Units and Dimensions in OpenMath

James H. Davenport[1] and William A. Naylor[2]

[1]Department of Computer Science
University of Bath
Bath BA2 7AY
England
[2]Victoria University of Wellington
New Zealand

Abstract

We first review the current state of units and dimensions in OpenMath, and compare it with the state of play in MathML [7]. We then discuss the fundamental unsolved question: how does one write "2 grammes" in OpenMath? We present the various solutions, and recommend one to the OpenMath community for debate and decision. We then do the same for the follow-up question: how does one write "2 kilometres" in OpenMath? We conclude with a set of principles to guide the evolution of the definition and use of units and dimensions in OpenMath.

Contents

1 Units in OpenMath: the current state

The current[1] state of units and dimensions in OpenMath is given in the following Content Dictionaries:

1. `dimension1.ocd` — version 3 dated 2002-09-17;
2. `units_metric1.ocd` — version 2 dated 2000-08-13;
3. `units_imperial1.ocd` — version 2 dated 2000-08-13;

and the corresponding STS [4] files.

There is no architecture document, but the following general principles were followed, and can be reconstructed from the files.

1.1 Dimensions

1. The ordinary symbols from `arith1.ocd` are used to compose dimensions, e.g. area is defined with the following Formal Mathematical Property:

```
<FMP><OMOBJ>
  <OMA>
    <OMS cd="relation1" name="eq"/>
    <OMS cd="dimensions1" name="area"/>
    <OMA>
      <OMS cd="arith1" name="times"/>
      <OMS cd="dimensions1" name="length"/>
      <OMS cd="dimensions1" name="length"/>
    </OMA>
  </OMA>
</OMOBJ></FMP>
```

   This seems compatible with the traditional usage in dimensional analysis. One could argue that this is not the "normal" meaning of this `times`, but Occam's razor would operate against a new symbol. Though not binding, we note that MathML [7, section 3] appears to recommend the use of their standard `times` content symbol for multiplication of units[2].
2. The names of dimensions are used as types in the OpenMath Small Type System [4] for units, as in the following signature for `metre`.

[1] As at 1 July 2003.

[2] MathML seems to have no particular concept of dimension, apart from a few words in [7, section 6.1 and Appendix C].

```
<Signature name="metre" >
<OMOBJ>
  <OMS cd="dimensions1" name="length"/>
</OMOBJ>
</Signature>
```

This is clearly sensible, and in line with the rôle of dimensions in physics.

1.2 Metric Units

1. There are no formal or commented mathematical properties in this file. This is probably due to the absence of any agreed mechanism for attaching units to quantities, or whether it would be legal OpenMath to write, for example, the definition of square metre[3], as in table 1. The authors believe that this is legal OpenMath.

Table 1: Putative definition of square metre
One possible definition of a square metre to go into `units_metric1`.

```
<FMP><OMOBJ>
  <OMA>
    <OMS cd="relation1" name="eq"/>
    <OMS cd="units_metric1" name="metre_sqrd"/>
    <OMA>
      <OMS cd="arith1" name="times"/>
      <OMS cd="units_metric1" name="metre"/>
      <OMS cd="units_metric1" name="metre"/>
    </OMA>
  </OMA>
</OMOBJ></FMP>
```

2. The `Newton_per_sqr_metre` is defined, but not its "proper" name, the `Pascal`. *Now fixed:* `Newton_per_sqr_metre` *retained for compatibility*[4]*, but deprecated.*
3. There are inconsistencies on the capitalisation of unit names. Those that are directly people's names are capitalised (in accordance with the principles in [3]), but those that are merely based on them (e.g. `volt` from Volta, or `amp` from Ampère) are not, even though the abbreviation is in fact capitalised. This needs to be checked with the appropriate standards. **JHD**

[3] Whether one should define such a compound unit is a different matter. MathML (see item 3 in section 2) would not, and indeed leaving it as `metre`^{2} is simpler for renderers and reasoners. This question is asked as 5.4 in section 5.

[4] The OpenMath standard [1] forbids removing symbols from CDs, but, since the `units_metric1` CD is not yet official, this ban probably does not apply.

1.3 Imperial Units

1. Despite the idea that FMPs could be used to do conversion, there are none in this file. Probably this is due to the problem described in item 1 in the previous section.
2. Curiously, the `mile` is defined, but not the `foot`, which is now[5] the standard of length, being fixed at 0.3048m.

