EXAMPLE 2

```
kg^-2     # per kilogram
kg^(-2)   # per kilogram with group
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_kg-2"
dimensionURL="#D_M-2">
  <UnitSystem name="SI" type="SI_base" xml:lang="en-US"/>
  <UnitName xml:lang="en">kg^-2</UnitName>
  <UnitSymbol type="HTML">kg<sup>&#x2212;2</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <msup>
        <mrow>
          <mi mathvariant="normal">kg</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>2</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="gram" prefix="k" powerNumerator="-2"/>
  </RootUnits>
</Unit>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="3" xml:id="NISTp10_3">
  <PrefixName xml:lang="en">kilo</PrefixName>
  <PrefixSymbol type="ASCII">k</PrefixSymbol>
  <PrefixSymbol type="unicode">k</PrefixSymbol>
  <PrefixSymbol type="LaTeX">k</PrefixSymbol>
  <PrefixSymbol type="HTML">k</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_M-2">
  <Mass symbol="M" powerNumerator="-2"/>
</Dimension>
```

9.4.6. Square root

Using `sqrt()` function.

EXAMPLE 1 — Statement with square root on a basic unit

AsciiUnits statement (for square root of hertz):

```
sqrt(Hz)
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_Hz0.5"
dimensionURL="#D_T-0.5">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">Hz^0.5</UnitName>
  <UnitSymbol type="HTML">Hz<sup>0.5</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <msup>
        <mrow>
          <mi mathvariant="normal">Hz</mi>
        </mrow>
        <mrow>
          <mn>0.5</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_T-0.5">
  <Time symbol="T" powerNumerator="-0.5"/>
</Dimension>
```

EXAMPLE 2 — Statement with square root on a prefixed unit

AsciiUnits statement (for square root of millimeter):

```
sqrt(mm)
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_mm0.5"
dimensionURL="#D_L0.5">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">mm^0.5</UnitName>
```

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
    <UnitSymbol type="HTML">mm<sup>0.5</sup></UnitSymbol>
    <UnitSymbol type="MathMl">
      <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
        <msup>
          <mrow>
            <mi mathvariant="normal">mm</mi>
          </mrow>
          <mrow>
            <mn>0.5</mn>
          </mrow>
        </msup>
      </math>
    </UnitSymbol>
    <RootUnits>
      <EnumeratedRootUnit unit="meter" prefix="m" powerNumerator="0.5"/>
    </RootUnits>
  </Unit>

  <Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
  prefixPower="-3" xml:id="NISTp10_-3">
    <PrefixName xml:lang="en">milli</PrefixName>
    <PrefixSymbol type="ASCII">m</PrefixSymbol>
    <PrefixSymbol type="unicode">m</PrefixSymbol>
    <PrefixSymbol type="LaTeX">m</PrefixSymbol>
    <PrefixSymbol type="HTML">m</PrefixSymbol>
  </Prefix>

  <Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L0.5">
    <Length symbol="L" powerNumerator="0.5"/>
  </Dimension>
```

9.4.7. Realized units

Realized composite units are formed by combining more than one basic unit through the above mentioned operations.

EXAMPLE

AsciiUnits statement (for joule per kilogram kelvin):

```
J/kg*K
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu13.
u27p10_3e-1/1.u5e-1/1" dimensionURL="#D_L2M0T-2Theta-1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">joule per kilogram kelvin</UnitName>
  <UnitSymbol type="HTML">J&#x22c5;kg<sup>&#x2212;1</sup>&#x22c5;K<sup>
&#x2212;1</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">J</mi>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">kg</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
            <mn>1</mn>
          </mrow>
        </msup>
        <mo>&#x22c5;</mo>
        <msup>
          <mrow>
            <mi mathvariant="normal">K</mi>
          </mrow>
          <mrow>
            <mo>&#x2212;</mo>
            <mn>1</mn>
          </mrow>
        </msup>
      </math>
    </UnitSymbol>
    <RootUnits>
      <EnumeratedRootUnit unit="joule"/>
      <EnumeratedRootUnit unit="gram" prefix="k" powerNumerator="-1"/>
      <EnumeratedRootUnit unit="kelvin" powerNumerator="-1"/>
    </RootUnits>
  </Unit>

  <Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
  prefixPower="3" xml:id="NISTp10_3">
    <PrefixName xml:lang="en">kilo</PrefixName>
    <PrefixSymbol type="ASCII">k</PrefixSymbol>
    <PrefixSymbol type="unicode">k</PrefixSymbol>
    <PrefixSymbol type="LaTeX">k</PrefixSymbol>
    <PrefixSymbol type="HTML">k</PrefixSymbol>
  </Prefix>

  <Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd40">
    <Length symbol="L" powerNumerator="2"/>
    <Time symbol="T" powerNumerator="-2"/>
    <ThermodynamicTemperature symbol="Theta" powerNumerator="-1"/>
  </Dimension>

  <Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L2M0T-
  2Theta-1">
    <Length symbol="L" powerNumerator="2"/>
    <Mass symbol="M" powerNumerator="0"/>
    <Time symbol="T" powerNumerator="-2"/>
    <ThermodynamicTemperature symbol="Theta" powerNumerator="-1"/>
  </Dimension>
```

9.5. Symbol ambiguity resolution

Multiple units can be represented by the same symbol. This is often differentiated according to historical or contextual usage, and in some cases they cannot be disambiguated by the context of the unit.

EXAMPLE 1

- `m` can represent *meter* or *mole*.
- `B` can represent both *bel* (as in decibel) and *byte*.

These units that are associated with multiple symbols are considered to have:

— a primary unit symbol

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

— one or more alternative unit symbols.

To ensure unambiguous identification of units, AsciiUnits uses unique identifiers for these ambiguous symbols accross units.

When using a potentially ambiguous symbol, AsciiUnits will attempt to resolve the ambiguity based on context where possible.

A comprehensive table of ambiguous unit symbols and their disambiguated identifiers is provided in Appendix C.

In UnitsDB, there is a notion of a “prefixed unit”, which is a unit that has a prefix attached to it (marked by `prefixed: true` in the UnitsDB entry).

In case of ambiguity, the interpretation with no prefix is prioritised over the interpretation as a unit.

Exceptionally, kg is decomposed into “kilo-gram” (a unit) rather than treated as a basic unit, for consistency with other prefixes of grams.

EXAMPLE 2

`ct` is interpreted as *hundredweight*, rather than *centi-ton*.

Units with multiple symbols have those symbols encoded separately in the UnitsDB Units’ database.
NOTE While different symbols can be used to reference the same Unit in UnitsML, the default unit symbol will still apply during presentation of the unit unless explicitly specified using Clause 10.

10. Unit metadata

10.1. General

Metadata to a unit’s usage can be supplied after a comma after an AsciiUnits statement.

Syntax:

```
{unit-code}, {metadata}
```

Figure 8

Where,

`{unit-code}`	is the unit code
`{metadata}`	is the metadata

EXAMPLE

```
m, quantity: NISTq103              # meter with quantity reference
cal_th/cm^2, name: langley         # calorie thermal per square
centimeter with name
cm*s^-2, symbol: cm cdot s^-2      # custom symbol representation
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
cm*s^-2, multiplier: xx             # custom multiplier
```

10.2. Quantity

10.2.1. General

A unit of measurement is used to describe a quantity, and one unit can be used to measure different quantities.

EXAMPLE

The unit “meter” (m) can be used to measure length, wavelength, radius, height, displacement.

For interoperability between systems and value conversions, it is important to specify the quantity being measured. This ensures that the unit is used correctly and consistently in scientific and engineering contexts.

10.2.2. Default quantity

An AsciiUnit unit is associated with a default quantity.

— If a single quantity is associated with the unit in UnitsDB, that quantity is added automatically.
— Otherwise, no quantity is added unless explicitly specified as in Clause 10.2.3.

EXAMPLE 1

The unit `m` (meter) is associated with the quantities `length`, `diameter`, `length of path`, and so on in the UnitsDB database, with the `length` quantity being listed first. Therefore, the default quantity for `m` is “length”.

The XML representation of an AsciiUnits unit always includes the default quantity association in the `<Quantity>` element unless a different quantity is specified.

EXAMPLE 2

AsciiUnits statement:

```
m # meter
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu1"
dimensionURL="#NISTd1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">meter</UnitName>
  <UnitSymbol type="HTML">m</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">m</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd1">
  <Length symbol="L" powerNumerator="1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq1"
quantityType="base" dimensionURL="#NISTd1">
```

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
    <QuantityName xml:lang="en-US">length</QuantityName>
  </Quantity>
```

10.2.3. Specific quantity

AsciiUnits allows specifying the quantity a unit measures using the `quantity` metadata parameter, which provides the UnitsDB identifier for the quantity being measured.

Syntax:

```
{unit-code}, quantity: {quantity-code}
```

Figure 9

Where,

`{unit-code}`	is the unit code
`{quantity-code}`	is the reference to a quantity identifier or its shortname in the UnitsDB database

EXAMPLE 1

The AsciiUnits statement of `s, quantity: NISTq109` indicates that the second is used to measure period duration.

EXAMPLE 2 — Examples of units with quantity references

```
kg, quantity: NISTq2              # kilogram with mass quantity reference
kg, quantity: mass                # kilogram with mass quantity reference
s, quantity: NISTq3               # second with time quantity reference
s, quantity: duration             # second with time quantity reference
m, quantity: NISTq103             # meter with length quantity reference
m, quantity: position_vector      # meter with length quantity reference
l, quantity: NISTq105             # liter with volume quantity reference
l, quantity: volume               # liter with volume quantity reference
```

The XML representation of an AsciiUnits unit with quantity reference includes the specific `<Quantity>` element that provides information about the quantity being measured.

EXAMPLE 3 — AsciiUnits representation for kilogram with mass quantity reference

AsciiUnits statement:

```
kg [NISTq2]
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu1"
dimensionURL="#NISTd1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">meter</UnitName>
  <UnitSymbol type="HTML">m</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">m</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd1">
```


© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
  <Length symbol="L" powerNumerator="1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq2"
quantityType="base" dimensionURL="#NISTd1">
  <QuantityName xml:lang="en-US">mass</QuantityName>
</Quantity>
```

EXAMPLE 4 — AsciiUnits representation for meter with position vector quantity reference

AsciiUnits statement:

```
m [NISTq103]
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu1"
dimensionURL="#NISTd1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">meter</UnitName>
  <UnitSymbol type="HTML">m</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">m</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd1">
  <Length symbol="L" powerNumerator="1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq103"
quantityType="base" dimensionURL="#NISTd1">
  <QuantityName xml:lang="en-US">position vector</QuantityName>
</Quantity>
```

10.2.4. Inferred quantity

In composite units, when a quantity is not explicitly specified, AsciiUnits will attempt to infer the appropriate quantity based on the unit's dimension. This inference is based on the standard quantities defined in the UnitsDB database.

EXAMPLE 1

A unit with dimension `L` (length) will be detected to be associated with the quantity "length".

A unit with dimension `L*T^-1` will be detected to be associated with the quantity "velocity".

EXAMPLE 2 — Inferred quantity based on UnitsDB lookup

AsciiUnits statement:

```
m/s
```

Given that the unit `m/s` is a composite unit, AsciiUnits can detect the dimension `L*T^-1` (NISTd11), which allows an inference to the quantity "velocity" (NISTq12).

XML (with inferred quantity):

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTu1.u3e-1/
1" dimensionURL="#NISTd11">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">meter per second</UnitName>
  <UnitSymbol type="HTML">m&#x22c5;s<sup>&#x2212;1</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">m</mi>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">s</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>1</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="meter"/>
    <EnumeratedRootUnit unit="second" powerNumerator="-1"/>
  </RootUnits>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd11">
  <Length symbol="L" powerNumerator="1"/>
  <Time symbol="T" powerNumerator="-1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq12"
quantityType="derived" dimensionURL="#NISTd11">
  <QuantityName xml:lang="en-US">velocity</QuantityName>
  <QuantityName xml:lang="en-US">speed</QuantityName>
</Quantity>
```

Composite units that lead to a detected of dimension associated with multiple quantities will always infer the first quantity in the UnitsDB database.

NOTE Composite units that lead to a dimension that is not associated with any quantity in the UnitsDB database will not be inferred.

EXAMPLE 3 — Inferred quantity using composite units with detected dimension and quantity from UnitsDB

AsciiUnits statement:

```
cd*sr*m^(-2)
```

Given that the unit `cd*sr*m^(-2)` is a composite unit, AsciiUnits can detect the dimension `L^-2*J` (NISTd27), which allows an inference to the quantity “luminous flux” (NISTq47) or “luminance” (NISTq56), and the first detected quantity is always inferred since the exact quantity type is not specified.