1.4 U.S. Units

Nothing has so far been written on this subject apart from a note from JHD saying that it would be the existence of this CD (read "namespace") that distinguishes

```
<OMS name="pint" cd="units_imperial1"/>
```

from

```
<OMS name="pint" cd="units_us1"/>
```

We should note that this taxonomy is the other way up from that of MathML — see item 5 in section 2.

2 Units in MathML (Content)

It is beyond the scope of this document to consider the presentation of units in general, though the authors see no real problems with adapting to the presentation view of units in [7].

The main points relevant to units in content MathML in [7] are given below.

1. "In expressing a quantity with units, ..., it is recommended that the unit be the last child of of an `apply` element which has the `times` element as its first child." [7, section 3]

   This would seem to imply that 3.4m would be represented as

   ```
   <apply>
     <times/>
     <cn> 3.4 </cn>
     <csymbol> m </csymbol>
   </apply>
   ```

[5] And has been since the time of Henry II, the first King of England to demonstrate a knowledge of the Central Limit Theorem.

2. "It would also be best if compound units are kept separate as a nested apply at the end of a product." [7, section 3]

   This would seem to imply that 3.4m/sec would be represented as

```
<apply>
  <times/>
  <cn> 3.4 </cn>
  <apply>
    <divide/>
    <csymbol> m </csymbol>
    <csymbol> sec </csymbol>
  </apply>
</apply>
```

   The use of `<divide/>` is hypothetical here, but see the next point.

3. "The `csymbol` element should be used to identify units. It should not be used to encapsulate a compound units (*sic*) as in the following two representations of the compound unit of speed centimeters per second.

```
<csymbol>centimeters per second</csymbol>
<csymbol>cm/s</csymbol>
```

   Such compound units should be represented by explicit products or quotients of simple units." [7, section 5]

   We note though, that in [7, section 6.2], the authors use a `csymbol` to denote "watt", so the intention seems to be to bar those compound units that are comprised of multiple "recognised" units, not all those that can be reduced to basic units. This means that the Content MathML expressions for "1Nm" and "1W" would presumably be rather different, as in table 2.

4. MathML seems to regard, say, `centimetre` (`cm`) as a separate unit, not a derivation from `metre` (`m`), though this may only be in the context of the CGS system [7, section 6.5.1]. At other times it uses the fragment identifier on URLs, as in

```
<csymbol definitionURL='http.../units/pascal#k'>kPa</csymbol>
```

   (*loc.cit.*), though this is wrapped in a complex `semantics` encoding which also includes (our re-formatting)

```
<annotation-xml encoding='MathML'
                definitionURL='http.../SI-conversion-factor'>
<cn type='integer'>1000</cn>
</annotation-xml>
```

Table 2: Putative MathML for “1Nm” and “1W”

```
<apply>
 <times>
 <cn> 1 </cn>
 <apply>
  <times>
  <csymbol definitionURL="...">
    N</csymbol>
  <csymbol definitionURL="...">
    m</csymbol>
 </apply>
</apply>
```

```
<apply>
 <times>
 <cn> 1 </cn>
 <csymbol definitionURL="...">
  W</csymbol>
</apply>
```

This seems to imply that the meaning of `k` is local to this piece of MathML, which seems obscure.

5. MathML takes the approach of naming the unit, and then the variant, as in [7, section 5.3.4] `http://.../units/mile/gb-sct` for the Scottish mile, or `http://.../units/mile/survey/us`. In the latter case, the authors would suggest that the OpenMath name should be

   ```
   <OMS name="survey_mile" cd="units_us1"/>
   ```

   thus placing the system higher up the hierarchy than the name. The first example is more problematic, but could be represented by

   ```
   <OMS name="scottish_mile" cd="units_british_historic"/>
   ```

3 The Fundamental Question — Attribution

How should we signify that a quantity (ultimately numeric, but which may be a symbolic expression) is attributed with a unit? More specifically, since

```
<OMI> 2 </OMI>
```

is the OpenMath for the number 2, and

```
<OMS name="gramme" cd="units_metric1"/>
```

is the OpenMath for a gramme, what is the OpenMath for ”2 grammes”? There seem to be three options.

1. Use the `times` from `arith1`. Then the encoding for 2 grammes would be:

```
<OMA>
  <OMS name="times" cd="arith1"/>
  <OMI> 2 </OMI>
  <OMS name="gramme" cd="units_metric1"/>
</OMA>
```

   pro Aids MathML compatibility and conversion — see item 1 in section 2.

   pro Requires no extra symbols, and indeed means that a system that does not fully understand units can at least manipulate them as symbolic variables. [6]?

   con Somewhat of a perversion of this symbol, since what is going on is not multiplication in any homogeneous sense. Since this `times` is declared to be `nassoc` [4], this means that the result of multiplying two Newtons by three metres is

```
<OMA>
  <OMS name="times" cd="arith1"/>
  <OMI> 2 <\OMI>
  <OMS name="Newton" cd="units_metric1"/>
  <OMI> 3 <\OMI>
  <OMS name="metre" cd="units_metric1"/>
</OMA>
```

   whereas the OpenMath equivalent of the MathML way of representing this (see table 2) would be as follows.

```
<OMA>
  <OMS name="times" cd="arith1"/>
  <OMI> 2 <\OMI>
  <OMI> 3 <\OMI>
  <OMA>
    <OMS name="times" cd="arith1"/>
    <OMS name="Newton" cd="units_metric1"/>
    <OMS name="metre" cd="units_metric1"/>
  </OMA>
</OMA>
```

2. Using a different symbol, with different semantics, e.g. possibly only binary. It is not totally clear what the signature would be: probably

$$\texttt{NumericalValue} \times \texttt{Dimension} \rightarrow \texttt{DimensionedValue}.$$

   Possibly the best choice might be

```
<OMS name="times" cd="units_ops"/>
```

pro Unambiguous, and systems that can't understand units would not support this operator.
con Occam's razor, and the greater complexity that a new symbol, which would need to be widely understood, would bring to OpenMath.

3. Use OpenMath attributes. Then the encoding for the problem that started this section would be as follows.

```
<OMATTR>
  <OMATP>
    <OMS name="quantity_with_units" cd="units_ops"/>
    <OMS name="gramme" cd="units_metric1"/>
  </OMATP>
  <OMI> 2 </OMI>
<\OMATTR>
```

pro Attributes are already built into OpenMath.
con Attributes can be silently discarded by applications, which seems dangerous for something as vital, and mission-crashing, as units.

Without wishing to prejudge the debate in the OpenMath community, the rest of this document, and the associated CDs, use option 1 above.

4 Unit Prefixing

The SI system[6] has a uniform scheme for prefixing units to create larger or smaller one, generally in multiples of 10^3. How can we apply this scheme uniformly in OpenMath?

As in the previous section, there are various possibilities.

1. Treat them as separate units, as MathML sometimes seems to do: see item 4 in section 2.

   pro No new mechanisms involved.
   con Very verbose: each unit would also acquire 20 variants.
   con Problems of consistency.

2. As MathML (see item 4 in section 2, or [7, section 5.3.5]) does, use the XML 'prefix' notation: `#`. Hence in MathML, the kiloPascal `kPa` has a `definitionURL` of `http://.../units/pascal#k`. Of course, strictly speaking this is now a URI reference, rather than a URL, but this seems to be becoming common in MathML. More seriously, OpenMath already uses the prefix notation to distinguish individual symbols in a CD. Hence this is not a viable option for OpenMath.

[6] And others. In particular there is the IEC scheme for binary prefixes [7, Appendix B.2].

pro None that I can see for OpenMath, as opposed to MathML.

con Unusable[7] in OpenMath.

con Contrary to the OpenMath philosophy [1] of semantic extensibility through CDs, requires every application to understand this `#` syntax.

con The 10^{-6} prefix, μ, has to be replaced by `u`, which means that faithful rendering needs an extra rule to translate `u` into μ.

con The connection between the syntax `#k` and the semantics of $\times 10^3$ seems to depend on the MathML-generator generating the correct combination of `definitionURL` and `annotation-xml` attributes.

3. Use of `<OMS name="times" cd="arith1"/>`, as was suggested in the previous section for attributing units.

   pro No extra syntax or semantics required, so that a kiloPascal would be as follows.