XML (with inferred quantity):

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_cd.sr.m-2"
dimensionURL="#NISTd27">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">cd*sr*m^-2</UnitName>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
  <UnitSymbol type="HTML">cd&#x22c5;sr&#x22c5;m<sup>&#x2212;2</sup></
UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">cd</mi>
      <mo>&#x22c5;</mo>
      <mi mathvariant="normal">sr</mi>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">m</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>2</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="candela"/>
    <EnumeratedRootUnit unit="steradian"/>
    <EnumeratedRootUnit unit="meter" powerNumerator="-2"/>
  </RootUnits>
</Unit>

<!--
  The dimension NISTd27 is detected given the length and luminous intensity
  power numerators of L = -2 and J = 1.
-->
<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd27">
  <Length symbol="L" powerNumerator="-2"/>
  <LuminousIntensity symbol="J" powerNumerator="1"/>
</Dimension>

<!--
  The dimension NISTd27 is used in two UnitsDB quantities: NISTq47
(luminous
  flux) and NISTq56 (luminance), detection always infers the first.
-->
<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq47"
quantityType="derived" dimensionURL="#NISTd27">
  <QuantityName xml:lang="en-US">luminous flux</QuantityName>
</Quantity>
```

10.3. Symbol representation

10.3.1. General

The symbol representation of a unit can be customized using the `symbol` metadata parameter. This allows for the specification of a custom symbol representation for the unit.

10.3.2. Default symbol

The default symbol representation is the first symbol entry as associated to the unit in the UnitsDB database.

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

EXAMPLE

The unit `s` (NISTu3, second) is associated with the symbols `s`, `"_s`, and `"` in the context of time measurement.

Therefore, the default symbol for `s` is “s”.

10.3.3. Custom symbol

The custom symbol representation is specified using the `symbol` metadata parameter.

This allows for the specification of a custom symbol representation for the unit in AsciiUnits format.

A custom symbol can be assigned to an AsciiUnits unit whether it is a unit of measure or a composite unit.

Syntax:

```
{unit-code}, symbol: {custom_symbol}
```

Figure 10

Where,

`{unit-code}` is the unit code

`{custom_symbol}` is the custom symbol representation in AsciiUnits format

NOTE While the `unit-code` gives the canonical representation of the unit, `custom_symbol` is what will be rendered.

EXAMPLE 1

`unitsml(cal_th/cm^2, name: langley, symbol: La)` assigns the custom symbol `La` to the unit `cal_th/cm^2` (calorie thermal per square centimeter).

`unitsml(mm*s^-2, symbol: mm cdot s^-2)` assigns the custom symbol `mm cdot s^-2` to the unit `mm*s^-2` (millimeter per second squared).

EXAMPLE 2

AsciiUnits statement (for candela steradian per square meter), with a custom symbol representation:

```
cd*sr*m^(-2),symbol:cd cdot sr cdot m^(-2)
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_cd.sr.m-2"
dimensionURL="#D_L-2J">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">cd*sr*m^-2</UnitName>
  <UnitSymbol type="HTML">cd&#x22c5;sr&#x22c5;m<sup>&#x2212;2</sup></
UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">cd</mi>
      <mo>&#x22c5;</mo>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
        <mi mathvariant="normal">sr</mi>
        <mo>&#x22c5;</mo>
        <msup>
          <mrow>
            <mi mathvariant="normal">m</mi>
          </mrow>
          <mrow>
            <mo>&#x2212;</mo>
            <mn>2</mn>
          </mrow>
        </msup>
      </math>
    </UnitSymbol>
    <RootUnits>
      <EnumeratedRootUnit unit="candela"/>
      <EnumeratedRootUnit unit="steradian"/>
      <EnumeratedRootUnit unit="meter" powerNumerator="-2"/>
    </RootUnits>
  </Unit>

  <Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd27">
    <Length symbol="L" powerNumerator="-2"/>
    <LuminousIntensity symbol="J" powerNumerator="1"/>
  </Dimension>
```

10.4. Name

10.4.1. General

The name of a unit can be customized using the `name` metadata parameter. This allows for the specification of a custom name for the unit.

10.4.2. Default name

The default name is the first name entry as associated to the unit in the UnitsDB database.

EXAMPLE

> The unit `metric_ton` (NISTu88, metric ton) is associated with the unit names “metric ton” and “ton”. The default name for `metric_ton` is the first entry, “metric ton”.

10.4.3. Custom name

The custom name representation is specified using the `name` metadata parameter, allowing for the specification of a custom name representation for the unit.

This is especially relevant if the unit is a composite unit, as the custom name representation can be used to provide a more meaningful description of the unit, and when its name is not already available from UnitsDB.

Syntax:

```
{unit-code}, name: {custom_name}
```

Figure 11

Where,

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

`{unit-code}`	is the unit code
`{custom_name}`	is the custom name representation in AsciiUnits format
`name: NAME`	is an optional name for the unit, if one is not already available from UnitsDB.

EXAMPLE 1

`unitsml(cal_th/cm^2, name: langley)` assigns the custom name “langley” to the unit `cal_th/cm^2` (calorie thermal per square centimeter).

EXAMPLE 2

The AsciiUnits statement (for the “metric ton” unit with a custom name “ton”), with a custom name representation:

```
metric_ton, name: ton
```

XML:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTu88"
dimensionURL="#NISTd2">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">ton</UnitName>
  <UnitSymbol type="HTML">t</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant='normal'>t</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd2">
  <Mass symbol="M" powerNumerator="1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq2"
quantityType="base" dimensionURL="#NISTd2">
  <QuantityName xml:lang="en-US">mass</QuantityName>
</Quantity>
```

10.5. Multiplier

10.5.1. General

A multiplier is the conjoining symbol that is used to combine multiple units in a composite unit.

It is typically represented as a multiplication sign (×) or a dot (·). The choice of symbol can depend on the context in which the unit is used and the conventions of the field of study.

10.5.2. Default multiplier

The default AsciiUnits multiplier adopts the specified default from MathML Units, the middle dot (&#xB7) symbol.

This default multiplier symbol is rendered in various formats as:

— ASCII: a single empty space

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

— Unicode: the invisible times symbol (`⁢`)
— MathML: middle dot (`&#xB7`)
— LaTeX: a middle dot (`\cdot`)
— HTML: a non-breaking space

10.5.3. Custom multiplier

The multiplier of a unit can be specified using the `multiplier` metadata parameter.

Syntax:

```
{unit-code}, multiplier: {custom_multiplier}
```

Figure 12

Where,

`{unit-code}`	is the unit code
`{custom_multiplier}`	is the custom multiplier representation in AsciiUnits format. It accepts the following values:
`:default`	is the default multiplier.
`:space`	a spacing invisible times.
`:nospace`	a non-spacing invisible times
`custom_multiplier`	is an arbitrary UTF-8 string that can be supplied instead; it will be encoded as XML entities.