   ```
   <OMA>
     <OMS name="times" cd="arith1"/>
     <OMS name="kilo" cd="units_siprefix1"/>
     <OMS name="Pascal" cd="units_metric1"/>
   </OMA>
   ```

   con A fairly simplistic simplifier, which knew about `<OMS name="times" cd="arith1"/>`, might multiply kW by km to get

   ```
   <OMA>
     <OMS name="times" cd="arith1"/>
     <OMS name="metre" cd="units_metric1"/>
     <OMS name="Watt" cd="units_metric1"/>
     <OMA>
       <OMS name="power" cd="arith1"/>
       <OMS name="kilo" cd="units_siprefix1"/>
       <OMI> 2 </OMI>
     </OMA>
   </OMA>
   ```

   con There is no guarantee that prefixes will in fact be prefixes, and certainly no rule to enforce this.

4. A special operator, such as `<OMS name="prefix" cd="units_ops1"/>`, whose signature would be as given in table 3. We recall that, by the rules of the Small Type System [4], this means that the first argument must have type `unit_prefix` (i.e. must be a prefix), the second argument can be anything (the human reads "something of an unknown dimension") but must return something of the *same* species (the human being reads "same dimension"). For example the prefixed unit *millisecond* could be specified by the markup:

[7] This has been checked with David Carlisle.

Table 3: Proposed signature of the "prefix" operation

```
<Signature name="prefix" >
<OMOBJ>
  <OMA>
    <OMS name="mapsto" cd="sts"/>
    <OMS cd="units_sts" name="unit_prefix"/>
    <OMV name="dimension"/>
    <OMV name="dimension"/>
  </OMA>
</OMOBJ>
</Signature>
```

```
<OMA>
  <OMS name="prefix" cd="units_ops1"/>
  <OMS name="milli" cd="units_siprefix1"/>
  <OMS name="second" cd="units_time1"/>
</OMA>
```

pro The order of the arguments ensures that, at least locally, prefixes are genuinely prefixes.

pro Prefixes do not complicate any type-checking and consistency mechanism, as a result of that signature.

con An extra operator, and indeed an extra CD `units_sts` to hold the necessary type.

5. Attributes, as discussed in solution 3 in section 3.

   pro Attributes are already built into OpenMath.

   con Attributes can be silently discarded by applications, which seems even more dangerous than discarding units as a whole: what application would notice that a "Mega" had silently vanished?

Without prejudicing the conclusions of the OpenMath community, we will go with option 4 in this document.

5 Other Points for OpenMath

5.1 Nomenclature

Should not `units_metric1` be renamed `units_si1`, thus allowing for CDs such as `units_cgs1`, as well as for a `units_metricmisc1`, which might contain the (metric) horsepower, the

hectare[8], and so on.

5.2 Formal Mathematical Properties

The "defining mathematical property", as discussed in Pisa on 28–29.9.2002 should be resurrected[9], and applied to ensure that all non-fundamental dimensions were defined, possibly indirectly[10], in terms of fundamental ones. For example, the formal mathematical property in the definition of `area` discussed in item 1 of section 1.1 should be such a defining mathematical property. **JHD**[9]

5.3 Particular Dimensions

5.3.1 The STS of dimensions

At the moment, all dimensions have an STS entry as follows.

```
<OMV name="PhysicalDimension"/>
```

Might it not make more sense to make this an `OMS`, or possibly two `OMS`s — one for fundamental[11] dimensions and one for derived dimensions. This distinction might help reasoners about units to know when no further reduction was possible.

5.3.2 The Electrical Dimensions

These have no formal mathematical properties, and the commented ones are circular. There needs to be a proper investigation of the state of the SI system [5] in this area, and the correct base dimension[12] chosen and the rest reduced to it. **JHD**

5.3.3 Concentration

The current CMP defines it as $\frac{\texttt{mass}}{\texttt{length}^3}$, which would mean that `mole` would not be a unit of concentration. Concentration could be regarded as 'quantity/volume', but this poses the question "what is 'quantity'". This needs investigating. **JHD?**

[8] Unless we introduce the `are` (little used in its own right) as an alternative for $\texttt{metres}^2$, and rely on prefixing (see section 4) to generate the hectare for us. However, might it not generate the "hectoare" instead?

[9] With help from MK, who said in Pisa that he had a solution.

[10] Note that, in the Pisa proposal, the author(s) of defining mathematical properties must ensure that the system is not circular.

[11] In theory, of course, the choice of fundamental dimensions is arbitrary. However, convention has fixed a set, and this choice would probably be less arbitrary than some choices of branch cuts which OpenMath has already made [2].

[12] [7, section 6.1] implies that this is current.

5.4 Compound Units

By "compound unit", we mean a unit that does not have a specific name in [5], but is built up from other units. Should OpenMath, as it currently does, define compound units such as `metres_sqrd`, or leave them as, for example, `metres`^{2}, as MathML does (see item 3 in section 2)? The authors are inclined to vote against compound units, on the ground that they add little, and complicate life for renderers.

5.5 Particular Units

5.5.1 `units_time1`

Various other kinds of year, such as the sidereal year, and month, such as the lunar month, need to be defined. **JHD**

5.6 Currency

No thought has yet been given in OpenMath to currency, either with fixed rates ("100 cents = 1 dollar")[13], currently fixed rates ("6.55957 French Francs = 1 Euro") or variable rates. The first seems to fit within the current paradigm, but it would probably be unwise to press ahead with this without an eye to the broader picture.

6 Questions for MathML

1. [7, section 3] (see item 2 in section 2) should perhaps explicitly say that the first child of the `apply` defining the compound unit should be one of `times`, `divide` (as in "metres per second") or `power`.
2. Does MathML (Content) really wish to continue with defining "m" (metre) and "cm" (centimetre) as different units. See OpenMath's view in section 4.

7 Conclusion

The following principles are used in constructing the existing dimensions and units CDs, and should be followed in any additions to them.

1. The ordinary symbols from `arith1.ocd` are used to compose dimensions, as in point 1 of section 1.1.

[13] Or the more unusual, such as "3 English marks = 2 pounds Sterling". One might be tempted to class "20 guineas = 21 pounds Sterling" in this category, but in fact this is only true since Sir Isaac Newton was Warden of the Royal Mint: prior to that, the guinea floated against the pound Sterling.

2. The names of dimensions are used as types in the OpenMath Small Type System [4] for units, as in point 2 of section 1.1.
3. It is legal to declare that two dimensions, or dimensional expressions, are equal directly, as in item 1 of section 1.1.
4. It is legal to declare that two units, or unit expressions, are equal directly, as in item 1 of section 1.2. Of course, if the equivalence is not one-for-one, then one needs to use the mechanism of section 3 to say that, say, one acre is 4840 square yards.
5. The conclusion of section 3.
6. The conclusion of section 4.
7. The conclusion of section 5.4.

References

[1] Caprotti, O., Carlisle, D.P. and Cohen, A.M. *The OpenMath Standard*. `http://www.openmath.org/std`

[2] Corless, R.M., Davenport, J.H., Jeffrey, D.J. and Watt, S.M. According to Abramowitz and Stegun. *ACM SIGSAM Bulletin 30(2)* 2000, 58–65.

[3] Davenport, J.H. On Writing OpenMath Content Dictionaries. *ACM SIGSAM Bulletin 30(2)* 2000, 12–15.

[4] Davenport, J.H. A Small OpenMath Type System. *ACM SIGSAM Bulletin 30(2)* 2000, 16–21.

[5] IEEE/ASTM SI 10-1997 *Standard for the Use of the International System of Units (SI): The Modern Metric System*. IEEE Inc., New York, 1997.

[6] Khanin, R. Dimensional Analysis in Computer Algebra. In *Proceedings ISSAC 2001* (2001), B. Mourrain, Ed., ACM, N. York, pp. 201–208.

[7] World-Wide Web Consortium (ed. D.W. Harder and S. Devitt), *Units in MathML*. W3C Note, 1 July 2003. `http://www.w3.org/Math/Documents/Notes/units.xml`.

A Changes made to the CDs

This appendix documents the changes made to the Content Dictionaries by JHD in the process of writing this document.

A.1 dimensions1.ocd

1. FMP added for `speed`, matching the CMP. This was not done for `velocity`, since it's not clear how to specify this formally. Clarified, in the "Description" field, the difference between speed and velocity. An FMP was added to `velocity` stating that "speed is the vector norm of velocity".
2. In the CMP for `acceleration`, "metres" was changed to "length". An FMP was added, based on that for `speed`.
3. FMP added for `force`, as

```
<FMP><OMOBJ>
  <OMA>
    <OMS cd="relation1" name="eq"/>
    <OMS cd="dimensions1" name="force"/>
    <OMA>
      <OMS cd="arith1" name="times"/>
      <OMS cd="dimensions1" name="mass"/>
      <OMS cd="dimensions1" name="acceleration"/>
    </OMA>
  </OMA>
</OMOBJ></FMP>
```

   In general, dimensions were related to their logical predecessors, rather than to the base dimensions.
4. FMPs added for `pressure`, `density`[14], `energy` (along with matching CMP) and `momentum`.
5. Added the dimension `power`, with an FMP and corresponding STS.