The `:space` value is rendered as:

— ASCII: a single empty space
— Unicode: the invisible times symbol (`⁢`)
— MathML: spacing invisible times (`<mo rspace='thickmathspace'>⁢</mo>`)
— LaTeX: a non-breaking space (~)
— HTML: ` ` (non-breaking space)

The `:nospace` value is rendered as:

— ASCII: no space
— Unicode: the invisible times symbol (`⁢`)
— MathML: non-spacing invisible times (`<mo>⁢</mo>`)
— LaTeX: no space
— HTML: no space

If the value is not one of the above, it will be treated as a custom multiplier, where an arbitrary UTF-8 string can be supplied. It will be rendered as:

— ASCII: the UTF-8 string
— Unicode: the UTF-8 string
— MathML: the UTF-8 string with XML entities
— LaTeX: the UTF-8 string
— HTML: the UTF-8 string

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

EXAMPLE

```
cm*s^-2, multiplier: xx             # custom multiplier
```

The `xx` character is the AsciiMath explicit multiplication sign (×). This statement will render the unit as `cm xx s^-2` in AsciiUnits format..

11. Using AsciiUnits in AsciiMath

11.1. General

AsciiUnits can be embedded within AsciiMath using the `unitsml()` expression. This provides a way to incorporate UnitsML markup within mathematical content.

The Plurimath library converts AsciiMath containing AsciiUnits into MathML that incorporates UnitsML expressions complying with https://www.w3.org/TR/mathml-units/, with unique identifiers for each distinct unit, prefix, and dimension.

Units expressions are identified in MathML as `<mtext>unitsml(...)</mtext>`, which in turn can be identified in AsciiMath as `"unitsml(…)"`.

11.2. Notation

The `unitsml(...)` expression consists of a unit string following the AsciiUnits syntax.

Syntax:

```
unitsml(unit-string{, unit-metadata})
```

Figure 13

Where,

`unit-string`	is the unit string, which can be a single unit or a combination of units, prefixes, and dimensions.
`unit-metadata`	is an optional comma-separated list of unit metadata parameters, described in Clause 10.

EXAMPLE

The AsciiMath with AsciiUnits expression for “millimeters per second squared” is represented as:

```
"unitsml(mm*s^-2)"
```

The `"…"` is an AsciiMath expression for string, which ensures backwards compatibility with AsciiMath parsers that are not UnitsML aware.

The `unitsml()` expression can take additional parameters as defined in Clause 10.

11.3. Representation in UnitsML in MathML

AsciiUnits in AsciiMath can be converted into a combination of UnitsML in MathML.

MathML unit expressions are expressed as `<mi>` (in accordance with W3C TR mathml-units) that cross-references to UnitsML definitions embedded in the XML that contains MathML.

The following MathML Units convention are adopted:

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

— inserting a spacing invisible times operator (`<mo rspace='thickmathspace'>⁢</mo>`) between any numbers (`<mn>`) and unit expressions in MathML;
— representing units in MathML as non-italic variables (`<mi mathvariant='normal'>`).

No whitespace is inserted between a number and a unit expression, when that unit expression wholly consists of punctuation.

EXAMPLE 1

`1 m` renders as *1 m*.

`1 deg_C` renders as *1 °C*.

However,

`9 arc_degree 7 arc_minute 22 arc_second` renders as *9° 7′ 22″*.

EXAMPLE 2

The AsciiUnits in AsciiMath expression:

```
9 "unitsml(C^3*A)"
```

is converted into:

```
<math xmlns='http://www.w3.org/1998/Math/MathML'>
  <mrow>
    <mn>9</mn>
    <mo rspace='thickmathspace'>&#x2062;</mo>
    <mrow xref='U_C3.A'>
      <msup>
        <mrow>
          <mi mathvariant='normal'>C</mi>
        </mrow>
        <mrow>
          <mn>3</mn>
        </mrow>
      </msup>
      <mo>&#xB7;</mo>
      <mi mathvariant='normal'>A</mi>
    </mrow>

    <Unit xmlns='http://unitsml.nist.gov/2005' xml:id='U_C3.A'
dimensionURL='#D_T3I4'>
      <UnitSystem name='SI' type='SI_derived' xml:lang='en-US'/>
      <UnitName xml:lang='en'>C^3*A</UnitName>
      <UnitSymbol type='HTML'>C<sup>3</sup> &#xB7; A</UnitSymbol>
      <UnitSymbol type='MathML'>
        <math xmlns='http://www.w3.org/1998/Math/MathML'>
          <mrow>
            <msup>
              <mrow>
                <mi mathvariant='normal'>C</mi>
              </mrow>
              <mrow>
                <mn>3</mn>
              </mrow>
            </msup>
            <mo>&#xB7;</mo>
            <mi mathvariant='normal'>A</mi>
```

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
            </mrow>
          </math>
        </UnitSymbol>
        <RootUnits>
          <EnumeratedRootUnit unit='coulomb' powerNumerator='3'/>
          <EnumeratedRootUnit unit='ampere'/>
        </RootUnits>
      </Unit>
      <Dimension xmlns='http://unitsml.nist.gov/2005' xml:id='D_T3I4'>
        <Time symbol='T' powerNumerator='3'/>
        <ElectricCurrent symbol='I' powerNumerator='4'/>
      </Dimension>

    </mrow>
  </math>
```

11.4. Representing alternative unit symbols

As described in Clause 9.5, a unit can be associated with alternative unit symbols, and can be referenced by any of those symbols.

When using AsciiUnits in AsciiMath, the unit symbol referenced in the `unitsml(...)` expression supports the use of alternative unit symbols.

It is crucial to note that if an alternative unit symbol used in the `unitsml(...)` expression, the unit symbol in the MathML representation will still be the canonical unit symbol unless explicitly specified otherwise using the syntax at Clause 10.3.

EXAMPLE

An expression like `1 unitsml(mL)` will be recognised as referring to the “microlitre” unit.

While the expression will be given under its canonical rendering of `ml` in UnitsML markup, the MathML rendering referencing that UnitsML expression will keep the notation `mL`.

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Appendix A (normative) EBNF grammar

A.1. General

This section provides a formal grammar for AsciiUnits syntax using Extended Backus-Naur Form (EBNF).

The grammar is derived from the Ruby implementation of AsciiUnits and the test suite at `unitsml-ruby/spec/unitsml/conv/xml_spec.rb`.

A.2. EBNF notation

The EBNF notation used in this document follows these conventions:

— `::=` defines a production rule
— `|` represents alternatives
— `[ ... ]` represents optional elements
— `{ ... }` represents elements that can be repeated zero or more times
— `( ... )` groups elements
— `" ... "` and `' ... '` represent literal strings
— `/* ... */` represents comments

A.3. Grammar definition

A.3.1. Top-level structure

```
/* Top-level AsciiUnits expression */
AsciiUnits ::= UnitExpression [ "," Metadata ]

/* Unit expression can be a basic unit, prefixed unit, composite unit, dimension
expression, or prefix expression */
UnitExpression ::= BasicUnit | PrefixedUnit | CompositeUnit |
DimensionExpression | PrefixExpression
```

Figure A.1

A.3.2. Basic units

```
/* Basic unit is a unit code from the UnitsDB database */
BasicUnit ::= UnitCode

/* Unit code is a sequence of letters and digits */
UnitCode ::= Letter { Letter | Digit }

/* Letter is any uppercase or lowercase letter */
Letter ::= "A" | "B" | "C" | ... | "Z" | "a" | "b" | "c" | ... | "z"

/* Digit is any decimal digit */
Digit ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

Figure A.2

A.3.3. Prefixed units

```
/* Prefixed unit is a prefix code followed by a unit code */
PrefixedUnit ::= PrefixCode UnitCode
```

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
/* Prefix code is a sequence of letters */
PrefixCode ::= Letter { Letter }

/* Prefix expression is a prefix code followed by a hyphen */
PrefixExpression ::= PrefixCode "-"
```

Figure A.3

A.3.4. Composite units

```
/* Composite unit is a combination of units with mathematical operations */
CompositeUnit ::= MultiplicationExpression | DivisionExpression |
ExponentExpression | SqrtExpression | GroupedExpression

/* Multiplication expression combines units with the * operator */
MultiplicationExpression ::= UnitTerm "*" UnitExpression

/* Division expression combines units with the / or // operator */
DivisionExpression ::= UnitTerm "/" UnitExpression | UnitTerm "//" UnitExpression

/* Exponentiation expression raises a unit to a power */
ExponentExpression ::= UnitTerm "^" Power

/* Square root expression takes the square root of a unit */
SqrtExpression ::= "sqrt(" UnitExpression ")"

/* Grouped expression is a unit expression in parentheses */
GroupedExpression ::= "(" UnitExpression ")"

/* Unit term is a basic unit, prefixed unit, or grouped expression */
UnitTerm ::= BasicUnit | PrefixedUnit | GroupedExpression | SqrtExpression

/* Power can be a positive or negative number, or a grouped expression */
Power ::= Integer | "-" Integer | "(" Integer ")" | "(" "-" Integer ")"

/* Integer is a sequence of digits */
Integer ::= Digit { Digit }
```

Figure A.4

A.3.5. Dimension expressions

```
/* Dimension expression represents a physical dimension */
DimensionExpression ::= DimensionTerm | DimensionMultiplication |
DimensionExponentiation

/* Dimension term is a basic dimension */
DimensionTerm ::= "dim_" DimensionSymbol

/* Dimension symbol is a dimension identifier */
DimensionSymbol ::= "L" | "M" | "T" | "I" | "Theta" | "N" | "J"

/* Dimension multiplication combines dimensions with the * operator */
DimensionMultiplication ::= DimensionTerm "*" DimensionExpression

/* Dimension exponentiation raises a dimension to a power */
DimensionExponentiation ::= DimensionTerm "^" Integer
```

Figure A.5

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

A.3.6. Metadata

```
/* Metadata provides additional information about the unit */
Metadata ::= MetadataItem { "," MetadataItem }

/* Metadata item is a key-value pair */
MetadataItem ::= MetadataKey ":" MetadataValue

/* Metadata key is a sequence of letters */
MetadataKey ::= Letter { Letter | "_" }

/* Metadata value is a sequence of characters */
MetadataValue ::= Character { Character }

/* Character is any printable character */
Character ::= Letter | Digit | Symbol

/* Symbol is any printable symbol */
Symbol ::= "+" | "-" | "*" | "/" | "^" | "(" | ")" | "." | "," | "_" | " " | ...
```

Figure A.6

A.4. Examples with grammar breakdown

A.4.1. Basic unit

```
m
```

Figure A.7

Grammar breakdown: * `AsciiUnits` → `UnitExpression` → `BasicUnit` → `UnitCode` → `"m"`

A.4.2. Prefixed unit

```
kg
```

Figure A.8

Grammar breakdown: * `AsciiUnits` → `UnitExpression` → `PrefixedUnit` → `PrefixCode` `UnitCode` → `"k" "g"`

A.4.3. Composite unit with multiplication

```
kg*m
```

Figure A.9

Grammar breakdown: * `AsciiUnits` → `UnitExpression` → `CompositeUnit` → `MultiplicationExpression` → `UnitTerm "*" UnitExpression` → `PrefixedUnit "*"` `BasicUnit` → `"kg" "*" "m"`

A.4.4. Composite unit with division

```
m/s
```

Figure A.10

Grammar breakdown: * `AsciiUnits` → `UnitExpression` → `CompositeUnit` → `DivisionExpression` → `UnitTerm "/" UnitExpression` → `BasicUnit "/" BasicUnit` → `"m"` `"/" "s"`

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

A.4.5. Composite unit with exponentiation

```
m^2
```

Figure A.11

Grammar breakdown: * `AsciiUnits` → `UnitExpression` → `CompositeUnit` → `ExponentExpression` → `UnitTerm "^" Power` → `BasicUnit "^" Integer` → `"m" "^" "2"`

A.4.6. Composite unit with square root

```
sqrt(Hz)
```

Figure A.12

Grammar breakdown: * `AsciiUnits` → `UnitExpression` → `CompositeUnit` → `SqrtExpression` → `"sqrt(" UnitExpression ")"` → `"sqrt(" BasicUnit ")"` → `"sqrt(" "Hz" ")"`

A.4.7. Composite unit with grouping

```
m^(-2)
```

Figure A.13

Grammar breakdown: * `AsciiUnits` → `UnitExpression` → `CompositeUnit` → `ExponentExpression` → `UnitTerm "^" Power` → `BasicUnit "^" "(" "-" Integer ")"` → `"m" "^" "(" "-" "2" ")"`

A.4.8. Unit with metadata

```
cd*sr*m^(-2),symbol:cd cdot sr cdot m^(-2)
```

Figure A.14

Grammar breakdown: * `AsciiUnits` → `UnitExpression "," Metadata` → `CompositeUnit "," MetadataItem` → `MultiplicationExpression "," MetadataKey ":" MetadataValue` → `UnitTerm "*" MultiplicationExpression "," "symbol" ":" "cd cdot sr cdot m^(-2)"`

A.5. Railroad diagrams

Railroad diagrams provide a visual representation of the grammar rules. Below are railroad diagrams for the main components of the AsciiUnits grammar:

```
@startuml
skinparam monochrome true
skinparam shadowing false
skinparam defaultFontName "Arial"
skinparam defaultFontSize 14

start
:UnitExpression;
if (Metadata?) then (yes)
  :,;
  :Metadata;
endif
stop
@enduml
```

Figure A.15 — AsciiUnits Expression