A.2 units_metric1.ocd

1. Enhanced opening comments to point out that it is SI units that are defined in this file, not all metric units. Similarly, changed "metric" to "SI" where appropriate
2. Said that the `second` here was the same as the `second` in `units_time1`, the general time CD.
3. Enhanced description for `gramme` to point out that, although kilogramme is the SI unit of mass, we take `gramme`, otherwise the gramme would be the milli-kilogramme! Stated that it was a basic unit of SI.
4. Gave `metre_sqrd` and `litre` FMPs in terms of `metre`.

[14] Or should this be specific gravity?

5. Is the litre SI? Yes, it is, but [7, Section 5.3.3], it changed definition (by about 0.0028%) in 1964. Therefore added, with a CMP but without a FMP, `litre_pre1964`[15] and suitable clarifying text.
6. Added `Joule` and `Watt`, with the corresponding STS.

A.3 `units_metric1.sts`

1. Changed the signature for `pint` (*sic*) to the obviously-intended `litre`.
2. Changed the STS for `metres_per_second` from `velocity` to `speed`, to conform with the CMP in the `.ocd` file.
3. Added `Pascal`, defined the existing `Newton_per_sqr_metre` to be equivalent to it (as an FMP), but deprecated in OpenMath.

A.4 `units_imperial1.ocd`

1. Added the `foot`, since this is the base unit, and defined (currently in a CMP and an FMP) as 0.3048 metres. Gave the mile a CMP and an FMP in terms of the foot. Also added the STS.
2. Added comments to the `mile` to point out that it was the land, or statute, mile.
3. Added `yard`, with a CMP and FMP in terms of the `foot`, and associated STS.
4. Gave `acre` a CMP and FMP in terms of square yards.
5. Added comment to `pint` to cross-reference to `pint` in `units_us1` (when written).
6. Added CMP and FMP to `miles_per_hr` and `miles_per_hr_sqrd`.

A.5 `units_time1.ocd`

A new CD (and corresponding STS file), since time is common to, at least, SI and Imperial systems of units, and probably also the US one, but this needs checking. **JHD**

1. `second` is defined as the fundamental unit.
2. `minute`, `hour`, `day` and `week` are derived in turn. The definition of `day` explicitly states that it does not take account of leap seconds.
3. `month` and `year` are more complicated. There are a variety of definitions, and the calendar definitions are not constant. We therefore explicitly define `calendar_month` and `calendar_year`, and will in the future need to define other variants. `calendar_month`

[15] [7, Section 5.3.3] calls this `liter/1901`, which is indeed the date when the previous definition was adopted, but seems less helpful than saying when it ceased to be the definition.

has the following FMP, and `calendar_year` an equivalent one, as well as one saying that a `calendar_year` is precisely[16] 12 `calendar_month`s.

```
<FMP><OMOBJ>
<OMA>
  <OMS name="in" cd="set1"/>
  <OMA>
    <OMS name="divide" cd="arith1"/>
    <OMS name="calendar_month" cd="units_time1"/>
    <OMS name="day" cd="units_time1"/>
  </OMA>
  <OMA>
    <OMS cd="interval1" name="integer_interval"/>
    <OMI> 28 </OMI>
    <OMI> 31 </OMI>
  </OMA>
</OMA>
</OMOBJ></FMP>
```

A.6 units_siprefix1.ocd

This contains all the SI prefixes. Note that the powers of 10 involved are explicit, rather than being a series of zeroes, or, worse, an approximate floating point number.

A.7 units_binaryprefix1.ocd

This contains all the IEC binary prefixes. Note that the powers of 2 involved are explicit, rather than being a series of digits, prone to error.

A.8 units_ops.ocd

So far, this only contains the `prefix` operator, as defined in item 4 in section 4.

A.9 units_sts.ocd

So far, this only contains the type `unit_prefix`, the type of all unit prefixes, and which is used in the STS for `<OMS name="prefix" cd="units_ops"/>`. The STS for `unit_prefix` describes it as being of type

[16] One could argue about the appropriateness of this. After all, 12 Februarys are not a calendar year. But there seems no way of expressing "12 consecutive months", and the proposition is valid in that one year can always be converted into twelve months.

```
<OMS name="Object" cd="sts"/>
```