```
@startuml
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
skinparam monochrome true
skinparam shadowing false
skinparam defaultFontName "Arial"
skinparam defaultFontSize 14

start
split
  :BasicUnit;
split again
  :PrefixedUnit;
split again
  :CompositeUnit;
split again
  :DimensionExpression;
split again
  :PrefixExpression;
end split
stop
@enduml
```

Figure A.16 — Unit Expression

```
@startuml
skinparam monochrome true
skinparam shadowing false
skinparam defaultFontName "Arial"
skinparam defaultFontSize 14

start
split
  :MultiplicationExpression;
split again
  :DivisionExpression;
split again
  :ExponentExpression;
split again
  :SqrtExpression;
split again
  :GroupedExpression;
end split
stop
@enduml
```

Figure A.17 — Composite Unit

A.6. Grammar validation

The EBNF grammar defined in this section has been validated against the test suite at `unitsml-ruby/spec/unitsml/conv/xml_spec.rb`. All test cases in the suite conform to the grammar rules.

The grammar provides a formal specification of the AsciiUnits syntax, which can be used to:

— Implement parsers and generators for AsciiUnits
— Validate AsciiUnits expressions
— Document the AsciiUnits syntax in a precise and unambiguous way
— Ensure interoperability between different implementations of AsciiUnits

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Appendix B (informative) Fixed symbols

B.1. General

Some symbols used in AsciiUnits are fundamental symbols that while exist in UnitsDB as individual entries, they are considered constant entries because of the fundamental nature of their values.

B.2. Dimension symbols

Dimensions are represented by the following symbols.

Table B.1

Dimension	UnitsDB symbol	UnitsDB ID
Length	`dim_L`	`NISTd1`
Mass	`dim_M`	`NISTd2`
Time	`dim_T`	`NISTd3`
Electric current	`dim_I`	`NISTd4`
Thermodynamic temperature	`dim_Theta`	`NISTd5`
Amount of substance	`dim_N`	`NISTd6`
Luminous intensity	`dim_J`	`NISTd7`
Plane angle (dimensionless)	`dim_phi`	`NISTd9`

EXAMPLE

`unitsml(dim_I)` for the dimension of electric current, I.

B.3. Non-SI units who values in SI units that are experimentally determined

This table maps from NIST SP 811, Table 7.

Table B.2 — Non-SI Units accepted for use with the SI by the CIPM and NIST SP 811 Table 7

Name	UnitsDB symbol	UnitsDB name	UnitsDB ID
electronvolt	eV	electronvolt	`NISTu240`
astronomical unit	ua	astronomical unit	`NISTu183`
unified atomic mass unit	u	unified atomic mass unit	`NISTu241`
dalton	Da	dalton	`NISTu289`

B.4. Natural units and atomic units

Natural units and atomic units are represented by the following symbols.

This table maps from NIST SP 811, Table 8.

Table B.3

Quantity kind	UnitsDB symbol	Description	UnitsDB name	UnitsDB ID
speed	`c`	Speed of light in vacuum (n.u.)	natural unit of velocity	`NISTu242`
action	`h-bar`	Planck constant divided by 2π (n.u.)	natural unit of action	`NISTu243`
mass	`m_e`	electron rest mass (n.u. and a.u.)	natural unit of action	`NISTu244`
electric charge	`e`	elementary charge (a.u.)	atomic unit of charge	`NISTu246`
length	`a_0`	Bohr radius (a.u.)	atomic unit of length	`NISTu249`
energy	`e_h`	Hartree energy (a.u.)	atomic unit of energy	`NISTu250`
time	`h/E_h`	ratio of action to energy (a.u.)	natural unit of time	`NISTu248`

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Appendix C (informative) Ambiguous unit symbols

The symbols used for units can be highly ambiguous. In order to guarantee accurate parsing, the symbols used to data enter units are unambiguous in UnitsDB's Units database.

EXAMPLE

`B` is ambiguous between *bel* (as in decibel) and *byte*.

These units are kept unambiguous by using `bel_B` and `byte_B` to refer to them, although they will still both be rendered as `B`.

The following table lists ambiguous symbols and how they are disambiguated in the symbol IDs.

Table C.1 — Ambiguous symbols supported in UnitsDB

Symbol	Unit + ID	Unit + ID	Unit + ID	Unit + ID	Unit + ID
′	minute (minute of arc): '	foot: '_ft	minute: '_min	minute of arc: prime	foot: prime_ft
″	second (second of arc): "	second: "_s	inch: "_in	second of arc: dprime	second: dprime_s
″Hg	conventional inch of mercury: "Hg	conventional inch of mercury: dprime_Hg	inch of mercury (32 degF): "Hg_32degF	inch of mercury (60 degF): "Hg_60degF	inch of mercury (32 degF): dprime_Hg_32degF
hp	horsepower: hp	horsepower (UK): hp_UK	horsepower, water: hp_water	horsepower, metric: hp_metric	horsepower, boiler: hp_boiler
Btu	British thermal unit_IT: Btu	British thermal unit (mean): Btu_mean	British thermal unit (39 degF): Btu_39degF	British thermal unit (59 degF): Btu_59degF	British thermal unit (60 degF): Btu_60degF
a	are: a	year (365 days): a_year	year, tropical: a_tropical_year	year, sidereal: a_sidereal_year	
d	day: d	darcy: darcy	day, sidereal: d_sidereal		
inHg	conventional inch of mercury: inHg	inch of mercury (32 degF): inHg_32degF	inch of mercury (60 degF): inHg_60degF		
inH_2O	conventional inch of water: inH_20	inch of water (39.2 degF): inH_20_39degF	inch of water (60 degF): inH_20_60degF		
min	minute: min	minim: minim	minute, sidereal: min_sidereal		
pc	parsec: pc	pica (printer's): pica_printer	pica (computer): pica_computer		
t	metric ton: t	long ton: ton_long	short ton: ton_short		
B	bel: bel_B	byte: byte_B			
cmHg	conventional centimeter of mercury: cmHg	centimeter of mercury (0 degC): cmHg_0degC			
cmH_2O	conventional centimeter of water: cmH_20	centimeter of water (4 degC): cmH_20_4degC			
cup	cup (US): cup	cup (FDA): cup_label			
D	debye: D	darcy: Darcy			

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Symbol	Unit + ID	Unit + ID	Unit + ID	Unit + ID	Unit + ID
ft	foot: ft	foot (based on US survey foot): ft_US_survey			
ftH_2O	conventional foot of water: ftH_20	foot of water (39.2 degF): ftH_20_39degF			
gi	gill (US): gi	gill [Canadian and UK (Imperial)]: gi_imperial			
h	hour: h	hour, sidereal: h_sidereal			
′Hg	conventional foot of mercury: 'Hg	conventional foot of mercury: prime_Hg			
$\hbar$	natural unit of action: h-bar	atomic unit of action: h-bar_atomic			
m_e	natural unit of mass: m_e	atomic unit of mass: m_e_atomic			
in	inch: in	inch (based on US survey foot): in_US_survey			
K	kelvin: K	kayser: kayser			
L	liter: L	lambert: Lambert			

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Appendix D (normative) Test suite

D.1. General

This test suite is used for:

— Validating implementations of AsciiUnits parsers and generators
— Ensuring compatibility between different implementations
— Providing examples for users of AsciiUnits
— Serving as a reference for the expected behavior of AsciiUnits

Implementers of AsciiUnits parsers and generators should ensure that their implementations can correctly process all the examples in this test suite.

This comprehensive test suite for AsciiUnits is based on the Ruby implementation's test suite at `unitsml-ruby/spec/unitsml/conv/xml_spec.rb`.

The test suite includes a variety of AsciiUnits statements and their corresponding XML representations, serving as a reference for implementers and users of AsciiUnits.

D.2. Basic units

D.2.1. Meter (m)

AsciiUnits statement:

```
m
```

Figure D.1

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu1"
dimensionURL="#NISTd1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">meter</UnitName>
  <UnitSymbol type="HTML">m</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">m</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd1">
  <Length symbol="L" powerNumerator="1"/>
</Dimension>
```

Figure D.2

D.2.2. Hertz (hz)

AsciiUnits statement:

```
Hz
```

Figure D.3

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu31"
dimensionURL="#NISTd24">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">hertz</UnitName>
  <UnitSymbol type="HTML">Hz</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">Hz</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd24">
  <Time symbol="T" powerNumerator="-1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq45"
quantityType="base" dimensionURL="#NISTd24">
  <QuantityName xml:lang="en-US">frequency</QuantityName>
</Quantity>
```

Figure D.4

D.2.3. Radian (rad)

AsciiUnits statement:

```
rad
```

Figure D.5

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu9"
dimensionURL="#D_L0">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">radian</UnitName>
  <UnitSymbol type="HTML">rad</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">rad</mi>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd9">
  <PlaneAngle symbol="phi" powerNumerator="1"/>
</Dimension>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L0">
  <Length symbol="L" powerNumerator="0"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq9"
dimensionURL="#NISTd9" quantityType="base">
  <QuantityName xml:lang="en-US">plane angle</QuantityName>
  <QuantityName xml:lang="en-US">angle</QuantityName>
```


© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Quantity>
```

Figure D.6

D.3. Prefixed units

D.3.1. Kilogram (kg)

AsciiUnits statement:

```
kg
```

Figure D.7

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu2"
dimensionURL="#NISTd2">
  <UnitSystem name="SI" type="SI_base" xml:lang="en-US"/>
  <UnitName xml:lang="en">kilogram</UnitName>
  <UnitSymbol type="HTML">kg</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">kg</mi>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="gram" prefix="k"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="3" xml:id="NISTp10_3">
  <PrefixName xml:lang="en">kilo</PrefixName>
  <PrefixSymbol type="ASCII">k</PrefixSymbol>
  <PrefixSymbol type="unicode">k</PrefixSymbol>
  <PrefixSymbol type="LaTeX">k</PrefixSymbol>
  <PrefixSymbol type="HTML">k</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd2">
  <Mass symbol="M" powerNumerator="1"/>
</Dimension>

<Quantity xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTq2"
quantityType="base" dimensionURL="#NISTd2">
  <QuantityName xml:lang="en-US">mass</QuantityName>
</Quantity>
```

Figure D.8

D.3.2. Micrometer (um)

AsciiUnits statement:

```
um
```

Figure D.9

XML representation:

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_um" dimensionURL=
"#NISTd1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">um</UnitName>
  <UnitSymbol type="HTML">&micro;m</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant='normal'>μm</mi>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="meter" prefix="u"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="-6" xml:id="NISTp10_-6">
  <PrefixName xml:lang="en">micro</PrefixName>
  <PrefixSymbol type="ASCII">u</PrefixSymbol>
  <PrefixSymbol type="unicode">μ</PrefixSymbol>
  <PrefixSymbol type="LaTeX">$mu$</PrefixSymbol>
  <PrefixSymbol type="HTML">&micro;</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd1">
  <Length symbol="L" powerNumerator="1"/>
</Dimension>
```

Figure D.10

D.3.3. Millibar (mbar)

AsciiUnits statement:

```
mbar
```

Figure D.11

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu362">
  <UnitSystem name="not_SI" type="not_SI" xml:lang="en-US"/>
  <UnitName xml:lang="en">millibar</UnitName>
  <UnitSymbol type="HTML">mbar</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">mbar</mi>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="bar" prefix="m"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="-3" xml:id="NISTp10_-3">
  <PrefixName xml:lang="en">milli</PrefixName>
  <PrefixSymbol type="ASCII">m</PrefixSymbol>
  <PrefixSymbol type="unicode">m</PrefixSymbol>
  <PrefixSymbol type="LaTeX">m</PrefixSymbol>
  <PrefixSymbol type="HTML">m</PrefixSymbol>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd14">
  <Length symbol="L" powerNumerator="-1"/>
  <Mass symbol="M" powerNumerator="1"/>
  <Time symbol="T" powerNumerator="-2"/>
</Dimension>
```

Figure D.12

D.4. Composite units

D.4.1. Multiplication (kg*m)

AsciiUnits statement:

```
kg*m
```

Figure D.13

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_kg.m"
dimensionURL="#D_LM">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">kg*m</UnitName>
  <UnitSymbol type="HTML">kg&#x22c5;m</UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">kg</mi>
      <mo>&#x22c5;</mo>
      <mi mathvariant="normal">m</mi>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="gram" prefix="k"/>
    <EnumeratedRootUnit unit="meter"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="3" xml:id="NISTp10_3">
  <PrefixName xml:lang="en">kilo</PrefixName>
  <PrefixSymbol type="ASCII">k</PrefixSymbol>
  <PrefixSymbol type="unicode">k</PrefixSymbol>
  <PrefixSymbol type="LaTeX">k</PrefixSymbol>
  <PrefixSymbol type="HTML">k</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_LM">
  <Length symbol="L" powerNumerator="1"/>
  <Mass symbol="M" powerNumerator="1"/>
</Dimension>
```

Figure D.14

D.4.2. Division (m/s)

AsciiUnits statement:

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
m/s
```

Figure D.15

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_m.s-1"
dimensionURL="#NISTd11">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">meter per second</UnitName>
  <UnitSymbol type="HTML">m&#x22c5;s<sup>&#x2212;1</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">m</mi>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">s</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>1</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="meter"/>
    <EnumeratedRootUnit unit="second" powerNumerator="-1"/>
  </RootUnits>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="NISTd11">
  <Length symbol="L" powerNumerator="1"/>
  <Time symbol="T" powerNumerator="-1"/>
</Dimension>
```

Figure D.16

D.4.3. Exponentiation (m^2)

AsciiUnits statement:

```
m^2
```

Figure D.17

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_m2" dimensionURL=
"#D_L2">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">m^2</UnitName>
  <UnitSymbol type="HTML">m<sup>2</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <msup>
        <mrow>
          <mi mathvariant="normal">m</mi>
        </mrow>
        <mrow>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
          <mn>2</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L2">
  <Length symbol="L" powerNumerator="2"/>
</Dimension>
```

Figure D.18

D.4.4. Negative exponentiation (kg^-2)

AsciiUnits statement:

```
kg^-2
```

Figure D.19

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_kg-2"
dimensionURL="#D_M-2">
  <UnitSystem name="SI" type="SI_base" xml:lang="en-US"/>
  <UnitName xml:lang="en">kg^-2</UnitName>
  <UnitSymbol type="HTML">kg<sup>&#x2212;2</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <msup>
        <mrow>
          <mi mathvariant="normal">kg</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>2</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="gram" prefix="k" powerNumerator="-2"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="3" xml:id="NISTp10_3">
  <PrefixName xml:lang="en">kilo</PrefixName>
  <PrefixSymbol type="ASCII">k</PrefixSymbol>
  <PrefixSymbol type="unicode">k</PrefixSymbol>
  <PrefixSymbol type="LaTeX">k</PrefixSymbol>
  <PrefixSymbol type="HTML">k</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_M-2">
  <Mass symbol="M" powerNumerator="-2"/>
</Dimension>
```

Figure D.20

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

D.4.5. Square root (sqrt(hz))

AsciiUnits statement:

```
sqrt(Hz)
```

Figure D.21

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_Hz0.5"
dimensionURL="#D_T-0.5">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">Hz^0.5</UnitName>
  <UnitSymbol type="HTML">Hz<sup>0.5</sup></UnitSymbol>
  <UnitSymbol type="MathML">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <msup>
        <mrow>
          <mi mathvariant="normal">Hz</mi>
        </mrow>
        <mrow>
          <mn>0.5</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_T-0.5">
  <Time symbol="T" powerNumerator="-0.5"/>
</Dimension>
```

Figure D.22

D.4.6. Complex composite (j/kg*k)

AsciiUnits statement:

```
J/kg*K
```

Figure D.23

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_NISTu13.u27p10_3e-
1/1.u5e-1/1" dimensionURL="#D_L2M0T-2Theta-1">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">joule per kilogram kelvin</UnitName>
  <UnitSymbol type="HTML">J&#x22c5;kg<sup>&#x2212;1</sup>&#x22c5;K<sup>&#x2212;1
</sup></UnitSymbol>
  <UnitSymbol type="MathML">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">J</mi>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">kg</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
```

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
          <mn>1</mn>
        </mrow>
      </msup>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">K</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>1</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="joule"/>
    <EnumeratedRootUnit unit="gram" prefix="k" powerNumerator="-1"/>
    <EnumeratedRootUnit unit="kelvin" powerNumerator="-1"/>
  </RootUnits>
</Unit>

<Prefix xmlns="https://schema.unitsml.org/unitsml/1.0" prefixBase="10"
prefixPower="3" xml:id="NISTp10_3">
  <PrefixName xml:lang="en">kilo</PrefixName>
  <PrefixSymbol type="ASCII">k</PrefixSymbol>
  <PrefixSymbol type="unicode">k</PrefixSymbol>
  <PrefixSymbol type="LaTeX">k</PrefixSymbol>
  <PrefixSymbol type="HTML">k</PrefixSymbol>
</Prefix>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L2M0T-2Theta-
1">
  <Length symbol="L" powerNumerator="2"/>
  <Mass symbol="M" powerNumerator="0"/>
  <Time symbol="T" powerNumerator="-2"/>
  <ThermodynamicTemperature symbol="Theta" powerNumerator="-1"/>
</Dimension>
```

Figure D.24

D.5. Dimension expressions

D.5.1. Basic dimension (dim_l)

AsciiUnits statement:

```
dim_L
```

Figure D.25

XML representation:

```
<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L">
  <Length symbol="L" powerNumerator="1"/>
</Dimension>
```

Figure D.26

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

D.5.2. Dimension multiplication (dim_theta*dim_l^2)

AsciiUnits statement:

```
dim_Theta*dim_L^2
```

Figure D.27

XML representation:

```
<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L2Theta">
  <Length symbol="L" powerNumerator="2"/>
  <ThermodynamicTemperature symbol="Theta" powerNumerator="1"/>
</Dimension>
```

Figure D.28

D.6. Units with metadata

D.6.1. Unit with symbol metadata

AsciiUnits statement:

```
cd*sr*m^(-2),symbol:cd cdot sr cdot m^(-2)
```

Figure D.29

XML representation:

```
<Unit xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="U_cd.sr.m-2"
dimensionURL="#D_L-2J">
  <UnitSystem name="SI" type="SI_derived" xml:lang="en-US"/>
  <UnitName xml:lang="en">cd*sr*m^-2</UnitName>
  <UnitSymbol type="HTML">cd&#x22c5;sr&#x22c5;m<sup>&#x2212;2</sup></UnitSymbol>
  <UnitSymbol type="MathMl">
    <math xmlns="http://www.w3.org/1998/Math/MathML" display="block">
      <mi mathvariant="normal">cd</mi>
      <mo>&#x22c5;</mo>
      <mi mathvariant="normal">sr</mi>
      <mo>&#x22c5;</mo>
      <msup>
        <mrow>
          <mi mathvariant="normal">m</mi>
        </mrow>
        <mrow>
          <mo>&#x2212;</mo>
          <mn>2</mn>
        </mrow>
      </msup>
    </math>
  </UnitSymbol>
  <RootUnits>
    <EnumeratedRootUnit unit="candela"/>
    <EnumeratedRootUnit unit="steradian"/>
    <EnumeratedRootUnit unit="meter" powerNumerator="-2"/>
  </RootUnits>
</Unit>

<Dimension xmlns="https://schema.unitsml.org/unitsml/1.0" xml:id="D_L-2J">
  <Length symbol="L" powerNumerator="-2"/>
  <LuminousIntensity symbol="J" powerNumerator="1"/>
```


© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

```
</Dimension>
```

Figure D.30

© The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved

Bibliography

[1] unitsml/unitsml-ruby, — the Ruby implementation of AsciiUnits from which the grammar was extracted, and the reference parser validated against it

[2] unitsml/asciiunits, — the PARG grammar artifact and conformance corpus repository

 © The Calendaring and Scheduling Consortium, Inc. 2025 – All rights reserved